HP Service Virtualization

For the Windows ® operating system

Software Version: 2.01

User Guide

Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notice

© Copyright 2011-2012 Hewlett-Packard Development Company, L.P.

Trademark Notices

Adobe™ is a trademark of Adobe Systems Incorporated.

Internet Explorer®, SQL Server®, Microsoft®, Windows®, Windows® XP and Windows 7® are U.S. registered trademarks of Microsoft Corporation.

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

IBM® and WebSphere® are trademarks or registered trademarks of International Business Machines Corporation, IBM, in the United States and in other countries.

TIBCO® is either the registered trademark or the trademark of TIBCO Software, Inc. and/or its subsidiaries in the United States and /or other countries.

Intel® is trademark of Intel Corporation in the U.S. and other countries.

Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.

- Document Release Date, which changes each time the document is updated.

- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition of a document, go to:

http://h20230.www2.hp.com/selfsolve/manuals

This site requires that you register for an HP Passport and sign in. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

Or click the **New users - please register** link on the HP Passport login page.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HP sales representative for details.

Support

Visit the HP Software Support Online web site at:

http://www.hp.com/go/hpsoftwaresupport

This web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support web site to:

- Search for knowledge documents of interest

- Submit and track support cases and enhancement requests

- Download software patches

- Manage support contracts

- Look up HP support contacts

- Review information about available services

- Enter into discussions with other software customers

- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

To find more information about access levels, go to:

http://h20230.www2.hp.com/new_access_levels.jsp

Contents

Chapter 1

Welcome to HP Service Virtualization

Welcome to Service Virtualization, HP's tool for simulating services during testing.

HP Service Virtualization software allows developers and testers access to limited or unavailable services in a simulated, virtual environment. This helps application teams lower costs and reduce testing times by finding defects earlier in the application life cycle when they are easier, faster, and less expensive to fix. It helps improve quality by enabling quality assurance (QA) teams to test what otherwise couldn't be tested. It also helps isolate problems that are based on dependencies between services in composite applications. This helps significantly reduce delays and manage the costs and complexity of composite application development and testing.

This chapter includes:

- "How This Guide Is Organized" below

- "Service Virtualization Documentation" on next page

- "What's New in Service Virtualization 2.0" on next page

How This Guide Is Organized

This guide contains the following chapters:

- **Chapter 1** "Welcome to HP Service Virtualization" above

  About this guide.

- **Chapter 2** "Getting Started" on page 11

  Describes how to install and configure Service Virtualization.

- **Chapter 3** "Virtual Services" on page 13

  Describes what Virtual Services are and how they are used.

- **Chapter 4** "Simulation Modeling" on page 32

  Describes how to use simulation models.

- **Chapter 5** "Composite Application Topology" on page 52

  Describes how to use the topology interface.

- **Chapter 6** "Supported Technologies And Environments" on page 61

  Describes how to install and work with agents.

- **Chapter 7** "Security" on page 74

  Describes how to define security settings.

- **Chapter 8** "HP Test Automation Tools Integration" on page 83

Service Test, Load Runner and Performance Center integration.

- **Chapter 9** "Service Virtualization Demos" on page 92

A series of functional demonstrations.

Service Virtualization Documentation

Service Virtualization includes the following online documentation:

HP Service Virtualization Online Help. Available from the Service Virtualization user interface by clicking in the window and pressing F1 or clicking the **Help** button.

Printer Friendly Documentation. Online books can be viewed and printed using Adobe Reader, which can be downloaded from the Adobe Web site. To access, click www.adobe.com.

- **HP Service Virtualization User Guide.** can be accessed from **HP Service Virtualization 2.01 > Documentation > User Guide** from the Start menu.

What's New in Service Virtualization 2.0

Enhanced support for service virtualization protocols

HP Service Virtualization 2.0 now supports a broad set of communication protocols. Developers, functional and performance test engineers can quickly create or provision virtual integration and test environments for Software as a Service (SaaS), Cloud and Mobile applications using **Representational State Transfer (REST) access protocol**.

In addition, HP Service Virtualization 2.0 now supports the virtualization of **IBM WebSphere MQ** systems to enable rapid delivery of new applications without the need to access high cost production business systems. Now, users are able to continuously test applications depending on mainframe systems without a need to wait for their availability.

HP Service Virtualization 2.0 also adds support for **Generic XML and Binary** protocol virtualization. It allows creation of virtual testing environments without a necessity of understanding specific message structures. Generic XML allows virtualization of XML oriented protocols with access to non-specific XML messages structure. Generic binary virtualization allows simulation of any other protocol with access to text/binary representation of messages.

Another Service Virtualization 2.0 enhancement is support for **protocols metadata**. Users can now very easily add, modify and manage protocol headers or other transport dependent information that must be used in creation of virtual service models.

Enhanced Virtual Services modeling

Ease of use and intuitive navigation is a key to any productive tool. HP Service Virtualization 2.0 introduces usability enhancements to Service Virtualization Designer. **Enhanced Data Model Editor** allows easier navigation and manipulation of learned, imported or entered data into virtual service models. It also introduces enhanced Virtual Service wizards with easy navigation and

filtering of out of the box and custom protocols. Data driven virtualization now support also REST and MQ protocols.

Another adition is new **Service Description Learning and Editing**. Service Virtualization 2.0 allows creation of Virtual Services without Service Description file (e.g. XSD schema) by selecting non-service-description option in virtual service creation wizard. It also introduces possibility to add schemas to already learned Virtual Services in generic virtualization or edit structure of undelying Service Description.

Message oriented communication is very often asynchronous in its nature. Responses may be in some cases delivered as a sequence of multiple messages. HP Service Virtualization 2.0 now adds support for **Multi-response message simulation**. Virtual services may now learn and simulate response messages split to multiple parts (e.g. large documents), simulate real-time process generating a series of responses (e.g. information stream) or transactions formed as responses from multiple subsystems (e.g. complex provisioning system).

Data driven testing is important for using pre-prepared test datasets, esp. when shared among serveral tools. Service Virtualizalition 2.0 now adds support for **Data driving arrays and choices**. This completes set of message type constructs for which external data can be used.

Using **Services Administration** there is now possible to manage services of any connected HP Service Virtualization Server, without necessity to open a project.

Functional and technical improvements

Service Virtualization becomes a critical piece for agile application development and testing. Demand for performance and scalability of service virtualization solutions grows as customers start leveraging its benefits out of project teams across business units and enterprises. HP Service Virtualization 2.0 introduces **15% improvements in performance** of Service Virtualization Server when comparing to previous version as well as enhanced support for centralized deployments.

Chapter 2

Getting Started

This chapter includes:

- "Start Page" below

- "Updating Projects" on next page

🖈Start Page

When you start the application the Start Page opens.
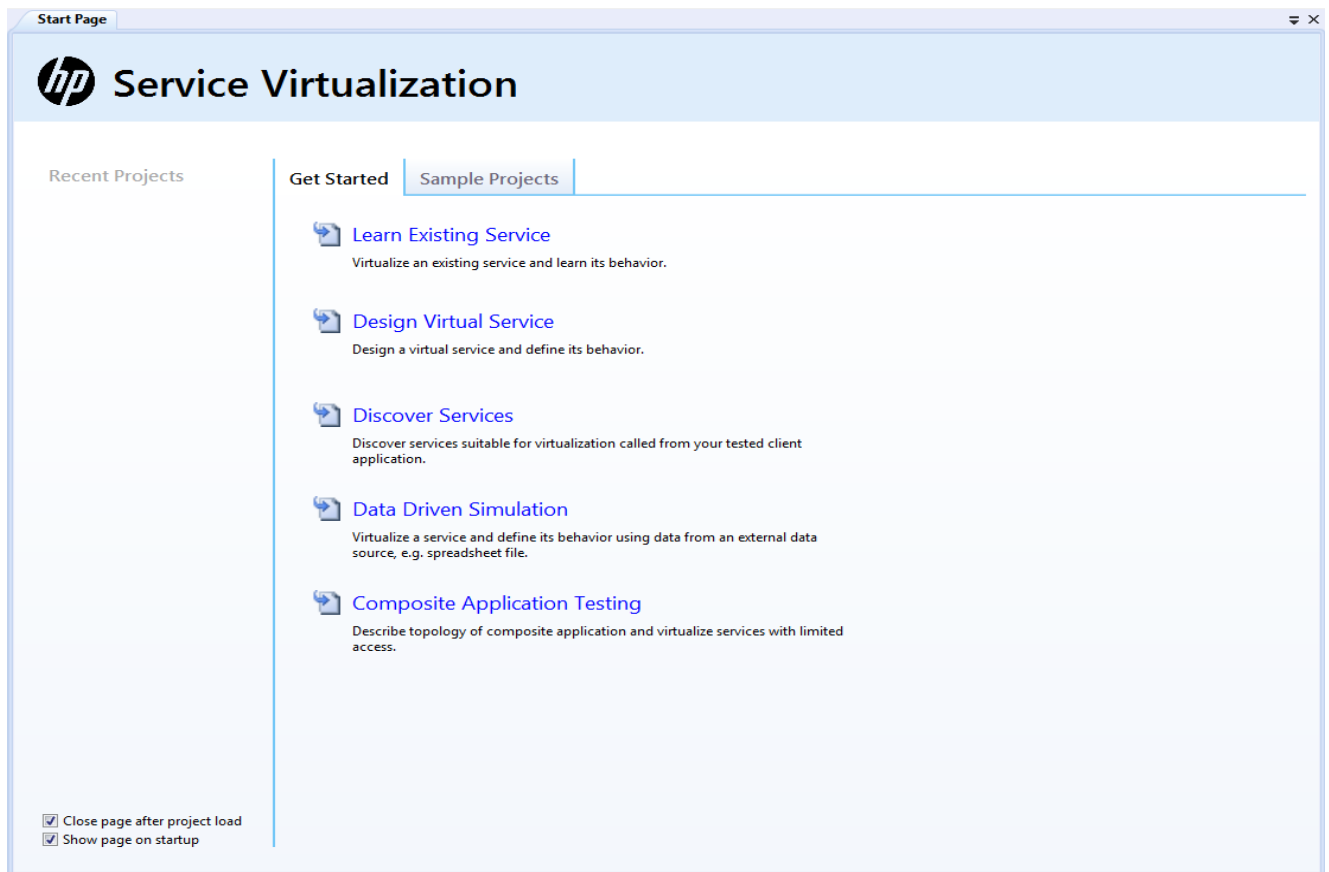
The Start Page includes links to the most recent projects, links to common procedures, and a tab for the installed sample projects.

See "How to Create a Virtual Service" on page 14

Select **Close page after project load** to close the Start Page as soon as you load a project.

Select **Show page on startup** to show the Start Page on startup.

Updating Projects

Projects created in earlier versions of HP Service Virtualization are fully compatible with Service Virtualization 2.01.

When opening an earlier project, a pop up window will inform you that the project contains resources in an older format and ask you if you want to update them.

Click **Yes** to update your existing project.

Chapter 3

Virtual Services

Service Virtualization places a Virtual Service between the service under test and the real service you want to simulate.

The Virtual Service uses the same service description as the real service. During Learning Sessions, the Virtual Service passes the communication between the service under test and the real service and records the service behavior and performance to the Simulation Model.

During Simulation Sessions, the Virtual Service does not pass the communication to the real service, and instead returns responses and performance defined by the Simulation Model.

This Chapter includes:

Dealing with locked services.

Virtualization Explorer

The Virtualization Explorer is initially the left hand pane of the application.

Open projects are presented in a tree format displaying the logical structure of the virtualization project.

Click on an object in the list to display the details of the object in the lower pane.

Double-click an object in either pane to view a description of that object.

Right clicking on objects brings up the context menu allowing different options depending on the object.

Virtual Service Design

The service virtualization process means creating virtual service endpoints. When you create an endpoint you can reconfigure your client application to use this endpoint instead of the real service endpoint. The purpose of the virtual service is to mediate between the client and the real service. Service Virtualization allows you to manipulate virtual services to get different results.

A virtual service has three basic modes- Learning, Simulating, and Standby:

- **Learning mode**: A virtual service can work as a proxy to record and learn the behavior of a real service – it forwards real communication between a client and a service.

- **Simulating mode**: A virtual service is typically switched to a Simulating mode. In this mode the virtual service answers a client according to learned behavior and the real service does not receive any communication.

- **Standby mode**: Real service responds to client requests.

How to Create a Virtual Service

The most important step in simulating a service is the creation of a Virtual Service.

To Create a Virtual Service:

1. If no project exists yet, click **Learn Existing Service** or **Design Virtual Service** in the Get Started tab of the Start Page. If a project does exist, open the project context menu in the Virtualization Explorer and select **Add>Virtual Service** to open the Import Real Service Description dialog box.

   Note: If no project exists, it is created with the creation of a new virtual service.

   a. If you have a real service description available to import such as a SOAP service (`.wsdl`) or an XML schema (`.xsd`), enter the file name, or click **Browse** to go to the file location and click **Start**.

      i. If more than one Real Service Endpoint is detected in a SOAP service, you will have to select which endpoints you require. If using an XML schema, select a transport protocol and provide destinations.

      ii. Click **Next**.

      iii. Your selections are presented in the Summary Of Virtualization. Click **Change** to alter your selections or **Virtualize** if your selections are correct.

      See "How to Import Service Descriptions" below.

  b. If you do not have a real service for import, select **I don't have a service description** and click **Start** to open the Choose Service Protocol dialog box.

      i. The available protocols are presented and may be filtered by protocol type and transport type. Click on the required protocol and fill in the details required for selected protocol. See "How to Create REST Services" on page 19 and "How to Create XML or Binary Services" on page 19.

      ii. Click **Next** .

      iii. Your selections are presented in the Summary Of Virtualization. Click **Change** to alter your selections or **Virtualize** if your selections are correct.

Service Virtualization creates a new virtual service, a data model, and a performance model as part of the virtualization project, and opens the "Virtual Service Editor" on page 23 view.

How to Import Service Descriptions

In order to virtualize a service, Service Virtualization needs to know what it does and what endpoints it uses. The most common way to obtain this information is to import service description documents.

Service Virtualization supports the following service description document types:

- **WSDL**

  Web-Service Definition Language documents are the most common way to describe SOAP services. They often contain references to other WSDLs and XSDs that must be available in the referenced locations in order to import them and correctly describe the services they define.

  For details, see "How to Import SOAP Service Descriptions" below.

- **XSD**

  XML Schema documents may also describe XML services. They often contain references to additional XSDs that must be available in the referenced locations in order to import them and correctly describe the services they define.

  For details, see "How to Import XML Service Descriptions" on page 17.

How to Import SOAP Service Descriptions

The most common way to describe a SOAP Service is with a WSDL document. Service Virtualization provides an import wizard that analyzes the content of the WSDL and enables you to

associate it with a particular virtualized service.

To Import SOAP Service Descriptions:

1. Access the Import Real Service Description wizard in one of the following ways:

   ▪ During the creation of a Virtual Service. For details, see "How to Create a Virtual Service" on page 14.

   ▪ In the Topology Editor view, virtualize a service that does not contain a service description. For details, see "How to Create Virtual Services in Topologies" on page 57.

   ▪ In the Topology Tasks view, virtualize services that do not contain a service description. For details, see "How to Virtualize Services in the Tasks View" on page 54.

   ▪ Select **New** and *Virtual Service* from the **File** menu

2. To select the WSDL specifying the SOAP Service, do one of the following:

   ▪ Input the path to the WSDL or use the browse button to find the location.

   ▪ Input the URL where the WSDL is exposed.

3. Click **Start** to process the selected WSDL.

   ▪ If the WSDL describes multiple services or a single service specifying multiple ports, select one from the list of available ports and click **Next** to proceed to the Summary of Virtualization.

   ▪ If the WSDL only describes one service with a single port, the Summary Of Virtualization is presented.

   ▪ If the description is for SOAP over JMS, a dialog will open allowing the change of the destination name and the possibility of entering a reply to address.

4. **Optional**: Configure real and endpoint for virtual service if different from what is presented in the Summary of Virtualization:

   a. For **SOAP over HTTP(s)**:

      i. Click **Change** under the Virtual Service description to open the **Change Virtual Service** dialog.

         The Virtual Service name, Agent type and Path can be altered. Click **Save** to save changes or **Discard** to go back to default values.

      ii. Click **Change** under the Real Service description to open the **Change Real Service** dialog.

         The Real Service Endpoint can be altered. Click **Save** to save changes or **Discard** to go back to default values.

         Note: If service is secured, you must manage the credential store. For details, see "How to Set Authentication Credentials" on page 22.

      iii. When all values are correct, click **Virtualize** to create the service.

   b. For **SOAP over JMS**:

Click **Change** under the Virtual Service description to open the **Change Virtual Service** dialog.

    i.   Keep the default or rename the Virtual Service.

    ii.   Select different agent from the drop down list if required.

    iii.   Specify **Destination Name** (JNDI name) for virtual **Destination** endpoint.

    iv.   (Optional) If you are using a permanent **Reply To** destination, fill its JNDI name. Otherwise, leave the field blank and a temporary **Reply To** destination is used.

    v.   Click **Save** to save changes or **Discard** to go back to default values.

Click **Change** under the Real Service description to open the **Change Real Service** dialog.

    i.   Check if the **Destination Name**, **Reply To** (blank if temporary **Reply To** is used) and **Connection Factory** in the Change Real Service dialog have been pre-filled with correct JNDI names. If necessary, change these values according to your JMS configuration.

    ii.   Click **Save** to save changes or **Discard** to go back to default values.

  c. For **SOAP over TIBCO Enterprise Message Service™** (hereafter TIBCO EMS):

Click **Change** under the Virtual Service description to open the **Change Virtual Service** dialog.

    i.   Keep the default or rename the Virtual Service.

    ii.   Select appropriate agent from the drop down list.

Click **Change** under the Real Service description to open the **Change Real Service** dialog.

    i.   Check if **Destination Name** has been pre-filled correctly and change the values according to your EMS configuration.

    ii.   Select a **Destination Type** from the drop down list.

    iii.   Click **Save** to save changes or **Discard** to go back to default values.

5. Review the settings for the virtual service and click **Virtualize** to virtualize the selected service, add the virtual service to the virtualization project, and put the virtual service in Standby Mode.

How to Import XML Service Descriptions

To Import XML Service Descriptions:

1. Access Import Real Service Description in one of the following ways:

  ■ During the creation of a virtual XML Service. For details, see "How to Create a Virtual Service" on page 14.

  ■ In the Topology Editor view, virtualize a service that does not contain a service description. For details, see "How to Create Virtual Services in Topologies" on page 57.

- In the Topology Tasks view, virtualize services that do not contain a service description. For details, see "How to Virtualize Services in the Tasks View" on page 54.

- Select **Add** and *Virtual Service* from the File **menu**.

2. Enter a path, a URL, or use the browse button to an existing XML schema (`.xsd`).

3. Click **Start**.

4. Select the required protocol and click **Next**.

   Note: Protocols can be filtered by either entering text in the search field or using the provided Protocol and Transport filters.

5. Select an input and output message type from the drop down lists and click **Next**.

6. Fill in the properties for the specific service:

   XML over generic JMS:

   Virtual Service

   Destination name: JNDI name of destination where virtual service will expect requests.

   Reply to: JNDI name of destination where virtual service will send responses. If the client provides a ReplyTo JMS property, this field can be left empty.

   Real service

   Destination name: JNDI name of destination where real service expects requests.

   Reply to: JNDI name of destination where real service sends responses. If this field is left empty, the application will create a temporary destination for receiving response from the real service and will set the ReplyTo JMS property in request to point to it.

