Adding a Proxy Service

A proxy service defines virtual services hosted on the ESB that can accept requests, mediate them, and deliver them to an actual service. A proxy service can perform transport or interface switching and expose different semantics from the actual service: WSDL, policies, and QoS aspects like WS-RM, WS-Security, etc. It can mediate the messages before they are delivered to the actual endpoint and mediate the responses before they reach the client. You can also list a combination of tasks to be performed on the messages received by the proxy service and terminate the flow or send a message back to the client even without sending it to the actual service.

Follow the instructions below to add a new proxy service.

1. Sign in. Enter your user name and password to log on to the ESB Management Console.

2. Click on "Main" in the left menu to access the "Manage" menu.

3. In the "Manage" menu, click on "Proxy Service" under "Web Services."

4. The "Create Proxy Service from Template" page appears.
5. Select a template that suits your mediation requirements:

- **Pass Through Proxy** - Creates a simple proxy service on a specified endpoint. The proxy service does not perform any processing on the messages but simply forwards them to the back-end service.
- **Secure Proxy** - Creates a proxy service with WS-Security engaged. The service will process WS-Security on incoming requests and forward them to an unsecured back-end service.
- **WSDL Based Proxy** - Creates a proxy service based on the WSDL of an existing web service. Endpoint information is extracted from a remotely hosted WSDL of an actual service.
- **Logging Proxy** - Creates a proxy service which logs all the incoming requests and forwards them to a given endpoint. If necessary responses coming back from the back-end service can be logged before routing them to the client.
- **Transformer Proxy** - Creates a proxy service which transforms all the incoming requests using XSLT and then forwards them to a given endpoint. If required responses coming back from the back-end service can be transformed as well.
- **Custom Proxy** - Launches the proxy service creation wizard and creates a new proxy service from scratch. Customize each and every aspect of the proxy including sequences, endpoints, transport, and other QoS settings.

For example, if you want to expose an existing service with WS-Security, you can choose the "Secure Proxy" option to create a proxy service with WS-Security enabled with a specified security policy. If you implement a scenario where an existing service is exposed over a different schema (message format), you can use the "Transformer Proxy" option to easily set up a proxy service that transforms requests and responses based on specified XSLT configurations.

6. Specify the options for the selected template as described in the following topics:

- **Pass Through Proxy Template**
- **Secure Proxy Template**
- **WSDL Based Proxy Template**
- **Logging Proxy Template**
- **Transformer Proxy Template**
- **Custom Proxy Template**

7. Specify an option for publishing a WSDL for this proxy service.

8. Select the transports this proxy service will use. Only enabled transports are available in the list.

9. Click **Create** or **Finish** at the bottom of the page to create the proxy service. It now appears in the Proxy Service list on the Deployed Services screen.
Publish WSDL Options

When you create a proxy service, a default WSDL is generated for it automatically. However, the default WSDL only shows the `mediate` operation and does not provide the client with any information about the message format or parameters expected by the back-end service. To provide clients with this information, you can publish a custom WSDL based on the back-end service's WSDL, or if you don't want to publish all the operations of the back-end service, you can publish a modified version of the WSDL. For example, if the back-end service expects a message to include the name, department, and permission level, and you want the proxy service to inject the permission level as it processes the message, you could publish a WSDL that includes just the name and department and omits the permission level parameter, since that data will be added by the proxy service.

To publish a custom WSDL for this proxy service, select one of the "Publishing WSDL" options:

- Specify in-line
- Specify source URL
- Pick from registry

If the WSDL definition includes dependencies on other resources that are in the registry, add those resources.

**Specify in-line**

Enter the WSDL definition in the text area.

**Specify Source URL**
Enter the URI of the WSDL in the text box, and then click **Test URI** to ensure it is available. A URI consists of a URL and URN. The URL defines the host address of the network resource (can be omitted if resources are not network homed), and the URN defines the resource name in local “namespaces.”

For example: URI = ftp://ftp.dlink.ru/pub/ADSL


Pick From Registry

If the WSDL is saved as a registry entry, select this option and choose the reference key of that registry entry from the Governance Registry or Configuration Registry. For more information, see [Working with the Registry](#).

Adding WSDL resources from the registry

If the WSDL has dependencies on items that you have saved in the registry, you map their location as specified in the WSDL to their key in the registry. For example, assume the WSDL imports the metadata schema as follows:

```xml
<xsd:import namespace=http://www.wso2.org/test/10 schemaLocation="metadata.xsd" />
```

In this case, the location is specified in the WSDL as `metadata.xsd`, so enter `metadata.xsd` in the Location field and then specify the registry key of the `metadata.xsd` file either by typing the key or clicking the appropriate registry and selecting it. After specifying the location and key, click **Add Resource**.

You can delete a WSDL Resource using the “Delete” button in the “Action” column.
Using publishWSDL option with auto generated jaxws web service WSDL

WSDLs auto generated via JAX-WS have an empty SOAPAction value. Therefore, message dispatching does not work when using such WSDLs. This issue has been solved in ESB 4.8.0 and later versions. You can do one of the following to overcome this in ESB 4.7.0 and ESB 4.6.0:

- Append the operation name to the endpoint URL when sending messages to the proxy service (e.g., http://serverName/app/serviceName/operationName).
- Modify the client level code to send the expected SOAPAction value.
- If the WSDL of the back end service has "" as the SOAPAction value, you should modify the back end service to have a value other than "" for each operation. Note that there is a violation of the service contract by the client if the WSDL of the back end service has a SOAPAction value different to "" when the client sends "" as the SOAPAction value. This should be fixed on client level.