   Connection factory: JNDI name of connection factory to use.

   XML over TIBCO EMS

   Destination name: Name of destination where requests are sent.

   Destination type: Type of destination where requests are sent.

   Since Service Virtualization records messages on TIBCO EMS non-intrusively, all parameters in the configuration are only related to the real service. At the moment when the virtual service mode is switched to emulation mode, the real service is automatically disconnected from TIBCO EMS and it is replaced by Service Virtualization.

   There is no response destination name since the response destination is always read from request properties.

   XML over IBM® WebSphere® MQ

   Virtual Service

   Destination name: Name of queue where the virtual service should expect requests.

   Reply to: Name of queue where virtual service will send responses. If the client provides a ReplyToQueue message property, this field can be left empty.

   Real service

Destination name: Name of queue where real service expects requests.

Reply to: Name of queue where real service sends responses. If this field is left empty, The application will create a temporary queue for receiving responses from the real service and set the ReplyToQueue message property in the request to point to it. Note that WebSphere MQ must be configured so that the application has permission to create temporary queues.

7. Click **Next**.

8. Your selections are presented in the Summary Of Virtualization. Click **Change** to alter your selections or **Virtualize** if your selections are correct.

Note: If using and HTTP or HTTPS Gateway agent, the virtual service may have only one specified endpoint.

How to Create REST Services

1. Access Import Real Service Description in one of the following ways:

   ■ During the creation of a Virtual Service. For details, see "How to Create a Virtual Service" on page 14.

   ■ In the Topology Editor view, virtualize a service that does not contain a service description. For details, see "How to Create Virtual Services in Topologies" on page 57.

   ■ In the Topology Tasks view, virtualize services that do not contain a service description. For details, see "How to Virtualize Services in the Tasks View" on page 54.

2. Select **I don't have a service description**.

3. Click **Start**.

4. Select the REST Service Protocol and click **Next**.

5. Enter the endpoint. If multiple endpoints are required, separated them with a space, comma, semicolon or a new line and ensure that they are all for the same service.

6. Click **Next**.

7. Your selections are presented in the Summary Of Virtualization. Click **Change** to alter your selections or **Virtualize** if your selections are correct.

How to Create XML or Binary Services

If you have an existing XML or binary service, it can be added to the simulation and if not, it can be created from scratch. If the service is of an unknown type, creating a binary service is the best solution as if the application cannot understand the format, it can always record it in binary format although Service Virtualization will not recognize the structure.

1. Access Import Real Service Description in one of the following ways:

   ■ During the creation of a Virtual Service. For details, see "How to Create a Virtual Service" on page 14.

   ■ In the Topology Editor view, virtualize a service that does not contain a service description.

For details, see "How to Create Virtual Services in Topologies" on page 57.

- In the Topology Tasks view, virtualize services that do not contain a service description. For details, see "How to Virtualize Services in the Tasks View" on page 54.

2. Select **I don't have a service description**.

3. Click **Start**.

4. Select a binary messages type and click **Next**.

5. Fill in the properties for the specific service.

   XML/Binary over HTTP

   Endpoint: Supply Endpoints separated by space, comma, semicolon or a new line.

   XML/Binary over Generic JMS:

   Virtual Service

   Destination name: JNDI name of destination where virtual service will expect requests.

   Reply to: JNDI name of destination where virtual service will send responses. If the client provides a ReplyTo JMS property, this field can be left empty.

   Real service

   Destination name: JNDI name of destination where real service expects requests.

   Reply to: JNDI name of destination where real service sends responses. If this field is left empty, the application will create a temporary destination for receiving response from the real service and will set the ReplyTo JMS property in request to point to it.

   Connection factory: JNDI name of connection factory to use.

   XML/Binary over TIBCO EMS

   Destination name: Name of destination where requests are sent.

   Destination type: Type of destination where requests are sent.

   a. Fill in the properties for the specific service:

      XML over generic JMS:

      Virtual Service

      Destination name: JNDI name of destination where virtual service will expect requests.

      Reply to: JNDI name of destination where virtual service will send responses. If the client provides a ReplyTo JMS property, this field can be left empty.

      Real service

      Destination name: JNDI name of destination where real service expects requests.

      Reply to: JNDI name of destination where real service sends responses. If this field is left empty, the application will create a temporary destination for receiving response from the real service and will set the ReplyTo JMS property in request to point to it.

      Connection factory: JNDI name of connection factory to use.

XML over TIBCO EMS

Destination name: Name of destination where requests are sent.

Destination type: Type of destination where requests are sent.

Since Service Virtualization records messages on TIBCO EMS non-intrusively, all parameters in the configuration are only related to the real service. At the moment when the virtual service mode is switched to emulation mode, the real service is automatically disconnected from TIBCO EMS and it is replaced by Service Virtualization.

There is no response destination name since the response destination is always read from request properties.

XML over IBM® WebSphere® MQ

Virtual Service

Destination name: Name of queue where the virtual service should expect requests.

Reply to: Name of queue where virtual service will send responses. If the client provides a ReplyToQueue message property, this field can be left empty.

Real service

Destination name: Name of queue where real service expects requests.

Reply to: Name of queue where real service sends responses. If this field is left empty, The application will create a temporary queue for receiving responses from the real service and set the ReplyToQueue message property in the request to point to it. Note that WebSphere MQ must be configured so that the application has permission to create temporary queues.

b. Click **Next**.

c. Your selections are presented in the Summary Of Virtualization. Click **Change** to alter your selections or **Virtualize** if your selections are correct.

XML/Binary over WebSphere MQ

Virtual Service

Destination name: Name of queue where the virtual service should expect requests.

Reply to: Name of queue where virtual service will send responses. If the client provides a ReplyToQueue message property, this field can be left empty.

Real service

Destination name: Name of queue where real service expects requests.

Reply to: Name of queue where real service sends responses. If this field is left empty, The application will create a temporary queue for receiving responses from the real service and set the ReplyToQueue message property in the request to point to it. Note that WebSphere MQ must be configured so that the application has permission to create temporary queues.

6. Click **Next**.

7. Your selections are presented in the Summary Of Virtualization. Click **Change** to alter your selections or **Virtualize** if your selections are correct.

> **Note**: If using and HTTP or HTTPS Gateway agent, the virtual service may have only one specified endpoint.

Services Administration

As projects are ongoing, it may be required that virtual services being provided by different servers be made unavailable due to personnel or business changes. These can be managed in the Services Administration module.

To access Service Administration, click the **Service Administration** link on the Getting Started tab of the Start Page.

All virtual services in the project are presented with their status, associated models and server location and the view can be filtered by server location.

To manage the servers, from the **Tools** menu, select **Options** and the **Servers** tab or use the quick link from the **More Actions** button at the bottom of the window and select **Manage Servers**.

To add a server, click **Add**, provide the address of the server and click **Ok**.

To remove a server, click a server to highlight it and click **Delete** and then **Yes** in the confirmation dialog.

How to Change Servers

To move a project from one server to another server:

1. Right click the object in the Virtualization Explorer and select **Change Server** or from the Project menu, select **Change Server**.

2. Select an existing server or enter the URL for a server and click **Next**.

3. Select agent from the drop down lists and click **Finish**.

z

How to Set Authentication Credentials

Some services may require client authentication on either the transport or message level. When virtualizing these services the application needs to know the client credentials used to connect to the real service. The only exception is a scenario wherein a real service with HTTP transport authentication (Basic, Digest, NTLM) is virtualized through the HTTP(S) proxy agent. In this scenario, authentication requests are forwarded and the service virtualization doesn't require the credentials in the service's credential store.

Prerequisites:

User of the Virtual Service must exist in the Local User Store of a Windows installation.

To Set Service Security Settings:

1. Access Security Settings in one of the following ways:

   ▪ From the Virtualization Explorer view, open the Editor view for a virtual service and expand the **Security Settings** section.

   ▪ In the Topology Editor view, virtualize a secured service that does not have any authentication set. For details, see "How to Create Virtual Services in Topologies" on page 57.

   ▪ In the Topology Tasks view, virtualize secured services that do not have any authentication set. For details, see "How to Virtualize Services in the Tasks View" on page 54.

2. To modify the authentication credentials for a secured service, click **Edit Credential Store** and do any of the following:

   ▪ To add user credentials:

     i. Click **Add Identity** to open the Edit Identity Details dialog box.

     ii. Input a Username and Password.

     iii. *Optional:* Select **Show Password** to display the password in Security Settings.

     iv. *Optional:* Add Certificate

     v. Click **OK** to add the credentials to the Security Settings dialog box.

   ▪ To edit user credentials:

     i. Select the user to edit, and click **Edit Identity** to open the Edit Identity dialog box.

     ii. Input a Username and Password.

     iii. *Optional:* Select **Show Password** to display the password in Security Settings.

     iv. Click **OK** to change the credentials in the Security Settings dialog box.

   ▪ To remove user credentials:

     ○ Select the user to remove, click **Remove Identity**, and confirm your decision.

   ▪ To import credentials:

     i. Click Import to open the Import Identities dialog.

     ii. Select virtual service which contains identity you want to import.

     iii. Select identity you want to import.

     iv. Click OK to import the identity from selected virtual service.
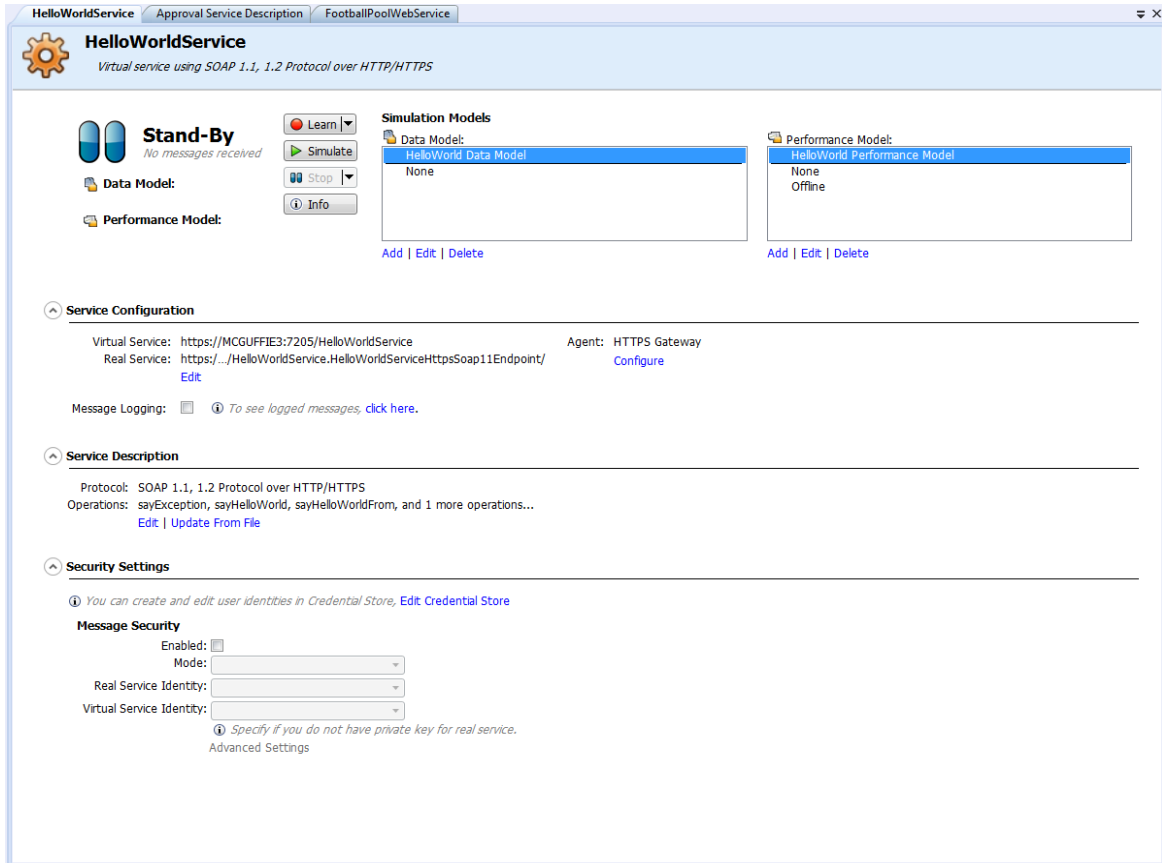
   ▪ Save the Virtual Service.

Virtual Service Editor

The Virtual Service Editor enables you to control the mode of the virtual service and the models currently in use, configure the endpoints, and configure any security settings.

Access the Virtual Service Editor in one of the following ways:
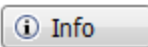
- In the Virtualization Explorer, double-click the Virtual Service you want to view or edit.

- In the Runtime View, click the name of the Virtual Service you want to view or edit.

- In the Topology Editor, open the context menu for the Virtual Service you want to view or edit and select **Open**.
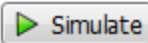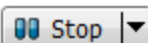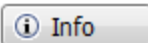
Virtual Service Editor

The Virtual Service Editor displays the following content and controls:

Virtual Service Editor Content

Section	Description
Virtual Service Name and Description	The name and description of the Virtual Service (click to Edit).
Current Mode and Models	The large icon displays whether the service is in Learning, Simulation, Standy Mode or if the service is Offline. Below the icon, the Data and Performance Models currently in use are displayed.
Virtual Service Controls	<table><tr><th>Control</th><th>Description</th></tr><tr><td>Info</td><td>Open the Runtime Report to display current information about the service.</td></tr></table>

Section	Description

Control	**Description**	
▶ Simulate	Finish the current learning session and add any data learned in the session to the Simulation Model.	
▌▌ Stop ▼	Put the service into Standby Mode with the option of maintaining or disposing of learned data.	
ⓘ Info	Open the Runtime Report to display current information about the service.	

Section	Description
Simulation Models	The Simulation Models section enables you to select which Data and Performance models are in use and manage the models associated with the Virtual Service.
Enable Message Logging	The checkbox toggles logging. The messages are stored on disk in the Designer log directory `%TEMP%\HPServiceVirtualizationLogs`. Messages processed by the embedded server are stored in the sub-directory `msg-embedded\[Virtual Service Name]`. Messages processed by the standalone server are stored in the sub-directory `msg-standalone\[Virtual Service Name]` on the standalone server itself. Each message is stored in a single file named `[Message Order Number]-[Message Id]`.
Service Configuration	Displays the endpoints for the Virtual Service with options to copy these details to the clipboard. Click **Edit** to change the endpoint details.
Agent	Displays the agent selected for the service. If using the embedded server, click **Configure** to change the agent settings. For details, see "Supported Technologies And Environments" on page 61. If using a standalone server, click configure to see documentation regarding agents.
Service Description	Displays service descriptions and any metadata associated with the service. Click **Edit** to add new operations or update metadata. Click **Update From File** to update the service description.
Security	Depending on protocol, displays any credentials used to access the service with management options. Click **Edit Credential Store** to modify credentials. For details, see "How to Set Authentication Credentials" on page 22. For Message Security, select Enabled to apply settings. For details, see "How to Set Message Security" on page 76.

⚙ Managing Virtual Services

The Virtual Services in the virtualization project can be managed in several ways:

- Switching between the simulation and standby modes, passing the control to the real service without reconfiguring clients.

- Activating learning mode where the real service responds and the selected data and/or performance models of virtual service get updated.

- Managing all services at once in the runtime view and changing the runtime where the virtual services run (i.e. from an embedded to a standalone server).

"Learning Service Behavior" below

"Simulating Services" below

Learning Service Behavior

Real services may not be available on a permanent basis due to cost, access, or if different model is required.

After a Virtual Service is created, the behavior of the real service must be recorded in order to see the requests and responses of the real services.

Once this behavior is captured by Service Virtualization, it can be used to create data and performance models.

For details, see "How to Learn Services " below.

How to Learn Services

Prerequisites:

- Create a virtual service. For details, see "How to Create a Virtual Service" on page 14.

- *Optional:* Select the Data and Performance Models to record to. For details, see "How to Select Models" on next page.

1. After a virtual service is created, it is displayed in Standby mode by default. Clicking on **Learn** stops or starts the learning mode. Click the arrow next to the Learn button to select the data model, the performance model, or both.

2. Run the application communicating with the real service. All service calls are recorded.

3. When you have enough recorded messages, click the **Stop** button to stop the learning process. The simulation model updates with the recorded data.

Simulating Services

The virtual service can use both the data and performance models for the simulation or use them individually.

Data Simulation Options

It is possible to simulate learned responses by selecting one of the data models. It is also possible to turn the data simulation off and let the real service respond, and thereby simulating only the performance using one of the performance models.

See "Data Model" on page 33

Performance Simulation Options

It is possible to simulate learned or customized performance by selecting one of the performance models.

See "How to Manage Models" on page 33

It is possible to turn the performance simulation off by selecting the **None** performance model. The response times are not affected by any model and the virtual service responds as fast as possible. It is also possible to simulate the absence of the service by selecting the **Offline** performance model. The virtual service does not respond back to the client in this mode.

"How to Select Models" below

"How to Start Simulations" below

How to Select Models

Before you start a learning or simulation session, you can select which Data and Performance Models you want to associate with each Virtual Service.

Prerequisites:

- Create a virtual service. For details, see "How to Create a Virtual Service" on page 14.

- *Optional but recommended:* Record the performance of the real service. For details, see "Learning Service Behavior" on previous page.

- *Optional:* Edit the Data and Performance Models. For details, see "Simulation Modeling" on page 32.

To Select a Data or Performance Model:

1. In the Virtualization Explorer, click the virtual service name to open the Virtual Service Editor.

2. The Simulation Model section shows the available models for the Virtual Service. Select your models.

Models can be added, edited, or deleted from the Virtual Service using the associated buttons. For details, see "How to Manage Models" on page 33

How to Start Simulations

Prerequisites:

- A simulation model must already exist. For details, see "How to Learn Services " on previous page

1. From the virtual service page, click the **Simulate** button to start simulation of a real service.

2. Run the application communicating with the original real service.

Runtime View

The Runtime View provides an overview of all the Virtual Services in your Virtualization Project and the functionality to control them. During a learning or simulation session, the Runtime View provides an overview of the communication through the Virtual Services.

Access the Runtime View using the **View>Runtime View** menu option.

Runtime View						▼ ⇧ ✕
☐ All ▼ **Virtual Service**	Status	Messages (Unique)	Data Model	Performance Model	Project	View All Services ▼
☐ ⏸ **Member Accounts**	2 Problem(s)	8 (8)	Real Service	Maximal Performance	Simulation	

⏺ Learn ▼ | ▶ Simulate | ⏸ Stop | More Actions ▼
▣ Runtime View | ⬛ Problem List
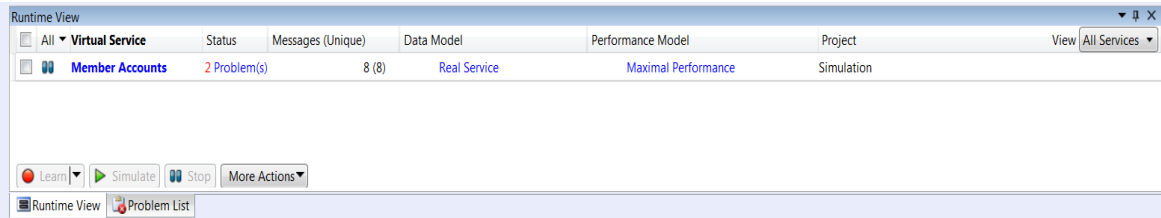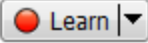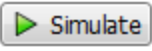
The Runtime View displays a table of all the Virtual Services in your project with the following options and information for each service:

Runtime Display

Column	Description
Selection Boxes	Select services and apply the controls described in "Virtual Service Controls".
Virtual Service Mode	Displays whether the services are in Learning or Simulation Mode with an option to filter the table.
Virtual Services	Click the service name to open its editor.
Problems	Indicates any problems which occurred during the learning or simulation.
Messages / Unique	Indicates the number of messages and unique messages passed through the Virtual Service during the current learning or simulation session.
Data Model (Accuracy)	Indicates the Data Model currently in use and its accuracy during simulation.
Performance Model (Accuracy)	Indicates the Performance Model currently in use and its accuracy during simulation.
Project	Displays the name of the Project to which the Virtual Service belongs.

The Runtime View contain the following controls:

Control	Description
⏺ Learn ▼	Put the service into Learning Mode. Any communication through the Virtual Service is added to the Simulation Model in this mode. Use the drop down arrow to select which models to update: • Data & Performance (default) • Data Model

Control	Description
	• Performance Model
▶ Simulate	Finish the current learning session and add any data learned in the session to the Simulation Model.
❚❚ Stop ▼	Put the service into Standby Mode with the option of maintaining or disposing of learned data.
Info	Located in the **More Actions** menu. Open the Runtime Report to display current information about the service.
Unlock	Located in the **More Actions** menu. Used to unlock a blocked service. See "Service Locking" on next page.

Problem List

The Problem list displays problems that occurred during the application or server run. The source of problems can be either runtime errors in the application or a problem occurring during a service lifecycle ie: deployment/ standby/ learning/ simulating.

	Problems ▼	Type	⚙ Fix It	Member Accounts ▼	Description	Time
☐	⚠	**3x** Service Configuration	Fix User Store	Member Accounts	The credential store does not contain user hpguest.	9/20/2011 5:02:2
☐	⚠			Member Accounts	User tried to connect to service but has not supplied name.	9/20/2011 5:02:2

Copy   Delete

Runtime View   Problem List
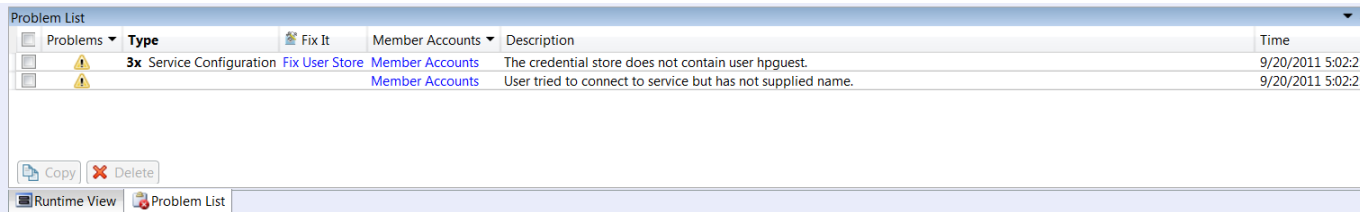
Column	Description
Problems	Filters different problem types. Error/ Warning/ Information
Type	Problem Category and number of occurrences.
Fix It	Some problems can be resolved with user interaction. These contain a Fix It link to the part of the UI that may be source of the problem.
Source Filter	Filters problems for a specific service call.
Description	A description of the problem.
Time	Time and date of problem occurrence.

Updating Virtual Services

After a virtual service is created you may need to update the service description.

To update the service description:

1. Right click the service in the Virtualization Editor and select **Update Service Description**. If the service is not locked (see "Service Locking" below), the wizard starts.

2. Enter the path or URL, or use the Browse button to enter the new WSDL. Click **Next**.

3. The designer now examines the new description and displays the supported and unsupported transformations. If there is an unsupported transformation, the update stops and the WSDL must be altered. If all transformations are supported, click **Finish** to apply the update.

Note: In some cases, the designer may prompt the reloading of open editors.

Service Discovery

Service Discovery can be used to find all services used by an application via a proxy gateway. It can only be used in the case of an embedded server.

To use Service Discovery:

1. Either create a new **Service Discovery Project** through the **New Virtualization Project** Wizard or right click in the Topology view of an existing project.

2. Select the server and proxies.

   Note: The client application must be using the Service Virtualization proxy before attempting to find services.

3. Click **Run Service Discovery**.

4. When Services are discovered, click **Stop Discovery** to continue with the project. Discovered services are presented in a grouping called Discovered Services and can be used in the topology. For details, see "How to Model Composite Applications" on page 52.

Service Templates

Service Templates can be created in order to:

- Reuse Virtual Services in multiple testing environments.
- Reuse configuration, custom functions, data and views.

Note: Service Templates can be created by Professional services with complex functionality.

To save a service as a template:

1. Right click the service in the Virtualization Explorer and select **Save As Template**.

2. Give the template a name and specify the location.

3. Click **Save**.

Service Locking

A Designer or a test emanating from HP LoadRunner / HP Service Test may require indicating that they own a virtual service (start session) or that virtual service is free (stop session) in order to prevent conflicts. They may also need to know who is the present owner of a service as a service

can be only owned by one client at a time. A service may be locked by the owner and other clients can see who the owner is as each client has a unique "client ID".

When a service is locked then its configuration and all its Data and Performance models are also locked. The owner of the service can modify the service and its model, but other clients are not allowed. If a user tries to modify the service or its model then the UI displays that the service is locked and which client is the owner of the service. If a Designer or a test is the owner then modification is allowed.

If a service is locked, the following message is displayed :
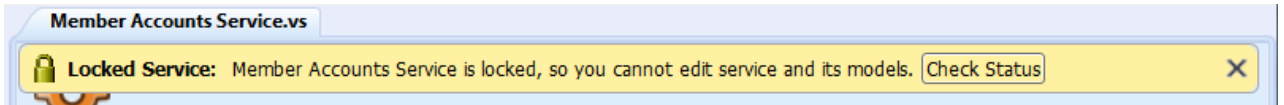
If a technical problem occurs or a test runs too long, you can force an unlock in Designer. The action **Unlock** can be found in the **More Actions** menu in the **Runtime View**.

Note: No changes are allowed to a service and its models during the learning process. This process must be completed regardless of the owner of the virtual service. **Unlock** is not available during this time.

Chapter 4

⚛Simulation Modeling

The most important part of Service Virtualization are the Simulation Models. Simulation Models are one of the following model types:

- **Data Model**

  The **Data Model** enables you to record actual requests and responses for the real service and then use this data for simulation using the Virtual Service. You can edit the data model from scratch or add to recorded data, add service calls, and model *stateful* behavior. This enables you to model the interaction between the service under test and the simulated service to meet many integration test cases. For details, see "Data Model" on next page.

- **Performance Model**

  The Performance Model enables you to record the performance for the real service and then use this as a model for the Virtual Service. You can customize the performance criteria of the model to meet many performance use cases. For details, see "Performance Model" on page 45.

When you create a Virtual Service, Service Virtualization creates a Data Model and Performance Model associated with it. These models serve as the default models for Learning and Simulation Sessions. For details, see "How to Create a Virtual Service" on page 14.

You can associate each Virtual Service with multiple customizable Data and Performance Models. For details, see "How to Manage Models" on next page

Prior to a Learning or Simulation Session, you can select which Data and Performance Models to use, including additional non-customizable options. For details, see "How to Select Models" on page 27.

Each model is customizable, enabling you to perform simulations that meet specific test use cases. For details, see "How to Edit Data Models" on page 41" and "How to Edit Performance Models" on page 48.

Also see:

"Data Model Editor" on page 35

A description of the Data Model Editor.

"Performance Model Editor" on page 46

A description of the Performance Model Editor.

"Service Description Editor" on page 49

A description of the Service Description Editor.

🎺How to Manage Models

By default, each Virtual Service is associated with one data model and one performance model, which are created with the Virtual Service. Additional models can be associated with a Virtual Service.

Prerequisites:

- Create a virtual service. For details, see "How to Create a Virtual Service" on page 14.

To Manage Models:

1. From the Virtualization Explorer, open the Editor view for a Virtual Service.

   The Virtual Service Editor displays a Simulation Model section showing the Data and Performance Models associated with the Virtual Service. Each type of model has **Add**, **Edit**, and **Delete** actions associated with it.

   Note: **None** and **Offline** are not actual models. They are options available for simulation purposes. **None** for Data Model messages are passed to the real service and its responses are sent back while still simulating the performance according to the selected Performance Model. **None** for Performance Model makes the Virtual Service respond as fast as possible. **Offline** Performance Model simulates the unavailability of the service.

2. For either Data Models or Performance Models, do any of the following:

   - **To Add a New Model:**

     i. Click the relevant **Add** link.

     ii. Input a name for the new model, and optionally for Performance Models, select to copy the performance metrics of the currently active model.

     iii. Click **OK** to add the new model to the relevant set of models.

   - **To Edit a Model:**

     ○ Select the model you want to edit, and click the relevant **Edit** link to open the Data Model Editor or Performance Model Editor view.

       For details, see "Data Model Editor" on page 35 and "Performance Model Editor" on page 46.

   - **To Delete a Model:**

     ○ Select the model you want to delete, click the relevant **Delete** link, and confirm your decision.

3. Save the Virtual Service.

🧊Data Model

The Data Model enables you to customize the requests and responses, and service call activity of a service during simulation.

When you create a Virtual Service, Service Virtualization creates a Data Model associated with it. For details, see "How to Create a Virtual Service" on page 14.

Each Virtual Service can have multiple Data Models. For details, see "How to Manage Models" on previous page.

Prior to recording or simulation you can select which model to use, including selecting to use the real service. For details, see "How to Select Models" on page 27.

This model is then available to learn the data behavior of the real service and can be customized to set specific data rules for its individual operations.

Data Rules

The main part of the Data Model consists of a set of Data Rules for each operation in the service. The following types of rules are available:

- **Learned Data Rule**

  The Learned Rule displays the requests and responses from Learning Sessions. In general, you do not customize this data but you may want to set conditions to ignore parts of the requests and responses and add service call activity.

- **Default Response**

  The Default Response provides a single custom response to apply in cases where there is no other data, or where you want to ignore specific parts of recorded response data.

- **Custom Rules**

  Custom Rules enable you to set custom responses and service call activity to specific requests enabling you to perform various testing use cases.

- **External Rules**

  External Rules are used to bind request and response data from external data source that can be used by several applications or exported from external applications like HP Service Test, HP LoadRunner or HP QuickTest. The data source can be edited by an external application and then the data can be refreshed in the Data Model.

You can set the priority order of multiple rules to meet various simulation testing use cases. Generally, the following order applies:

1. Custom or External Rules to provide specific responses and service call activity for the purpose of testing specific service behavior.

2. Learned Data Rule to provide typical responses and service call activity of the real service.

3. Custom Rules to provide responses and service call activity for requests that cannot be recorded or have not been recorded yet.

4. Default Response to provide a single generic response or generic parts of response data where other rules do not apply.

Service Call Activity

In many cases, the simulated service can call another service to perform some particular operation or to receive some additional data. Virtual Services can simulate this behavior by adding Service Call Activity to an operation. You can define static request data for the Service Call Activity for any row in the rule or use the Copy From function to copy data from the Virtual Service request or from

the response of another Service Call Activity. If a called service also has a response, you can use the Copy From function to copy some response data from a service call activity to a Virtual Service response. For more details, see "How to Add Service Call Activity" on page 43.

Tracks

The other main feature of the Data Model are Tracks that determine the order of the simulated service behavior.

In many test cases, the order of requests is important because a service may return different responses for the same request depending on the current state of the service. Service Virtualization enables you to simulate this *stateful* behavior using Tracks. Tracks enable you to construct sequences of requests and responses in the Data Model for the service. During a simulation session, Service Virtualization moves along the Tracks according to test requests that match the requests in the track and returns the appropriate response. For example, if the simulated service can return an approve or deny response which is determined by a particular state of the service, you can determine which response to return by specifying the sequence of requests and responses in the track. For more details, see "How to Edit Data Models" on page 41

Import Messages

New rows can be added to a rule by learning new data, by adding a new row and manually editing its cells or by importing messages.

Importing messages is useful in the case when it is not possible or it is difficult to learn communication between a tested application and a simulated service directly, but it is possible to listen to the communication and log transported messages via another tool. Importing Messages is available in the Data Model Editor from the context menu of the rules data table (only for Custom and Learned Rules). It is possible to import a request and/or response part of the message in the same format as it is sent via communication protocol from a clipboard or from a file. If a message is imported from a file, the file can contain just request or response part of one message. For more details, see "How to Add Data Sources" on page 44 .

Multi Response

In addition to the simple simulation of a request-response pattern, when some operations are marked one-way, Service Virtualization is able to simulate a request-response pattern where 0 to n responses are given per request.The number of responses can differ according to the state of the service and can be changed in the Data Model Editor. Service Virtualization permits the learning and editing of a different number of responses, their types and the state of the services (which may affect both the data in the response as well as their number). Multi Response can be used in templates, functions and custom functions, generation of responses from multiple rules, data driving, and activities. The performance simulation is limited to the learning and simulation of the response time of the first response. These features are available on both standalone and embedded servers. The supported protocols are XML and binary services over WebSphere MQ and JMS.

Data Model Editor

The Data Model Editor enables you to configure requests and responses, and service activity calls for individual operations of a Virtual Service during simulation.

Access the Data Model Editor in one of the following ways:

- In the Virtualization Explorer, double-click the Data Model you want to view or edit.

- In the Virtual Service Editor, in the Data Model section, select the model you want to view and click **Edit**.

The Data Model Editor opens for the selected model showing details at the operation level.

Data Model Editor Overview
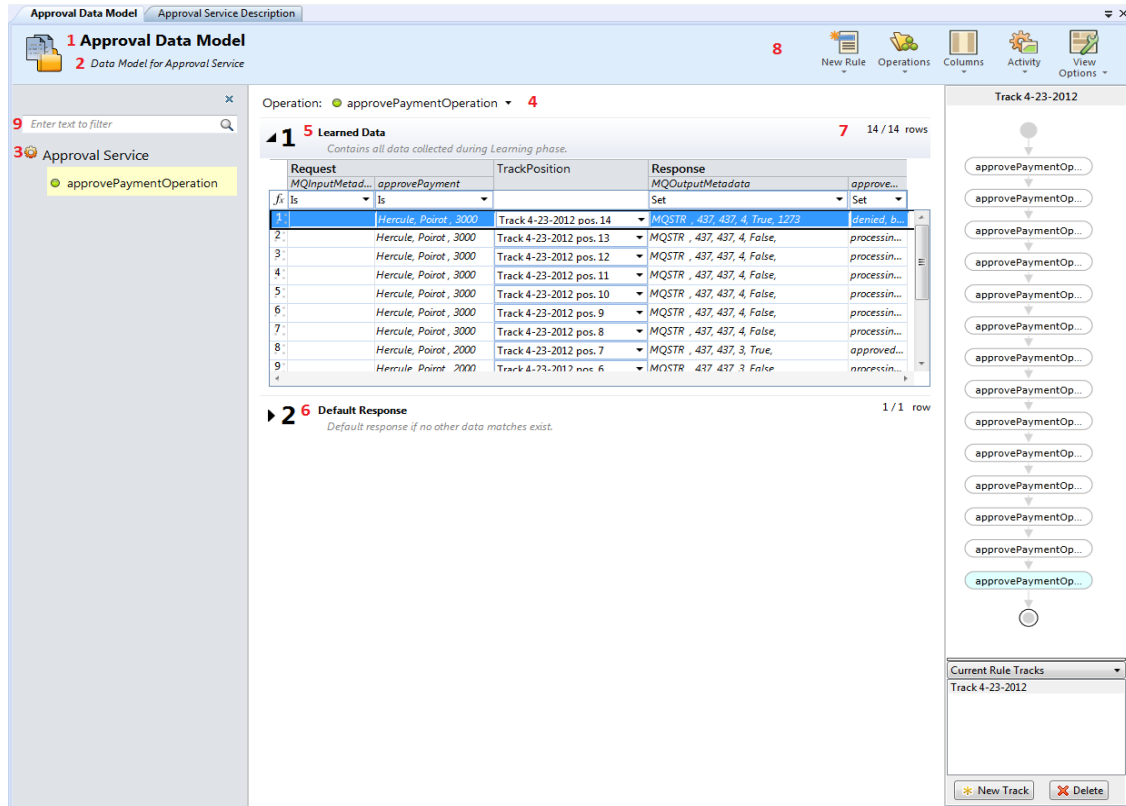
1. Name of data model which can be edited in place.

2. Description of data model which can be edited in place.

3. List of operations for the virtual service associated with the model.

4. List of global rules. A global rule exists across all operations for the virtual service.

5. The rule **Learned Data** contains learned data. It displays the requests and responses learned during the learning state of a virtual service. In general, you do not customize this data but you may want to set conditions to ignore parts of the requests and responses for performance testing and add service call activity.

6. The rule **Default Response** provides a single custom response to apply in cases where there is no other data for a given column or where you want to ignore specific parts of recorded response data.

7. Number of rows in the global rule across all operations.

8. Toolbar with action buttons.

9. Element filter.

Custom Rule
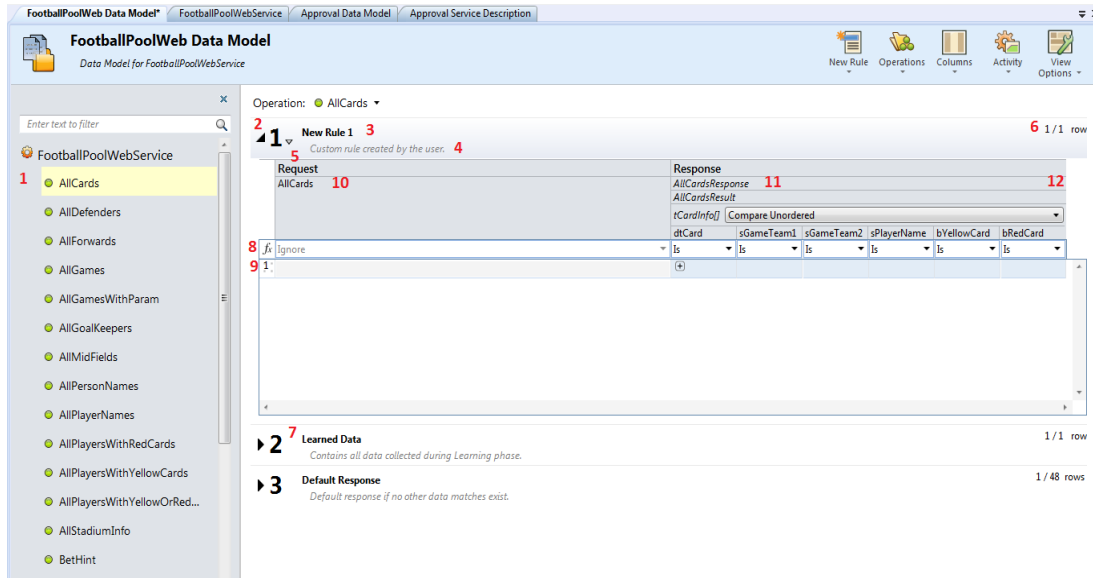
1. Selected operation.

2. Custom rule.

3. Name of the rule. Click to edit.

4. Description of the rule. Click to edit.

5. Action button to change the priority of a custom rule. The priority of the custom rule can be increased or decreased.

6. The term **1/1 rows** indicates that the global rule **New Rule 1** contains one row across all operations and one row for the selected operation in this rule.

7. Priority of the rule. Priority is from 1 to n. 1 is the highest priority.

8. Row with conditions for data columns. Use the drop down controls for each column to control the condition to be applied to the data.

9. Row with data.

10. Request column header.

11. Response column header.

12. Action to collapse whole message section. For example a request, a response or a service activity call.

Stateful View
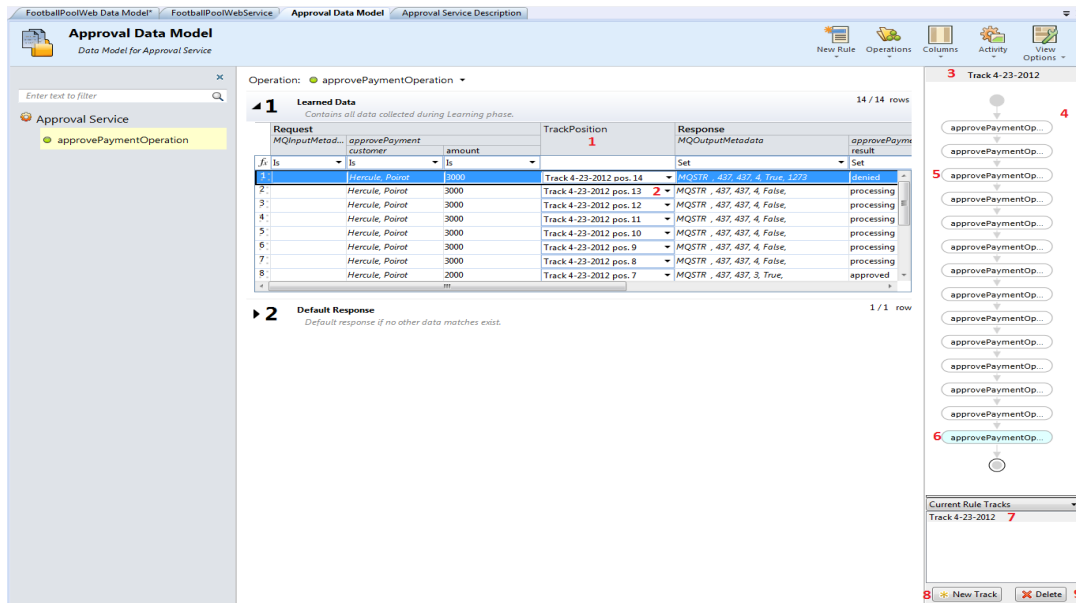
1. **Track Position** column contains information about how rows are used in a stateful simulation. This information is represented by an association with a given track and a position in that track.

2. User can see how the given call (request/response pair) is used during a stateful simulation. User can use **Add to Track** to insert the given call into a sequence of calls in track on a position.

3. Name of track which can be edited in place.

4. Sequence of calls in displayed track.

5. One call of given operation in a sequence of calls for the track.

6. Selected call of a given operation. The associated row for the call is selected in the table and its rule is opened.

7. List of all tracks which model stateful behavior. This list can be filtered by the current opened rule.

8. Button to create a new track to model stateful behavior.

9. Button to delete selected/displayed track.

Rule button menu
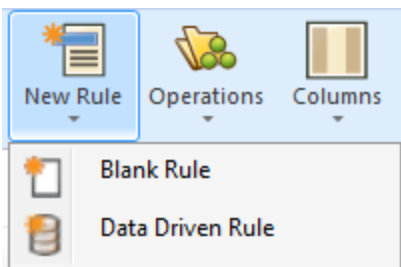
Blank Rule - Create a new global rule with an embedded data source. The global rule exists across all operations.

Data Driven Rule - Use an external file, i.e.: MS Excel, as a data source for a rule. A new read-

only rule is created for this data source. This action opens the **Use Existing Data Source** dialog allowing the user to reference an MS Excel file as a data source with the options of using the first row for column names.

See "How to Manage Rules" on page 41.

Operations/Uri Spaces button menu
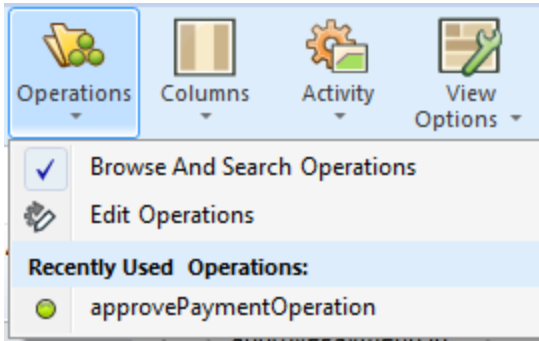
Browse and Search Operations/Uri Spaces - opens/closes left pane with operations/uri spaces list - this pane will require its own chapter as well as there is additional functionality.

Edit Operations/Uri Spaces - opens service description editor allowing to modification of operations/uri spaces, their data formats and metadata.

Recently Used Operations - a list of the most recently used operations for quicker access.

See "Service Description Editor" on page 49.

Columns button menu
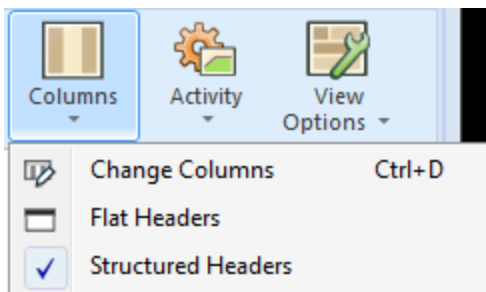
Change Columns - Opens Change Columns dialog. This dialog is used to configure table column visibility. See "Changing Columns" on next page.

Flat Headers - Toggles between flat and structured column headers.

Structured Headers - Toggles between flat and structured column headers.

Activity button menu
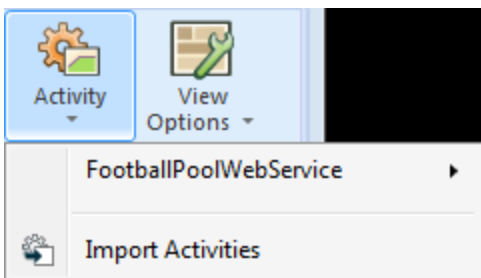
List of imported services whose operations can be used as a service call activity.

Import Activities - Import a service description of a service whose operations can be later used as service call activities.

See "How to Add Service Call Activity" on page 43.

View Options button menu
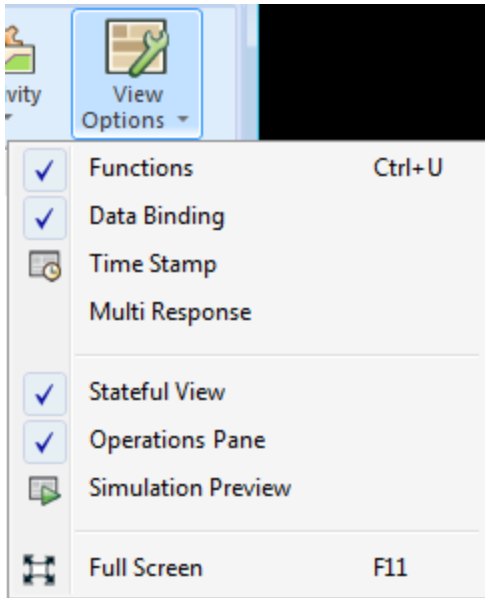
Functions - Displays the row with conditions for data columns.

Data Binding - Display row with binding to a data source in rules, i.e:MS Excel file. Binding is only visible for rules which are connected to an external data source.

Time Stamp - Displays the column in rules. This column contains the time of creation or last modification.

Multi Response - In the case that response data contains more than one response, these are presented. For details, see "Data Model" on page 33

Stateful View - Displays the**Track Position** column in rules and the **Track View**.

Operations - Toggles the display of individual operations in the left hand pane.

Simulation Preview - displays in real time how changes in the Data Model are affecting the simulation.

Full Screen - Switches the **Data Model** editor to the full screen view.

Changing Columns

The Change Columns Dialog is accessed by clicking the **Columns** button in the Data Editor and selecting **Change Columns**, using the hot key **Ctrl+M** or by accessing the context menu on a rule. The main function is to toggle the visibility of structured columns in the data table of a rule by selecting checkboxes corresponding to the required view.

Columns are presented in a tree format and context menus are available allowing easy navigation within large structures.

Columns can be filtered using by entering text in the search field. Arrows on the right of the filter fields provides navigation between found results and clicking **x** clears the filters text.

A header selected in the Change Columns window will be highlighted in the open table.

Note: Changes in columns visibility are applied immediately. Changes can be reverted by clicking **Cancel** and confirmed by clicking **Ok**.

How to Edit Data Models

Each virtual service is associated with at least one data model which can contain the recorded behavior of the service and also customized data for simulation. Each data model contains a set of Rules defining data behavior for each operation in the service and Tracks to determine the order of stateful behavior. For more details about the Data Model, see "Data Model" on page 33.

Prerequisites:

- Create a virtual service. For details, see "How to Create a Virtual Service" on page 14.

- Optional but recommended: Record the behavior of the real service. For details, see "How to Learn Services " on page 26

To Access the Data Model Editor:

- Do one of the following:

   - In the Virtualization Explorer, double-click the Data Model you want to edit.

   - In the Virtual Service Editor, select the Data Model you want to edit, and click **Edit**.

   The Data Model Editor opens displaying the Rules and Tracks for the model. For UI details, see "Data Model Editor" on page 35.

To Change the Model Name or Description:

1. Click the name to make the name and description editable.

2. Input the new name and description.

3. Hit **Enter** to commit your changes.

You can edit various aspects of the Data Model. For details, see the following sections:

- "How to Manage Rules" below
- "How to Add Data Sources" on page 44

How to Manage Rules

Rule Function fields are used to give column data meaning. Setting a function to the column directly modifies the behavior of the simulation. Functions are set on a per column per operation principle. They can be split into two parts:

- *Condition functions*: Is, Ignore, Compare Ordered, Compare Unordered

- *Action functions*: Set, Copy From, Replace Array

The default function for all columns is Ignore: the value does not influence the simulation in any way. Functions can be applied to various levels of a data line. Learned data rules are set as *is* function on requests and *set* function on responses.

When the simulation is in progress, the simulator engine walks through the rule according to rule priorities attempting to find a single line (from each rule) that most precisely conforms to the condition functions used. When a single data line is selected, all the action functions are applied to that data line.

Custom Functions

To define more complex conditions, Custom Functions can be used. These are to be used by a advanced user aware of the complete system structure. Complex conditions can be set with structured query language.

There are two classes of variables: Input ($input_*) and Data ($data_*). Input variables are those present in a processing row during simulation. A more simplified explanation of Input variables is that they are requests and Data variables are present in the data model.

Available variables:

- `$input_string` - original string - is automatically quoted in the place of usage

- `$input_string_unquoted` - original string - is not automatically quoted

- `$input_int` - original string converted to int data type - contains NULL if input string is not of this data type

- `$input_float` - original string converted to float data type - contains NULL if input string is not of this data type

- `$input_date` - original string converted to date data type - contains NULL if input string is not of this data type

- `$data_string`

- `$data_int`

- `$data_float`

- `$data_date`

Examples of custom functions:

- Match request data (cast to integer) smaller than actual value (cast to integer) found in the column `$input_int < $data_int`

- Match request data (cast to integer) smaller than actual value (cast to float) found in the column `$input_int < $data_float`

- Match actual value found in the column (cast to string) equal to 'cat' string `$data_string = 'cat'`

- Match request data (cast to string) equal to 'cat' string `$input_string = 'cat'`

- Match request data (cast to string) equal to actual value (cast to string using an SQL 'LIKE'

operation) `$input_string LIKE $data_string`

- Match request data (cast to string) equal to any string starting with the actual value found in the column `$input_string LIKE $data_string + '%'`

- Match request data (cast to string) containing a substring 'cat' `$input_string LIKE '%cat%'`

- Match request data (cast to date) smaller than actual data `$input_date < $data_date`

- Match request data smaller than actual data OR request data equal to 'dogs' string `$input_date < $data_date` OR `$input_string = 'dogs'`

Simulation Preview

Simulation Preview displays in real time how changes in the Data Model are affecting the simulation. The Simulation Preview is in the **View Options** menu in the Data Model Editor. The Simulation Preview displays a simulation of the message that has the request part equal to the currently selected message in the Data Model Editor. The message is passed to the simulation engine and the result is displayed in the response section.

How to Add Service Call Activity

External services can be called by the Virtual Service. Only SOAP services are supported .

1. In the Data Model, select the operation for **Service Call Activity**.

2. Click **Import Activities** under the Activity menu.

3. Enter the path of the WSDL schema.

4. Click **Finish**.

5. Click on Activity menu again.

6. Select operation of the imported schema.

7. Service Call Activity is added into each rule for the selected operation.

8. You can turn Off/On calling of the service for any rule.

9. Use **Copy From** function on any Service Call Activity request column or Virtual Service response column.

    a. **Choose Copy** from function

    b. Move the cursor over the columns. The cursor type indicates if you can use a column as a source for the Copy From function or not (using Copy From on incompatible data types in not permitted).

    c. Click on a source column.

    d. The name of the source column is now displayed in brackets next to the Copy From function.

10. You can set data in the rule rows for another columns or set other functions available in Data Model Editor. For details, see "Data Model Editor" on page 35.

How to Add Data Sources

Additional data from external data sources can be added to a data model for testing purposes. The supported format for this data is MS Excel (*.xls, .xlsx*).

1. In the Data Model, click **Data Driven Rule** under the **New Rule** menu.

2. Enter the path and filename for the data source.

3. Optional: Use the first row of the table for column names.

4. Give the data source a name.

5. Click **Finish**.

How to Bind an External Data Source:

1. Select **External Data** rule.

2. Expand a table header and its child headers until you have a "leaf" header (only leaf headers can be bound).

   Click **bind data...** on the top of a column (if it is not visible, select **Data Binding** from the View Options menu).

   a. Optionally, an array can also be bound by selecting content that is not yet related to the target sheet. The **Edit Sheet Relations** dialog will appear and a Primary and Foreign Key must be assigned.

   Click **OK** to assign the keys and close the dialog.

3. Click on any column in the displayed external data source table.

4. Repeat previous two steps for other columns.

5. Click **OK**.

How to Refresh an External Data Source:

1. Select the **Refresh Data Source** command from the Data Source menu.

   Alternately:

   Right click on the rule number and select **Refresh Data Source**.

2. Click **OK**.

Note: Data source is refreshed when simulation is started/restarted.

How to change the path of a data source:

1. Select **Change Data Source Path** from the Data Source menu.

   Alternately:

   Right click on the rule number and select **Change Data Source Path**.

2. Enter the path and filename for the data source.

3. Optional: Use the first row of the table for column names.

4. Click **Finish**.

Note: To remove a binding from the table, Click **X** next to the **Bind data** button.

Performance Model

The Performance Model enables you to customize the performance of a service during simulation.

When you create a Virtual Service, Service Virtualization creates a Performance Model associated with it. For details, see "How to Create a Virtual Service" on page 14.

This model is then available to learn the performance of the real service and can be customized to set specific performance rules either for the whole service, or its individual operations.

You can edit the Performance Model in the following ways:

- **Boosters**

  Select to boost an aspect of the service performance (for example, CPU or Network performance) and set a multiplier. Service Virtualization applies the boost to the relevant performance criteria.

- **Basic Performance Criteria**

  Set levels for the following performance criteria for specific operations of the service:

  - Response Time [ms] - the time for the service to process a request and return a relevant response.

  - Threshold [hits/s] - the maximum number of requests and responses the service can process without any impact on performance.

  - Throughput Limit [MB/s] - the maximum data capacity the service can process.

- **Advanced Performance Criteria**

  In addition to the basic criteria, set levels for the following criteria for specific operations of the service:

  - Tolerance [%] - the acceptable range of variation in performance for the operation.

  - Maximum Hits per Second - the maximum number of requests and responses the operation is allowed to process.

  - Maximum Response Time - the maximum time for a response at peak performance levels.

For more details, see "How to Edit Performance Models" on page 48.

Each Virtual Service can have multiple performance models. For details, see "How to Manage Models" on page 33.

Prior to recording or simulation you can select which model to use, including non-customizable models to ignore the performance or simulate the unavailability of a service. For details, see "How to Select Models" on page 27.

Performance Model Editor
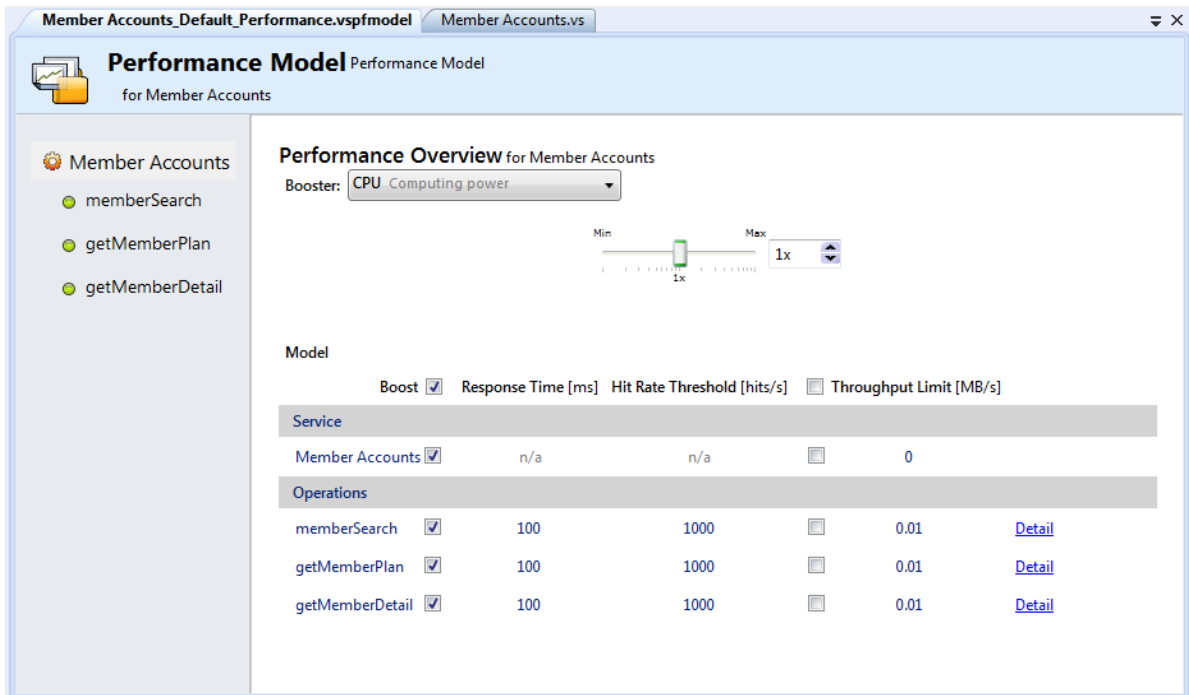
The Performance Model Editor enables you to configure performance metrics for a Virtual Service during simulation. The performance can be configured for the whole service or for its individual operations.

Access the Performance Model Editor in one of the following ways:

- In the Virtualization Explorer, double-click the Performance Model you want to view or edit.

- In the Virtual Service Editor, in the Performance Model section, select the model you want to view and click **Edit**.

The Performance Model Editor opens for the selected model showing details at the service level.

Performance Model Editor - Service Level

The service view of the Performance Model displays the following content and controls:

Performance Model Editor - Service Level Content

Column	Description
Model Name	The name of the model (click to edit).
Service / Operation Hierarchy	The hierarchy on the left shows the service name and a tree of its operations. Click an item in the hierarchy to view its details.
Boosters	A set of boosters to provide high-level control of the operations selected in the operations table.

Column	Description
Booster Controls	The sliding controls and inputs enable you to set the boost level for the selected booster. The setting affects the various performance criteria displayed in the operations table.
Operations Table	Enables you to set more granular settings for individual performance criteria for individual operations. Click **Detail** to open an operation level view of the performance model.

Selecting an operation from the hierarchy or clicking its Detail link in the operations table opens the detail for that operation.

Performance Model Editor - Operation Level
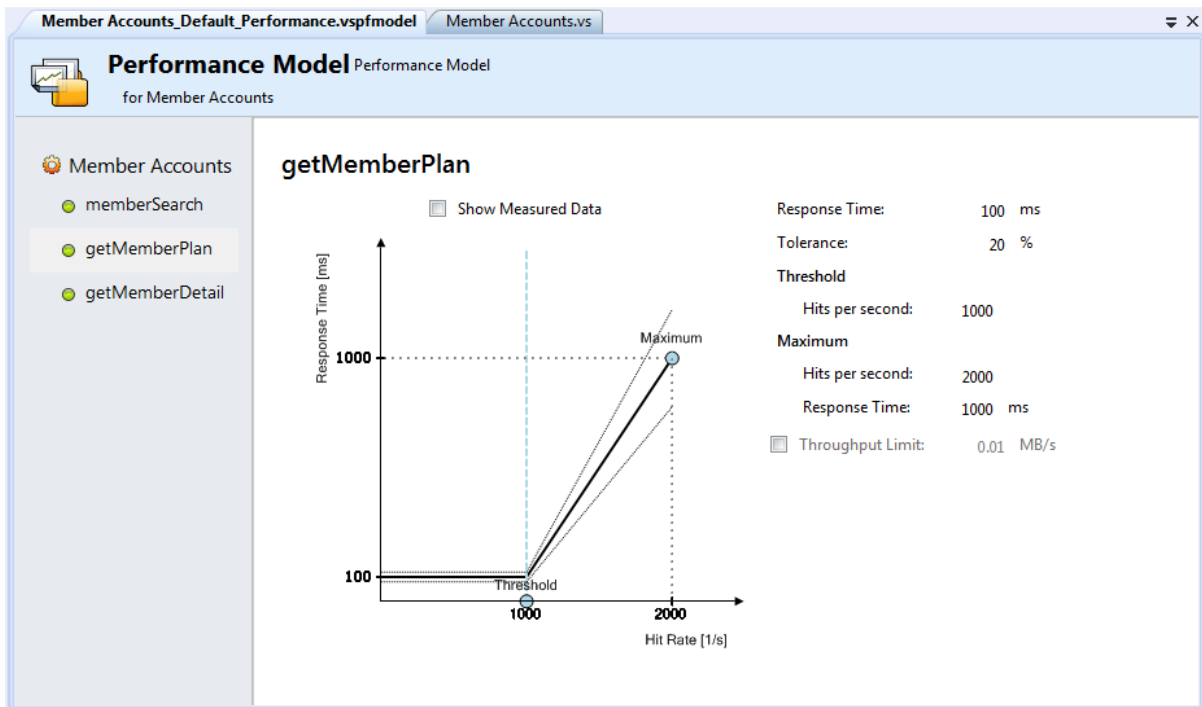
The operation view of the Performance Model displays the following content and controls:

Performance Model Editor - Operation Level Content

Column	Description
Model Name	The name of the model (click to edit).
Service / Operation Hierarchy	The hierarchy on the left shows the service name and a tree of its operations. Click an item in the hierarchy to view its details.
Operation Name	Performance criteria for the operation (click to edit).
Performance Graph	The graph displays the expected performance based on the criteria set for the operation. Select **Show Measured Data** to view any recorded performance data in the graph.

Column	Description
Performance Criteria	Displays the advanced performance criteria for the operation with the option to edit them.

For more details about the Performance Model Editor controls and their impact, see "How to Edit Performance Models" below.

🔧How to Edit Performance Models

Prerequisites:

- Create a virtual service. For details, see "How to Create a Virtual Service" on page 14.

- Optional but recommended: Record the performance of the real service.

To Change the Performance of a Virtual Service:

1. In the Virtualization Explorer, double click on **Performance Model** or once on **Edit** to open the Performance Model

2. Click the **Booster** drop down list to display available options:

- **CPU**: Use the slider, enter a value or use the arrow buttons to assign a CPU power multiplication factor .

- **Network:**Use the slider, enter a value or use the arrow buttons to assign a network throughput multiplication factor.

- **Cluster:**Use the slider, enter a value or use the arrow buttons to assign a scalability multiplication factor.

- **Expert:**Use the sliders, enter values or use the arrow buttons to assign multiplication factors to *Response Time*, *Hit Rate* and *Throughput Limit* values.

- **None:**Turn off all boosters.

Note: Simulation must be restarted to apply changes.

3. Clicking the **Boost** and **Throughput Limit** radio buttons applies the performance changes to the service and all operations. These can also be selected individually by clicking the radio button next to the required service or operation.

Advanced Performance Criteria

1. Click **Detail** or the name of the name of the operation in the side column to open the Operation Detail page for that operation.

2. Modify values by clicking on the value and entering a new value or by dragging the **Maximum** and **Threshold** indicators in the graph.

3. Optional: Click the **Show Measured Data** radio button to present Measured Data can be

presented in the graph.

4. Optional: Click the **Throughput Limit** radio button to change the value and modify the Throughput Limit .

5. Save the modifications with the **File>Save** menu or closing the model.

Service Description Editor

The Service Description Editor is used for editing a virtual service. It can be accessed by double clicking a Service Description in the Virtualization Explorer, clicking Edit in the Service Description section of the Virtual Service Editor, or selecting Edit URI Spaces from the URI Spaces menu of the Data Model Editor.

The Service Description Editor consists of two parts: Service Editing and the URI Space Editing.

Service Editing

Metadata are generally not part of a message body and mainly contain URL parameters and HTTP headers.

Some types of metadata, such as HTTP Headers, are not required for tested applications; they may be important for the protocol, but the application does not require them and they are not learned for purposes of virtualization. By default they are disabled in the request but it is possible to enable them and edit the data in the Data Model Editor for use in a simulation.

Metadata such as URI Parameters are generally used by tested applications and are enabled by default.

"How to Edit Metadata" below

"Working with XML Schema" on next page

Note: Currently, REST is fully supported and WebSphere MQ is partially supported for editing.

URI Space Editing

"How to Add a URI Space" on next page

"How to Delete a URI Space" on next page

"Working with Data Formats" on next page

Note:Not all operations are available for all protocols.

How to Edit Metadata

Click **Edit Request Metadata** or **Edit Response Metadata** in the Service Description Editor to open the Edit Request or Response Metadata window.

Select the item you would like to edit. Items can be filtered be entering text in the filter box.

- Click **Add** to Add New Metadata

1. Enter a name for the metadata

2. Select a type from the drop down box

3. Click **OK**.

- Select metadata and click **Delete** to remove the metadata from an item.

- Click **Enable/Disable** on either an item or its metadata to either activate or deactivate the selection.

- Click **Edit** to change the settings of any metadata.

Working with XML Schema

Default XML Schema are presented in the Service Description Editor and can be edited, added or removed from the service description.

- Click **Add** and select an existing schema (.xsd file) to have it added to available schema for the service description.

- Click **Edit** to open the schema and alter it in an xml editor.

- Click **Delete** to remove the schema from the service description.

How to Add a URI Space

In order to diversify your data, you may want to create more URI spaces to place the data in.

Existing URI spaces are presented in a tree structure on the left side of the Service Description Editor. Clicking on the name of an existing URI space opens it in the URI Space Editor.

Click **Add URI Space** in the left bottom of the Service Description Editor.

Enter a URI path and click **Add**.

Note: Segments are separated with a '/' and the wildcard character '*' is used for several characters or segments. An example being "`Customer/*/Europe/*`".

Caution: If a URI Space that is in conflict with an existing URI Space and the new URI Space is more specific, the conflicting URI Space is split to several new URI Spaces and data from all associated rules are moved to these new URI Spaces.

How to Delete a URI Space

Select a URI Space in the left column of either the Service Description Editor or the URI Space Editor and click **Delete URI Space** in the left column to remove the URI Space.

Working with Data Formats

Data formats can be added and removed from a URI Space as either Request or Response Data Formats.

In either format, click **Add** to open Add Message Type window and optionally:

1. Select **Copy from existing URI Space**.

2. Choose a URI Space from the drop down list.

3. Click **Finish** to add the format.

Or

1. Select **New Data Format**.

2. Choose a format from the drop down list.

3. Choose a type from the second drop down list.

   a. Optionally, click **External File** to import schema other sources.

4. Click **Finish** to add the format.

Chapter 5

Composite Application Topology

The Topology Editor and Tools enables you to model composite applications by creating a visual map of services, group them into larger composites, mark their types, and display the service calls between them. This document refers to these maps as a Topology.

This Chapter includes:

How to Add Topologies

To Add Topologies to the Virtualization Project:

1. In the Virtualization Explorer, open the project shortcut menu, and select **Add>Topology** or from the File menu, select **New>Topology** to open the New File dialog box.

2. Select **Topology**, input a name, and click **Add**.

   Service Virtualization creates a new topology as part of the virtualization project, and opens the Topology Editor view and Tools. A default topology contains a service calling a limited access service.

How to Model Composite Applications

Prerequisites:

- Add or import a topology to your project. For details, see "How to Add Topologies" above

To Edit Service Topologies:

1. Open the Topology Editor view.

   In the Virtualization Explorer, double-click a topology to open the Topology Editor and Tools.

---

2. Add items to the Topology.

   Drag and drop the required items from Tools to the Editor.

   Tools contains the following items:

   - ☀ **Service**

     A service with no particular notation.

   - ☀ **Limited Service**

     A service marked as having limited access.

   - ✹ **Secured Service**

     A service marked as requiring some authentication.

   - ▪▪ **Group**

     A generic box enabling you to organize services into larger composites for visual purposes.

3. Rename Services and Groups.

   Open the context menu for a topology item and select Rename, select the item and press F2 or double click the text.

4. Connect services together with Service Call Connectors.

   Move the cursor to the right edge of the calling service until you see the hand icon, and then click and drag the connector to the called service.

5. Mark relevant services as limited or secured.

   Open the context menu for a service to use the following options:

   - ◈ **Set / Unset Limited Access**

     Mark or unmark a service as having limited access.

   - ⌗ **Set / Unset Secured Access**

     Mark or unmark a service as requiring authentication.

Other Context Actions:

Context Action	Icon	Description
Test	🧪	Begin testing of the service.
Preview Test Impact	🔺	Switch On / Off the Test Impact highlighting.
Learn and Simulate		Shortcut combo for Create Virtual Service and Learn.
Create Virtual Service		Starts the Create Virtual Service wizard.
Rename (or double-click the service)		Give a different name to the service.
Delete		Remove the service from the topology.

🔧How To Test Composite Applications

In the topology view of a composite application, you can initiate a step-by-step process for testing a particular service which guides you through all the processes required to virtualize, learn, and simulate the limited access services that the tested service calls.

Prerequisites:

● Model your composite application by editing the topology, ensuring that you mark the limited and secured services. For details, see "How to Model Composite Applications" on page 52.

To Test Services in a Composite Application:

●
In the Topology Editor, open the context menu for the services you want to test and select 🧪
Test.

The Tasks view opens showing a guided process for end-to-end setup and configuration for testing the composite application. For details, see the following topics:

1. "How to Virtualize Services in the Tasks View" below

2. "How to Reconfigure Clients in the Tasks View" on next page

3. "How to Learn Services in the Tasks View" on next page

4. "How to Simulate Services in the Tasks View" on page 56

🔧How to Virtualize Services in the Tasks View

If you have not yet created virtual services for limited access services or if there is some missing information, you are prompted to provide this information.

Prerequisites:

1. Model your composite application by editing the topology, ensuring that you mark the limited and secured services. For details, see "How to Model Composite Applications" on page 52.

2. Mark at least one service as being tested. For details, see "How To Test Composite Applications" above.

To Virtualize Services in the Tasks View:

1. To virtualize services, do one of the following:

   ■ To provide all the missing information for all virtualized services, click **Virtualize Services**.

   ■ To provide the missing information for selected virtualized services, expand **Customize Virtualization of Services**, select the Services to virtualize, and click **Virtualize Selected Services**.

   The Virtualize Services dialog box opens.

2. *Optional:* The Virtualize Services dialog box displays the services requiring additional information. Click **Next** to continue.

> **Tip:** Select **Do not show this message again** if you want to skip this information panel in future testing.

3. **Specify Virtualization for Service**:

   You may either select an existing virtual service from the drop down list or create a new virtual service. See "How to Create a Virtual Service" on page 14.

When all the missing information for each service is complete, Virtualize Services is marked as complete and you proceed to Reconfigure Clients. For details, see "How to Reconfigure Clients in the Tasks View" below.

Service Virtualization adds the virtual services, any new service descriptions, and a data and performance model for each service to the Virtualization Explorer. The Runtime View also updates to include the new virtual services.

How to Reconfigure Clients in the Tasks View

In cases where Service Virtualization can only perform intrusive virtualization, you must reconfigure calling services to use the virtual services instead of the real ones.

Prerequisites:

1. Model your composite application by editing the topology, ensuring that you mark the limited and secured services. For details, see "How to Model Composite Applications" on page 52.

2. Mark at least one service as being tested. For details, see "How To Test Composite Applications" on previous page.

3. Provide any missing information required to virtualize limited and secured services. For details, see "How to Virtualize Services in the Tasks View" on previous page.

To Reconfigure Clients:

1. Click **View** to open the Instructions to Reconfigure Clients dialog box showing the particular details for the virtualized services in your composite application.

2. *Optional:* Click **Save As** to save a text document copy of these instructions.

3. Click **Mark as Completed** when you have reconfigured the tested service and are ready to proceed to Learn Services. For details, see "How to Learn Services in the Tasks View" below.

How to Learn Services in the Tasks View

When your virtual services are properly setup and the calling services reconfigured to use them, you can learn the behavior of the composite application.

> **Note:** If you already have simulation models for your composite application, you can click **Skip to Simulate Services** to proceed without recording any additional service communication. For details, see "How to Simulate Services in the Tasks View" on next page.

Prerequisites:

1. Model your composite application by editing the topology, ensuring that you mark the limited and secured services. For details, see "How to Model Composite Applications" on page 52.

2. Mark at least one service as being tested. For details, see "How To Test Composite Applications" on page 54.

3. Provide any missing information required to virtualize limited and secured services. For details, see "How to Virtualize Services in the Tasks View" on page 54.

4. Reconfigure any components that call the virtualized services. For details, see "How to Reconfigure Clients in the Tasks View" on previous page.

To Learn Services:

1. Click **Learn Services** to put the virtual services into Learning Mode.

2. Run your test through the composite application using a client or test script. Service Virtualization records the requests and responses for each virtualized service and creates a simulation model for each.

As you run your test, the Runtime View updates the details for each virtual service.

When you have finished recording, you are ready to proceed to Simulate Services. For details, see "How to Simulate Services in the Tasks View" below.

How to Simulate Services in the Tasks View

When you have a simulation model, you can simulate the behavior of the real services without using them. The tested service calls the virtual services and returns results based on the simulation model instead of actual responses from the real services.

Prerequisites:

1. Model your composite application by editing the topology, ensuring that you mark the limited and secured services. For details, see "How to Model Composite Applications" on page 52.

2. Mark at least one service as being tested. For details, see "How To Test Composite Applications" on page 54.

3. Provide any missing information required to virtualize limited and secured services. For details, see "How to Virtualize Services in the Tasks View" on page 54.

4. Reconfigure any components that call the virtualized services. For details, see "How to Reconfigure Clients in the Tasks View" on previous page.

5. Learn the behavior of the virtualized services. For details, see "How to Learn Services in the Tasks View" on previous page.

To Simulate Services:

1. Click **Simulate Services** to put the virtual services into simulate mode.

2. Run your test through the composite application using a client or test script. Service

Virtualization processes the requests to each virtualized service and returns responses based on the simulation model for each.

As you run your test, the Runtime View updates the details for each virtual service.

⚒How to Create Virtual Services in Topologies

Prerequisites:

- Your topology must contain at least one service. For details, see "How to Model Composite Applications" on page 52.

To Create Virtual Services:

1. In the Topology Editor, open the context menu for the service you want to virtualize, and select **Create Virtual Service**.

2. The Virtualize Services wizard steps through the following dialogs for the service, depending on which information is required:

   Specify Virtualization for Service

   a. You may either select an existing virtual service from the drop down list or create a new virtual service. See "How to Create a Virtual Service" on page 14.

   b. **Add Service Security Settings**

   If a service is marked as secure and does not already have any associated authentication, Service Virtualization cannot access the service and prompts you to provide authentication. For details, see "How to Set Authentication Credentials" on page 22.

When the service contains all the required information the topology diagram updates to indicate that the service is virtualized.

When you virtualize services, the components that call them may require reconfiguration. For details, see "How to Reconfigure Clients in Topologies" below.

⚒How to Reconfigure Clients in Topologies

When you virtualize a service in the topology editor, the editor indicates that the calling components may require reconfiguration by adding an exclamation icon to the calling component.
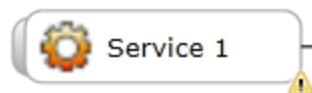
Prerequisites:

- Your topology must contain at least two services. For details, see "How to Model Composite Applications" on page 52.

- A called service must have been virtualized. For details, see "How to Create Virtual Services in Topologies" on previous page.

To Reconfigure Clients:

1. In the Topology Editor, click the exclamation icon to open the Reconfigure Service pop-up.

    The pop-up displays the endpoint details for the real and virtual services the component calls, providing the information you require to reconfigure the calling component.

2. When you have reconfigured the calling component, click **Mark Completed** and close the pop-up.

    The exclamation icon is removed from the calling component.

🔧How to Learn Service Behavior in Topologies

In the topology view of a composite application you can learn the behavior of individual services.

Prerequisites:

- Edit the topology, marking limited and secured services, and importing service descriptions for the services you want to learn. For details, see "How to Model Composite Applications" on page 52.

To Learn Service Behaviour:

1. In the Topology Editor, open the context menu for each service you want to record and select **Learn**. The topology diagram changes, displaying the service in Learning Mode.

2. Run your test through the composite application using a client or test script. Service Virtualization records the requests and responses for the virtualized services and creates simulation models for each one.

    As you run your test, the Runtime View displays details for each virtual service.

3. When you have finished recording, open the context menu for the services you are recording and select **Stop Learning**. The topology diagram changes, displaying the service in Simulate Mode. Optionally, select **Simulate** and the application stops the learning process and switches directly to simulating.

⚒How to Simulate Service Behavior in Topologies

In the topology view of a composite application you can simulate the behavior of individual services.

Prerequisites:

1. Edit the topology, marking limited and secured services, and importing service descriptions for the services you want to simulate. For details, see "How to Model Composite Applications" on page 52.

2. Learn the behavior of the services you want to simulate. For details, see "How to Learn Service Behavior in Topologies" on previous page.
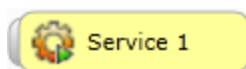
To Simulate Service Behavior:

1. In the Topology Editor, open the context menu for each service you want to record and select **Simulate**. The topology diagram changes, displaying the service in Simulate Mode.
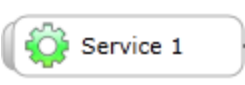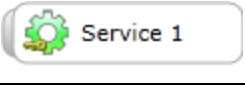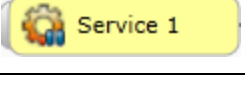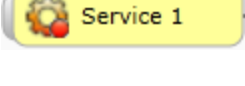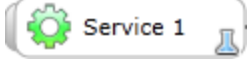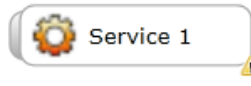
2. Run your test through the composite application using a client or test script. Service Virtualization processes the requests to each virtualized service and returns responses based on the simulation model for each.

   As you run your test, the Runtime View displays details for each virtual service.

📌Topology Editor Reference

The annotation for each service varies depending on its settings and various stages of configuration.

Service Diagram	Description
	An empty service which does not contain an associated service description.
	A service marked as having limited access.
	A service marked as secure, requiring credentials to access.
	A virtualized service.
	A virtualized service in Learning Mode. Service Virtualization records any requests and responses through this service and adds them to the associated Simulation Model.

Service Diagram	Description
Service 1	A virtualized service in Simulation Mode. Service Virtualization monitors any requests to this service and returns responses based on the associated Simulation Model.
Service 1	A service marked as being tested.
Service 2	A service suggested for virtualization.
Service 1	A service requiring attention because it calls a virtualized service and may require reconfiguration to call the virtual service instead of the real one.

Chapter 6

♣Supported Technologies And Environments

An agent represents a component between a client and a real service. It maintains endpoints where client communication is received.

See "How to Change Agent Configuration" on next page.

There are a few agent types in Service Virtualization:

HTTP(S) Gateway Agent

Serves to virtualize HTTP communication. A virtual HTTP endpoint is created to mediate between a client and a real service HTTP endpoint. During the learning process, real communication is forwarded to a real service HTTP endpoint and the communication is recorded. See "Configuring HTTP(S) Gateway Agent" on next page.

HTTP(S) Proxy Agent

Serves to virtualize HTTP and HTTPS communication. No endpoint is created, and an HTTP(S) proxy is used to receive and forward client communication to a real service HTTP or HTTPS endpoint. See "Configuring HTTP(S) Proxy Agent" on page 64

TIBCO EMS Non Intrusive Agent

Serves to virtualize JMS communication in TIBCO Enterprise Message Service ™ (TIBCO EMS). No endpoint is created and a client application does not require reconfiguration. The agent listens to system topics where all communication can be monitored. When the service is switched to Simulating mode, it manipulates the service JMS account permissions in the JMS bus (EMS) to forbid a real service from receiving client communication. Administrator account credentials in TIBCO EMS are required. See "Configuring TIBCO EMS Non Intrusive Agent" on page 66

Generic JMS Agent

Serves to virtualize JMS communication in any JMS provider (IBM® WebSphere® MQ, Weblogic, JBoss, ...). It uses general JMS API and JNDI to lookup and work with JMS resources (context factories, connection factories, queues, topics). It loads Java JVM (Java Virtual Machine) with JMS provider libraries required for JMS communication in process. See "Configuring Generic JMS Agent" on page 67

Note: Generic JMS Agent only supports JMS BytesMessage and TextMessage according to the SOAP over JMS specifications.

IBM® WebSphere® MQ

See "Configuring WebSphere MQ Agent" on page 69.

Forwarding HTTP(S) Gateway/Proxy Agent Communication through HTTP Proxy

See "Forwarding HTTP(S) Gateway/Proxy Agent Communication through HTTP Proxy" on page 71

Configuration of Windows Firewall and HTTP Settings

How to Change Agent Configuration

Agent configuration in Designer UI

If using the embedded server, the agent configuration can be changed in the Designer UI from the menu: **Tools>Options>Agents**.

Agent configuration in configuration files.

Agent configurations for both Embedded and Standalone servers are stored separately in the file system.

- Embedded server agent configuration at:

  - `%APPDATA%\Hewlett-Packard\VirtualServiceDesigner\Agents\AgentConfigurations.xml`

- Standalone server agent configuration at:

  - `%[INSTA-LLLOCATION]%\StandaloneServer\bin\Agents\AgentConfigurations.xml`

To change the agent configuration, you must change the agent configuration file manually. To apply the changes in the configuration files, restart the changed server.

Configuring HTTP(S) Gateway Agent
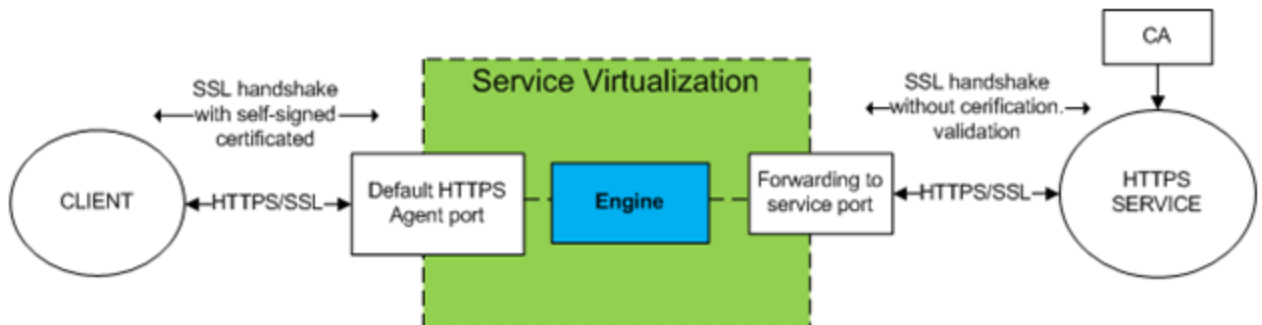
To configure the HTTP(S) Gateway Agent:

From the **Tools** menu, select **Options**, click the **Agents** tab and choose **HTTP** Gateway from the tree menu.

There are two configuration options:

- Port: The port of the Service Virtualization Server used by the agent to deploy virtual services.

- Forward Unknown Request To: The URL prefix that forwards requests other than the requests to virtual services.
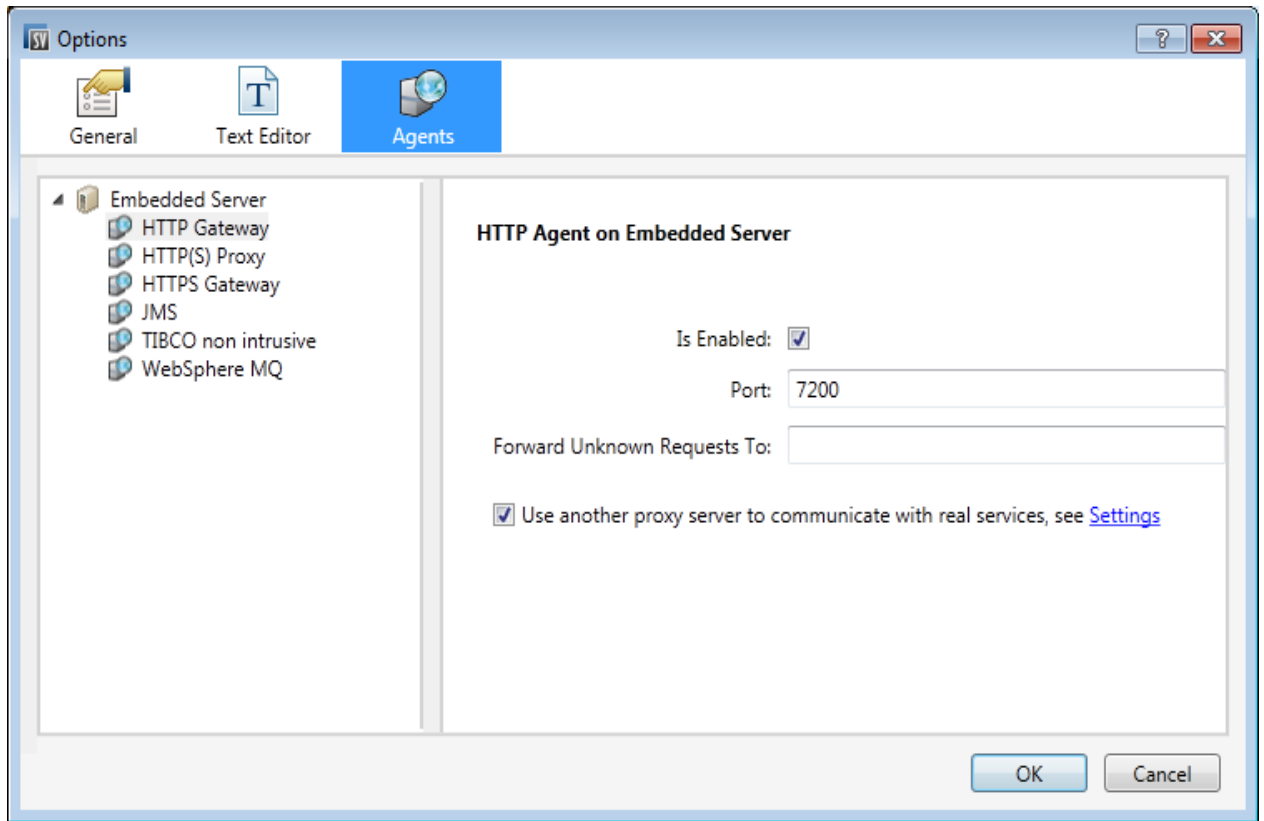
In the case you are not able to reconfigure clients on a service basis (ie: changing the endpoint of each service to a virtual service) and all of your HTTP(S) services reside on a single host, you can use the **Forward Unknown Request To** option of the HTTP(S) Gateway agent.

Example:

The client is calling several backend services:

```
http://esb.demo.hp.com:8080/BackendServices/MemberAccounts

http://esb.demo.hp.com:8080/BackendServices/ExchangeRate

http://esb.demo.hp.com:8080/BackendServices/Approval
```

You are only virtualizing the MemberAccounts service but are only able to reconfigure the application to use another host for all services rather than changing the endpoint of just the one MemberAccounts service in the application.

You reconfigure your application to use the SV Server HTTP Gateway at:

```
http://svserver.hp.com:7200 instead of http://esb.demo.hp.com:8080
```

The application will access backend services at these endpoints:

```
http://svserver.hp.com:7200/BackendServices/MemberAccounts

http://svserver.hp.com:7200/BackendServices/ExchangeRate

http://svserver.hp.com:7200/BackendServices/Approval
```

You create the MemberAccounts service so this functions but the other services would be inaccessible for the application until you virtualized all of them.

To avoid the virtualization of all backend services set the DEFAULT TARGET HOST to

`http://esb.demo.hp.com:8080`

Now all requests to non-virtualized services are forwarded to the
`http://esb.demo.hp.com:8080` host and reaching the real services you do not intend to
virtualize now.

Note: The Forward Unknown Request To can contain base URL in several formats: the host,
optional port and optional base path, i.e.: `http://esb.demo.hp.com,`
`http://esb.demo.hp.com:8080`

Additional HTTPS Gateway Agent configuration.

The HTTPS Gateway agent requires the same steps as the HTTP Gateway agent.

HTTPS requires assigning a certificate to a port used for listening.

Generate a certificate with a private key (if you don't have one), import the certificate to either a
current user's personal store (for an embedded server) or to a local machine's personal store (for
a standalone server). Grant access to the private key to the current user (for an embedded
server) or to the account running the standalone server.

To configure HTTP ports and to install a self-signed SSL certificate we provide the tool
`configureHttpAgent.bat`.

See How to Install Self-Signed Certificate in the installation guide.

To install a custom certificate we provide the tool `addCustomCertificate.bat`.

See How to Install Custom Certificate in the installation guide.

To set agent configuration properties, see "Configuring HTTP(S) Proxy Agent" below.

🔧Configuring HTTP(S) Proxy Agent

The HTTP(S) proxy agent dynamically generates certificates for requested hosts on the fly. The
certificates are signed by the configured certificate authority (CA). The CA certificate and private
key can be configured in the HTTP(S) Proxy agent configuration.

The client must trust certificates signed using a configured CA otherwise the communication may
fail due to the agent rejecting it.

HTTP(S) Agent Configuration provides options as per the following image:
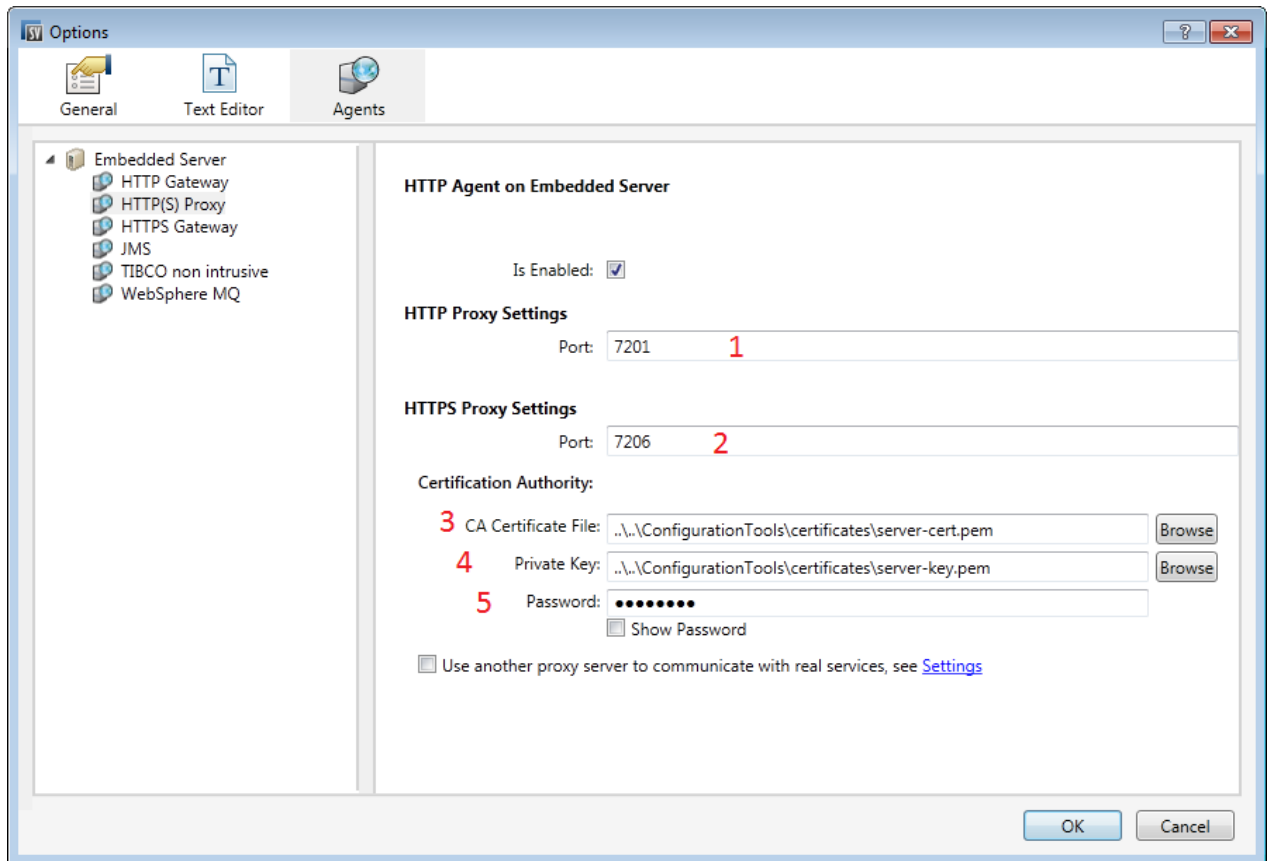
1. HTTP Proxy Settings Port – port for HTTP communication

2. HTTPS Proxy Settings Port - port for HTTPS communication

   Certification authority

3. CA Certificate File - file used to generate host certificates

4. Private key – private key for above configured file

5. Password – password for private key

Note: In case of Java based clients running on JDK 6 or newer and in the case of self-signed certificates, it may be necessary to add the "`-Dsun.security.ssl.allowUnsafeRenegotiation=true`" startup parameter or the client may reject communication with the HTTPS Agent. See http://java.sun.com/javase/javaseforbusiness/docs/TLSReadme.html for more details.

🔧Configuring TIBCO EMS Non Intrusive Agent
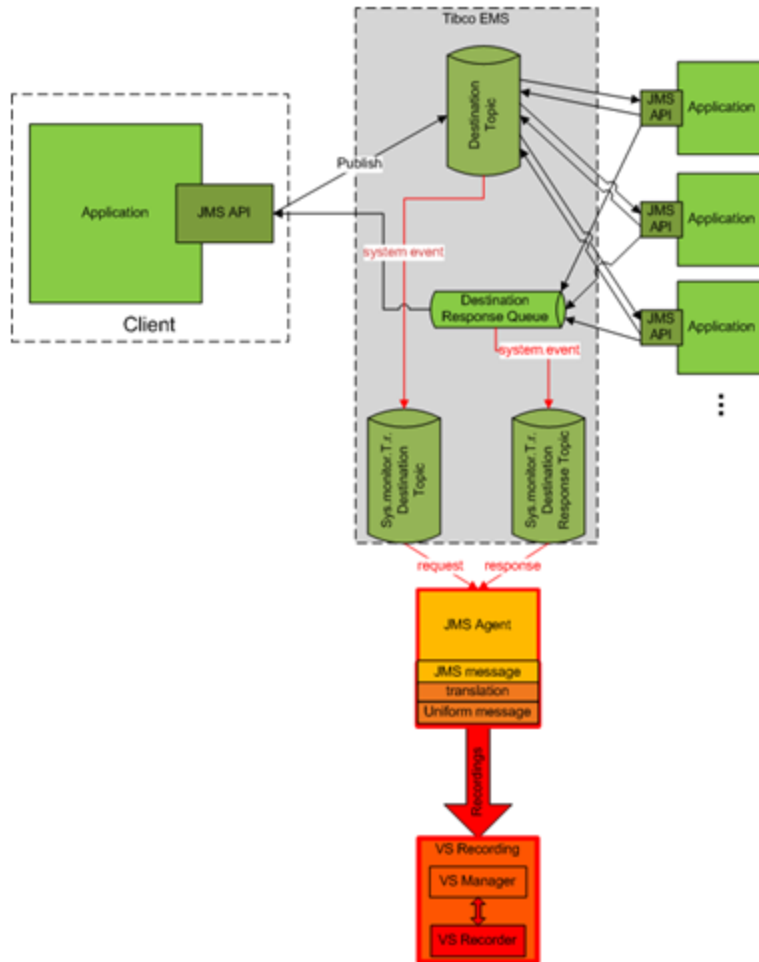
The TIBCO EMS Non Intrusive agent is not installed by default and if required, must be installed manually. The TIBCO EMS Non Intrusive agent requires 2 DLL libraries that are not supplied with the product: `Tibco.EMS.dll` and `Tibco.EMS.Admin.dll`. These libraries are supplied with the installation of TIBCO Enterprise Message Service ™ (EMS) and can be found in the `bin` directory in the EMS installation folder. The typical default location is `c:\tibco\ems\6.0\bin\`.

1. To install the TIBCO EMS agent, ensure that the TIBCO EMS libraries `Tibco.EMS.dll` and `Tibco.EMS.Admin.dll` are either registered in the Global Assembly Cache (GAC) or copied to the `Designer\bin` and `StandaloneServer\bin` directories in the application installation directory. Installation to GAC should be an optional part of the TIBCO EMS installation.

   The application must be restarted after libraries have been copied to the directory.

   When the agent is installed properly, configuration needs to be setup according to EMS specifics.

2. Setting agent configuration properties:

a. *Provider Url* – Host and Server Port where EMS is running.

b. *Credentials* – Username and Password for EMS account with appropriate privileges to change permissions on destinations and the JMS server (admin account).

Configuring Generic JMS Agent
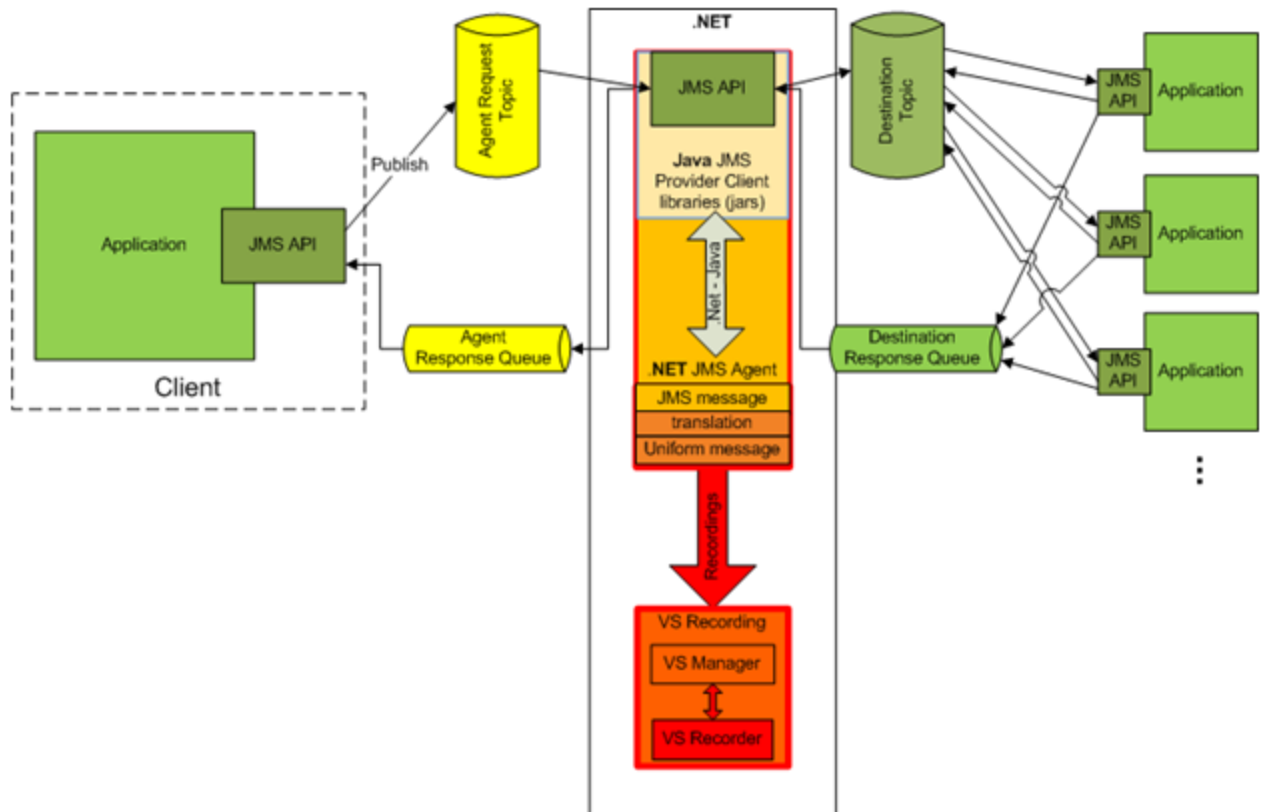
A standard JMS API with JNDI lookups is being used in Generic JMS Agent. Environment with JMS resources must be configured first:

1. Create virtual destination(s) in JMS server (Webpshere MQ, Weblogic, JBoss, …)

   a. Create Virtual Request destination

   b. Create Virtual Reply destination, only if you want to use permanent ReplyTo destination, otherwise temporary ReplyTo destination will be used

2. Configure JNDI mapping for virtual destination(s) to allow agents lookup destination(s) in JNDI.

   a. For Virtual Request destination

   b. For Virtual Reply destination only if permanent ReplyTo destination is used

3. When the JMS environment and resources are configured, the agent configuration needs to be setup according to JMS provider specifics.

   Setting Generic JMS Agent configuration properties:

JNDI URL – URL where JNDI provider and JNDI context with JMS resources is located

Context Factory – provider specific context factory

Class Path – class path with all necessary JMS provider specific libraries for JMS implementation

Username

Password

Note: To ensure maximum compatibility use the same JMS/J2EE jar libraries for the agent classpath that are in use by the client application. If not possible, please follow your application server documentation for selection of correct J2EE/JMS libraries.

Example of Generic JMS Agent configuration:

Agent configuration for WebLogic 10.3:

- JNDI URL: t3://czvm58.devlab.ad:7001/

- Context Factory: weblogic.jndi.WLInitialContextFactory

- Class Path:

  ```
  C:\Temp\WL103\wlthint3client.jar
  ```

  Note: The above sample used WebLogic Thin T3 Client. In case of issues, please use other Weblogic Client libraries (e.g. "WebLogic Full Client" using "wlfullclient.jar"). See chapter 2 of "Overview of Stand-alone Clients" in your "Oracle® Fusion Middleware Programming Stand-alone Clients for Oracle WebLogic Server 11g Release 1" documentation for more details.

Agent configuration for MQ-7.0.1.3 on WAS-6.1.0:

- JNDI URL: corbaloc::czvm24.devlab.ad:2809/NameServiceServerRoot

- Context Factory: com.ibm.websphere.naming.WsnInitialContextFactory

- Class Path:

  ```
  C:\Temp\WAS6\com.ibm.mq.jar;

  C:\Temp\WAS6\com.ibm.mq.jmqi.jar;

  C:\Temp\WAS6\com.ibm.mqjms.jar;

  C:\Temp\WAS6\com.ibm.ws.admin.client_6.1.0.jar;

  C:\Temp\WAS6\com.ibm.ws.runtime_6.1.0.jar;

  C:\Temp\WAS6\connector.jar;

  C:\Temp\WAS6\dhbcore.jar;

  C:\Temp\WAS6\fscontext.jar;

  C:\Temp\WAS6\ibmorb.jar;

  C:\Temp\WAS6\jms.jar;

  C:\Temp\WAS6\jndi.jar;
  ```

```
C:\Temp\WAS6\ldap.jar;

C:\Temp\WAS6\providerutil.jar
```

Agent configuration for JBoss 6.0:

- JNDI URL: jnp://[machine-name]:1099/

- Context Factory: org.jnp.interfaces.NamingContextFactory

- Class Path:

```
C:\Temp\JBAS6\concurrent.jar;

C:\Temp\JBAS6\hornetq-core-client.jar;

C:\Temp\JBAS6\hornetq-jms-client.jar;

C:\Temp\JBAS6\jboss-client.jar;

C:\Temp\JBAS6\jboss-ejb3-core-client.jar;

C:\Temp\JBAS6\jboss-ejb3-ext-api.jar;

C:\Temp\JBAS6\jboss-jms-api_1.1_spec.jar;

C:\Temp\JBAS6\jboss-logging.jar;

C:\Temp\JBAS6\jnp-client.jar;

C:\Temp\JBAS6\netty.jar
```

🔧Configuring WebSphere MQ Agent
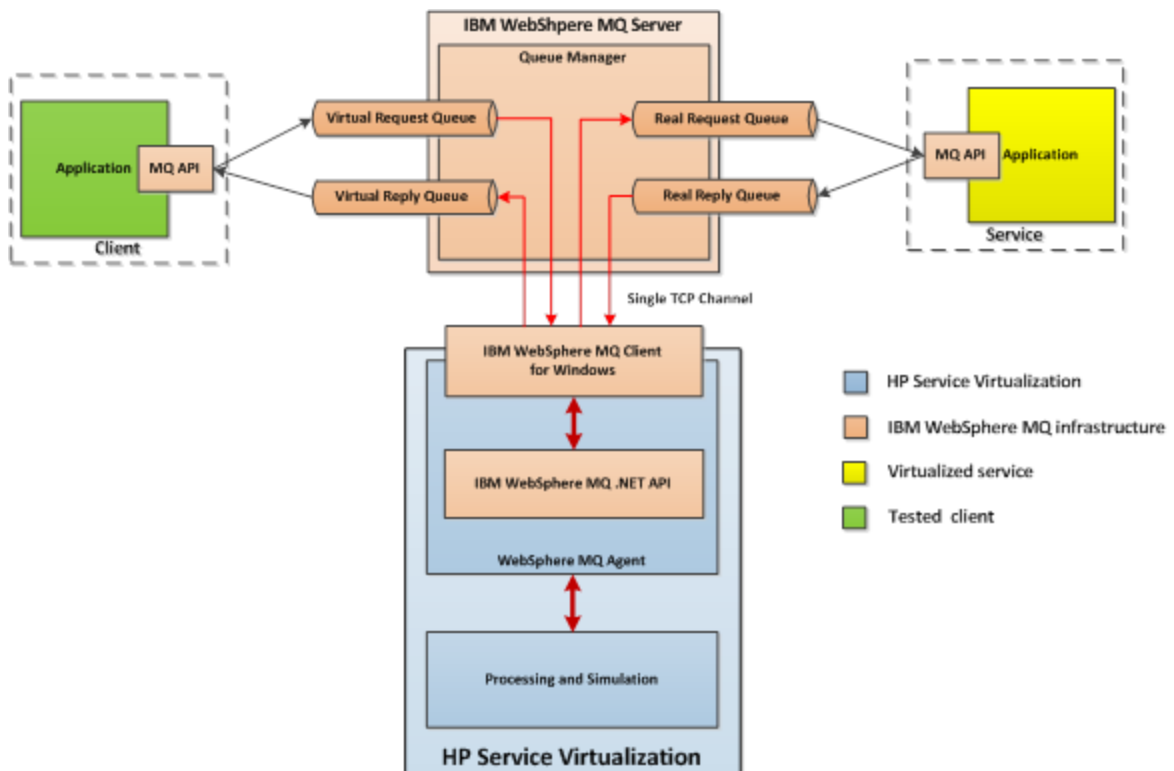
Prerequisites for using WebSphere MQ Agent:

The WebSphere MQ Agent is not installed by default. The agent is dependent on the IBM WebSphere MQ Client 7.0.1.6 or newer (`amqmdnet.dll` library with version 1.0.0.3 and its dependencies). The IBM WebSphere MQ Client full installation places the required library in the Global Assembly Cache (GAC) making the agent available (if the client is installed while the server or the designer is running, you must restart the application prior to enabling the agent).

Because the WebSphere MQ Agent uses the IBM WebSphere MQ Client internally, you can use all diagnostic tools and logs provided by the client if you need to investigate any issues with WebSphere MQ communication between HP Service Virtualization and the IBM WebSphere MQ Server.

To configure WebSphere MQ agent:

From the Tools menu, select Options and the Agents tab.

The following options are available:

- *IsEnabled* – turn on or off the agent. The agent can be turned on only if prerequisites were fulfilled.

- *Queue Manager Name* – the name of the queue manager the agent will be connecting to.

- *Channel Name* – the name of the TCP channel the agent will use to connect to the queue manager.

- *Host* – the name of the server hosting the queue manager.

- *Port* – the TCP port where the channel listens for client connections.

- *CCSID* – the character set used by the host.

- *User name* – the user connecting to the queue manager.

- *Password* – the password for the user connecting to the queue manager.

Additional global configuration

There are additional configurations available through the application configuration files:

```
%[INSTA-
LLLOCATION]%\StandaloneServer\bin\HP.SV.StandaloneServer.exe.config
```

```
%[INSTALLLOCATION]%\Designer\bin\VirtualServiceDesigner.exe.config
```

The following configuration elements can be defined in the appSettings section in those configuration files:

```
<add key="MQAgent.DefaultCharacterSet" value="0"/>
```

```
<add key="MQAgent.UseUtfMethods" value="false"/>
```

```
<add key="MQAgent.StripXmlDeclaration" value="true"/>
```

These configuration elements setup global behavior for processing WebSphere MQ messages:

- *MQAgent.DefaultCharacterSet* – configures default character set for simulated XML responses. This configuration is used if the character set is not recorded as message metadata. If the value is set to 0 the character set is not set.

- *MQAgent.UseUtfMethods* – HP Service Virtualization supports binary and XML messages over WebSphere MQ. When using an XML service, the message can be either transferred as a string (with a specified character set) or as binary data (UTF). By default the agent reads and writes messages as a string. If binary messages with UTF content should be used instead, set the value to true.

- *MQAgent.StripXmlDeclaration* – when the value is configured to true, XML messages sent from the agent will not contain XML declarations.

Agent limitations

- The WebSphere MQ Agent instance can only use queues from a single Queue manager.

- The WebSphere MQ Agent instance can only use a single TCP channel to connect to the Queue manager.

- CCSID configuration in the WebSphere MQ Agent instance requires an application restart.

- CCSID configuration is global for whole the application (there cannot be two agents with different CCSIDs).

- The WebSphere MQ Agent is not able to automatically reconnect when a TCP connection to the WebSphere MQ Server is lost. Each service using the agent must be restarted (alternately, restart the whole application).

Forwarding HTTP(S) Gateway/Proxy Agent Communication through HTTP Proxy

It is possible to forward the HTTP(S) communication between HTTP(S) Proxy or Gateway agents and real service through additional proxy (proxy chaining).

To enable proxy chaining, perform following steps:

1. From the Tools menu, select options and the Agents tab.

2. From either the HTTP Gateway, HTTP(S) Proxy or HTTPS Gateway settings, select the **Use another proxy server to communicate with real services** options and click the **Settings** link.

The following options are available:

- *Proxy host and Port* – Address or hostname of proxy machine and port number.

   Note:With the HTTP Proxy agent, both HTTP and HTTPS communication is forwarded through same port.

- *Credentials* – Credentials for authentication. Authentication is automatically detected and supports BASIC, DIGEST, NLTM and Negotiate authentication types.

   - *None* – No username / password used for proxy authentication.

   - *Current User* – Current username and password provided by windows integrated authentication is used. Note: Only NTLM and Negotiate authentication is supported.

   - *Custom Credentials* – Custom username and password.

- Do not use proxy server for addresses beginning with

- Defines list of addresses (separated with semicolon or new lines) for which proxy won't be used.

● Bypass proxy server for local addresses.

- If checked, proxy is not used when accessing local addresses (IPv4 and IPv6 loopback and current machine hostname).

⚒Configuration of Windows Firewall and HTTP Settings

The Windows Firewall and HTTP settings may require configuration if a user changes the HTTP(S) agent port, with the exception of HTTPS Proxy Agent.

The user may need to configure:

● HTTP settings (if UAC is enabled):

- Allow the new agent port using the provided configuration tool or manually from the command line.

● Windows Firewall (if Windows Firewall is enabled)

- Add Windows Firewall inbound rule for the new agent port using the provided configuration tool or manually from the command line.

HTTP and Windows Firewall settings for the default HTTP(S) agent ports are configured during the initial installation.

Caution: The port selected for the HTTP(S) agent must not be used by any other application and it must not be blocked by a firewall.

If UAC is enabled or the user is not in a local administrator role, the user must obtain permission to listen on the port. To assign such permission, use the command line interface with elevated privileges (such as administrator) and:

1. Use the provided tool `configureHttpAgent.bat.` See HTTP Port Configuration in the installation guide for command line options.

2. Or manually run the following command (the following example is for HTTP port 9000 granting permission to all users on the current machine):

   a. Windows Server 2008 and Windows 7:

   ```
   netsh http add urlacl url=http://+:9000/ user=EVERYONE

   listen=yes delegate=no
   ```

   b. Windows XP and Windows Server 2003 (See Prerequisites in the installation guide for httpcfg tool installation):

   ```
   httpcfg set urlacl -u http://+:9000/ -a "D:(A;;GX;;;WD)"
   ```

It is only required to run this command once for each port. The registration remains in the system until it is removed.

If Windows Firewall is enabled the firewall inbound rules must be added to allow HTTP communication from remote host with HP Service Virtualization. To add these exceptions, use the command line interface with elevated privileges (such as administrator) and do one of the following:

- Use the provided tool `configureHttpAgent.bat` to add Windows Firewall exceptions. See HTTP Port Configuration in the installation guide for details.

- Or add them manually (the following example is adding firewall exceptions for HTTP port 9000 for all applications):

***To add HTTP Proxy port exception for port e.g. 9000, run the following commands*:**

```
netsh firewall add portopening TCP 9000 "Port 9000 HTTP Proxy" ENABLE
netsh firewall add portopening UDP 9000 "Port 9000 HTTP Proxy" ENABLE
```

Chapter 7

⬣Security

Service virtualization supports virtualization of secured services using either HTTP transport level security or subset of SOAP message level security.

Security

Security consists of four components:

- Confidentiality – data are encrypted. Only an ultimate recipient can read data.

- Integrity – data are signed. A recipient can validate that data have not been modified during transmissions.

- Authentication – identity of a client is transferred with message.

- Authorization – service validates that an authenticated client can execute required operation.

Service virtualization doesn't deal with authorization. Authorization logic is left for the real service but service virtualization must deal with other parts:

- Confidentiality – service virtualization must be able to decrypt the message passed to virtual service to learn them and it must be able to encrypt messages passed to the real service

- Integrity – service virtualization must be able to validate signatures passed in requests from clients and responses a real service and it must also be able to sign requests passed to a real service and responses passed to clients.

- Authentication – service virtualization doesn't validate incoming clients but in some scenarios it must have access to their credentials (certificates with private keys or user names with passwords) to pass them to a real service.

Transport security

Transport level security is point-to-point. Security is ensured only on the transport level connection between two machines. Transport level integrity and confidentiality is ensured through SSL / TLS (HTTP(s) Proxy agent or the HTTPS Gateway agent) and transport level authentication is ensured with HTTP authentication mechanisms.

Transport authentication in service virtualization is used only in pass through and recording modes to access the real service. It is transparently managed by the HTTP Gateway or HTTP(S) proxy agents. Transport authentication only requires that correctly configured credentials are available in the Credential Store

See

Message security

Message level security is end-to-end. Security is ensured on the message level – security is part of message data which can be passed through many intermediaries (many connections) without revealing unsecured message content.

Message security in service virtualization is used for pass through, recording and simulation modes.

See "How to Set Message Security" on next page

Mixed security

Mixed security uses transport security to ensure confidentiality and integrity and message security to pass client credentials (authentication). This security configuration requires the usage of the HTTPS agent and message security modes with names ending with OverTransport.

How to Set Authentication Credentials

Some services may require client authentication either on either the transport or message level. When virtualizing these services, the application needs to know what client credentials are used to connect to the real service. The only exception is a scenario in which a real service with HTTP transport authentication (Basic, Digest, NTLM) is virtualized through an HTTP(S) proxy agent. In this case, authentication requests are forwarded and the service virtualization doesn't need the credentials in the service's credential store.

Prerequisites:

- User of the virtual service must exist in the local user store of a Windows installation.

To Set Service Security Settings:

1. Access Security Settings:

    In the Virtualization Explorer view, open the Editor view for a virtual service and expand the Security Settings section.

    In the Topology Editor view, virtualize a secured service that does not have any authentication set. For details, see "How to Create Virtual Services in Topologies" on page 57.

    In the Topology Tasks view, virtualize secured services that do not have any authentication set. For details, see "How to Virtualize Services in the Tasks View" on page 54.

2. To modify the authentication credentials for a secured service, click **Edit Credential Store** and do any of the following:

    a. To add user credentials:

        i. Click **Add Identity** to open the **Edit Identity Details** dialog box.

        ii. Input a Username and Password.

        iii. Optional: Select **Show Password** to display the password in Security Settings.

        iv. Optional: **Add Certificate**

        v. Click **OK** to add the credentials to the Security Settings dialog box.

    b. To edit user credentials:

      i.   Select the user to edit, and click **Edit Identity** to open the **Edit Identity** dialog box.

      ii.   Input a Username and Password.

      iii.   Optional: Select **Show Password** to display the password in Security Settings.

      iv.   Click **OK** to change the credentials in the **Security Settings** dialog box.

  c. To remove user credentials:

      i.   Select the user to remove, click **Remove Identity**, and confirm your decision.

      ii.   To import credentials:

      iii.   Click **Import** to open the **Import Identities** dialog

      iv.   Select the virtual service which contains identity you want to import

      v.   Select the identity you want to import

      vi.   Click **OK** to import the identity from selected virtual service

  d. Save the Virtual Service.

🔧How to Set Message Security

If the service is secured with message security, use the following security settings:

Basic Message Security Basic Settings

- Mode – Predefined supported message security mode

- Real Service Identity – The identity of the real service (certificate) is stored in the Credential Store. This setting is used if a service uses a certificate for message security. The public key of the certificate is used to encrypt messages sent to the real service.

- Virtual Service Identity – The identity of the virtual service (certificate) is stored in the Credential Store. The purpose of this configuration is the same as in Real Service Identity. If the Real Service Identity contains the certificate with a private key this setting doesn't need to be configured. In this scenario, the Real Service Identity is also used as the identity of the virtual service. If the Real Service Identity contains only a certificate without a private key, this setting must be configured to provide the identity of the virtual service. The configured identity must contain a certificate with a private key as the service requires the private key to decrypt the messages coming from the client. Clients must trust the certificate used as the identity of the virtual service.

Advanced Message Security Settings

- Protection Level – Describes the level of security applied to each message. This configuration has service scope levels . All messages must have the same security requirements.

- Message Protection Order – Describes the order of protection operations used to secure messages

- Message Security Version – Describes a set of WS-* specifications used to establish security

- Require Derived Keys – Defines if supporting tokens must use derived keys

- Include Timestamp – Defines if messages must contain security timestamp

- Allow Serialized Token on Reply – Defines if replies can contain a service token used to sign the message. This setting is only used for asymmetric security bindings.

Configuring CertificateOverTransport

Certificate over transport mode uses an endorsing supporting binary token over HTTPS:

- **Transport security binding**

  - Algorithm suite: Basic256

  - Layout: Strict

- **Endorsing supporting token**

  - X509Token (WssX509V3Token10) included always to recipient.

  - Inclusion type: MustSupportRefThumbprint / RequireThumbprintReference

Prerequisites:

• Create a Virtual Service. See "How to Create a Virtual Service" on page 14

• Credential Store must contain an identity with each used client certificate.

• Certificates must contain a private key.

To configure CertificateOverTransport:

1. Check **Enabled** in **Message Security**.

2. Select **CertificateOverTransport** in **Mode**.

- Do not configure **Real Service Identity** or **Virtual Service Identity**.

Advanced settings for CertificateOverTransport:

- *Protection Level* – this setting has no effect because encryption and signing is provided by the transport level (HTTPS)

- *Message Protection Order* – this setting has no effect because encryption and signing is provided by the transport level (HTTPS).

- *Message Security Version* –only WS-Security 1.1 is supported because this configuration mode requires thumbprint token inclusion mode which is not supported in WS-Security 1.0.

- *Require Derived Keys* – this setting should not be changed.

- *Include Timestamp* – this setting must be checked because the endorsing supporting token passed in the request must sign the timestamp header.

- *Allow Serialized Signing Token on Reply* – this setting has no effect.

Configuring UserNameOverTransport

User name over transport mode uses a signed supporting user name token over HTTPS:

- **Transport security binding**

  - Algorithm suite: Basic256

  - Layout depends on WS-Security version configured in Advanced settings:

    - WS-Security 1.0

      - Lax

    - WS-Security 1.1

      - Strict

- **Endorsing supporting token**

  - UserNameToken (WssUsernameToken10) included always to recipient.

    - Only PasswordText token type is supported.

Prerequisites:

- Create a Virtual Service. See "How to Create a Virtual Service" on page 14

- Credential Store must contain an identity with each user and password used to authenticate to the real service.

To configure UserNameOverTransport:

1. Check **Enabled** in **Message Security**.

2. Select **UserNameOverTransport** in **Mode**.

- Do not configure **Real Service Identity** or **Virtual Service Identity**.

Advanced settings for UserNameOverTransport:

- *Protection Level* – this setting has no effect because encryption and signing is provided by the transport level (HTTPS).

- *Message Protection Order* – this setting has no effect because encryption and signing is provided by the transport level (HTTPS).

- *Message Security Version* – layout used for security header.

  - Message security versions using WS-Security 1.0 use Lax layout for security header.

  - Message security versions using WS-Security 1.1 use Strict layout for security header.

- *Require Derived Keys* – this setting has no effect.

- *Include Timestamp* – this setting controls if requests and responses must contain a security timestamp.

- *Allow Serialized Signing Token on Reply* – this setting has no effect.

Configuring MutualCertificate

MutualCertificate is a mode with asymmetric security binding (WS-Security 1.0) which uses both client and server certificates to secure messages over unsecured transport (HTTP):

- Asymmetric security binding

  - Initiator token: X509Token (WssX509V3Token10) included always to recipient.

  - Recipient token: X509Token (WssX509V3Token10) never included.

  - Algorithm suite: Basic256

  - Layout: Strict

  - Token inclusion type:

    - MustSupportRefKeyIdentifier

    - MustSupportRefIssueSerial

Prerequisites:

- Create a Virtual Service. See "How to Create a Virtual Service" on page 14

- Credential Store must contain an identity with a real service certificate.

  - If the certificate doesn't contain a private key, Credential Store must also contain an identity for the virtual service with a certificate containing a private key.

- Credential Store must contain an identity with each used client certificate.

- Client certificates must contain a private key.

To configure MutualCertificate:

1. Check **Enabled** in **Message Security**.

2. Select **MutualCertificate** in **Mode**.

3. Select an identity configured in **Credential Store** for **Real Service Identity**.

4. If the identity for the real service doesn't contain a certificate with a private key or if you want to use separate identity for the virtual service select an identity configured in Credential Store for Virtual Service Identity. This identity must contain a certificate with a private key.

Advanced settings for MutualCertificate:

- *Protection Level* – configures the level of security applied on each message.

- *Message Protection Order* – configures the order of protection operations used to secure messages.

- *Message Security Version* – use only WS-Security 1.0.

- *Require Derived Keys* – this setting should not be changed.

- *Include Timestamp* – this setting controls if requests and responses must contain a security timestamp.

- *Allow Serialized Signing Token on Reply* – this setting has no effect.

Configuring MutualCertificateDuplex

MutualCertificateDuplex mode with asymmetric security binding (WS-Security 1.0 and 1.1) uses both client and server certificates to secure messages over unsecured transport (HTTP). The

difference between MutualCertificate and MutualCertificateDuplex is that this security mode also sends the recipient's signing token back to the initiator.

- Asymmetric security binding

  - Initiator token: X509Token (WssX509V3Token10) included always to recipient.

  - Recipient token: X509Token (WssX509V3Token10) included always to initiator.

  - Algorithm suite: Basic256

  - Layout: Strict

  - Token inclusion type depends on WS-Security version configured in Advanced settings:

    - WS-Security 1.0

      - MustSupportRefKeyIdentifier

      - MustSupportRefIssueSerial

    - WS-Security 1.1

      - MustSupportRefThumbprint / RequireThumbprintReference

Prerequisites:

- Create a Virtual Service. See "How to Create a Virtual Service" on page 14

- Credential Store must contain an identity with a real service certificate.

  - If the certificate contain doesn't contain a private key, the Credential Store must also contain an identity for the virtual service with a certificate containing a private key.

- The Credential Store must contain an identity with each used client certificate.

- Client certificates must contain private key.

To configure MutualCertificateDuplex:

1. Check **Enabled in Message Security**.

2. Select **MutualCertificateDuplex** in **Mode**.

3. Select an identity configured in **Credential Store for Real Service Identity**.

4. If the identity for the real service doesn't contain a certificate with a private key or if you want to use separate identity for the virtual service select an identity configured in **Credential Store for Virtual Service Identity**. This identity must contain a certificate with a private key.

Advanced settings for MutualCertificateDuplex:

- *Protection Level* – configures the level of security applied on each message.

- *Message Protection Order* – configures the order of protection operations used to secure messages.

- *Message Security Version* – this setting defines how the binary token is referenced in the request message.

- Message security versions using WS-Security 1.0 requires either issuer serial number or key identifier of the certificate.

- Message security versions using WS-Security 1.1 requires thumbprint of the certificate.

- *Require Derived Keys* – this setting should not be changed.

- *Include Timestamp* – this setting controls if requests and responses must contain a security timestamp.

- *Allow Serialized Signing Token on Reply* – this setting must be checked because the recipient's signing token is always send back to an initiator.

⚒How to Set Transport Security

Transport level security is completely handled by the HTTP based agent. The virtual and real services can use HTTP authentication to prevent unauthorized use. The service can use basic, digest, NTLM authentication or Mutual HTTPS.

HTTPS and mutual authentication

Services secured with HTTPS are supported by either the HTTPS Gateway agent or the HTTP(S) Proxy agent. Both agent types also support Mutual HTTPS where the client authenticates itself with a client certificate. To use Mutual HTTPS, the credential store must contain a client certificate and its private key for every client accessing the virtual service. See How to Set Authentication Credentials.

HTTP authentication over HTTP(S) Proxy agent

When using a proxy agent and running the virtual service in pass through (stand-by) or recording mode, the authentication is fully transparent and the virtual service doesn't need any further configuration. The entire security handshake is passed from the client to the real service through the proxy and client credentials are validated just by the real service.

HTTP authentication is not used when the service is in simulating mode.

HTTP authentication over HTTP / HTTPS Gateway agent

When using the gateway agent and running the virtual service in pass through (stand-by) or recording mode, the client authenticates to the virtual service and the virtual service authenticates to the real service. The virtual service must be able to validate a client's credentials and pass them to the real service, meaning that the service must have all user names and passwords in the credential store.

There are several steps to set this authentication:

1. All users who are authenticating to your service must be present in the Windows system where the virtual service is running. They can be added as local users of the machine or added to the domain to which the computer belongs. The username and password must be the same as the one the client uses to authenticate to the real service.

    Note: HTTP digest authentication only works with domain users, not local ones. The domain must have reversibly encrypted passwords. See IIS documentation for details.

2. To delegate requests to the real service (when learning or in stand-by mode), the username and

password must be in the service's credential store.

a. From the Virtual Service Editor, open the Security Settings and click **Edit Credential Store**.

b. Click **Add Identity**.

c. Enter Identity details and supply a certificate if required.

d. Click **OK** to add the identity and **OK** again to close the Credential Store.

Note: When using HTTP Basic authentication, credentials missing in the credential store are automatically detected and can be simply added via the Fix It command in the Problem List.

HTTP authentication is not used when the service is in simulating mode.

Windows accounts for HTTP authentication

Basic, digest and NTLM authentication in the HTTP / HTTPS Gateway agent is supported only with Windows accounts:

1. If the computer running Service Virtualization is in the same domain as the service host, make sure the domain users are able to log on to the machine the application is running on. The virtual service authenticates against the same users as the real one.

2. If machines cannot be placed in the same domain, create local windows or domain user accounts (domain users still need to be able to log on the machine the application is running on) with the same names that the client is using to authenticate to the service.

Note: If you would like to use the HTTP Digest authentication, use only domain user accounts as local user accounts will not authenticate.

Chapter 8

HP Test Automation Tools Integration

Service Virtualization can be integrated with HP test automation tools. The virtual services are managed via the tests and the performance monitors exposed by the virtual services are used by the performance testing tools.

Virtual services must be deployed to the Service Virtualization Server for this integration. For details, see "HP Service Virtualization Server" in the installation guide.
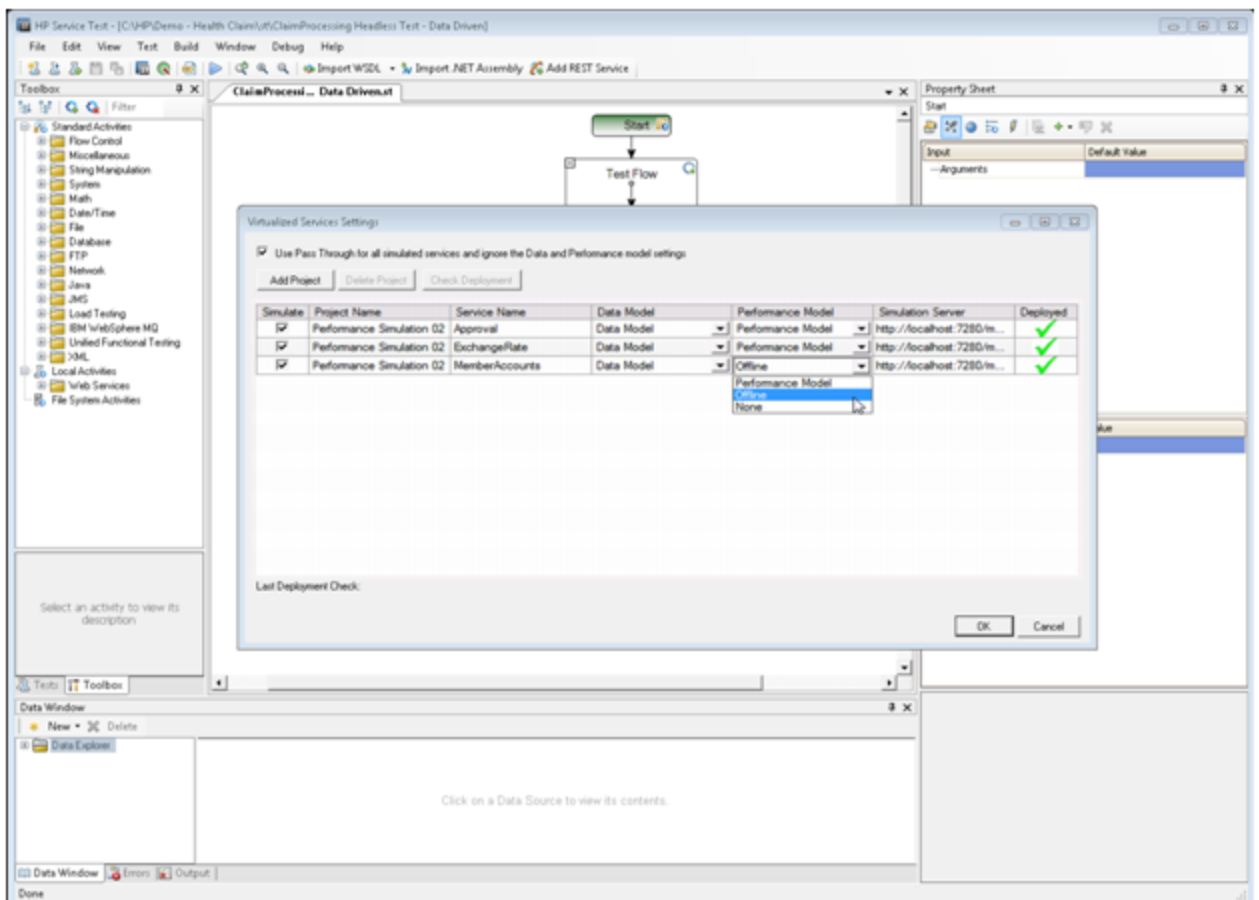
Service Virtualization integrates with Service Test 11.20, Load Runner 11.00 and Performance Center 11 with latest patches.

Service Test

Service virtualization projects can be integrated with HP Service Test (see HP Service Test documentation). The virtual services are then managed by the test after the integration.

- The simulation start is triggered by the test start.

- Particular data and performance models can be selected for the test.

- *Simulating* or *Standby* modes using the real service are chosen during the test.

Performance Center and Load Runner

Service virtualization projects can be integrated with the Performance Center or Load Runner Scenarios (see Performance Center or Load Runner Documentation). The virtual services are then managed by the test after the integration.
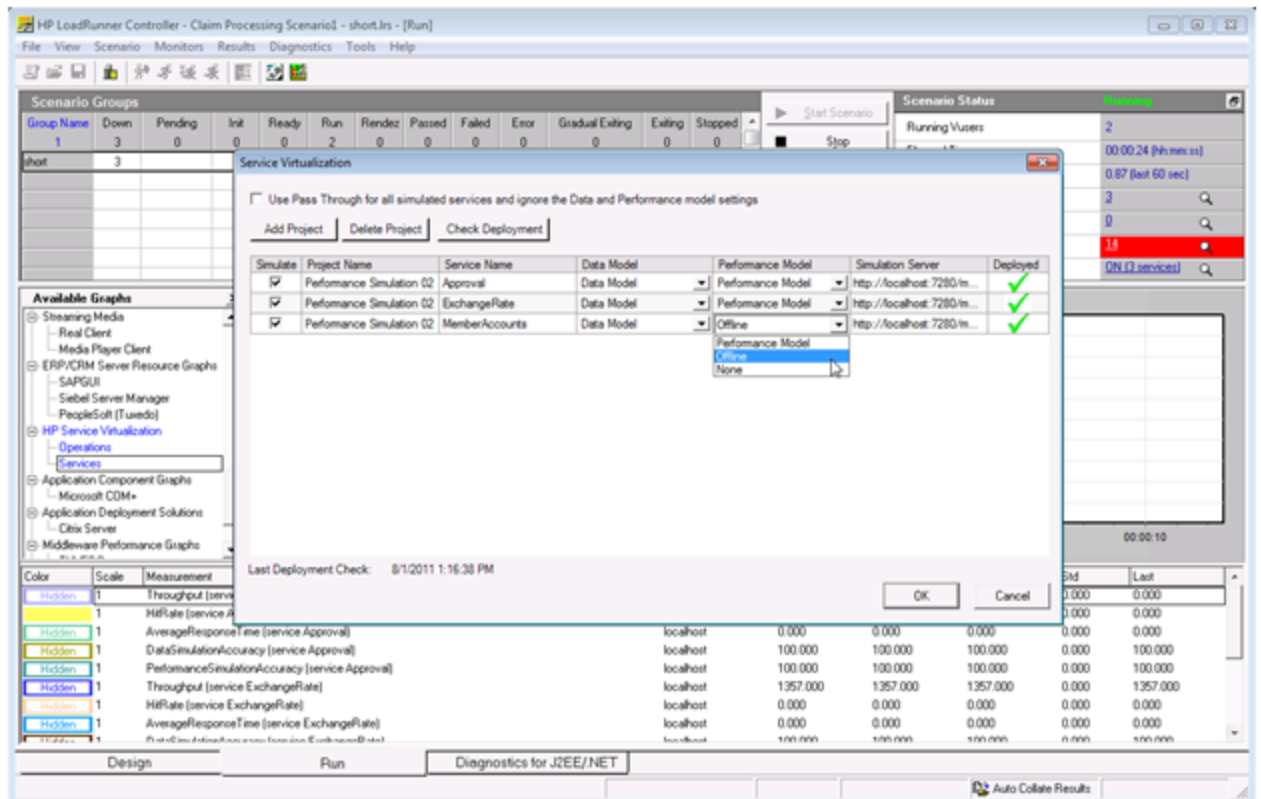
- The simulation start is triggered by the test start.

- Specific data and performance models can be selected for the test.

- *Simulating* or *Standby* modes using the real service are selected during the test.

Performance monitors exposed by virtual services are used immediately in the Performance Center or Load Runner Controller:
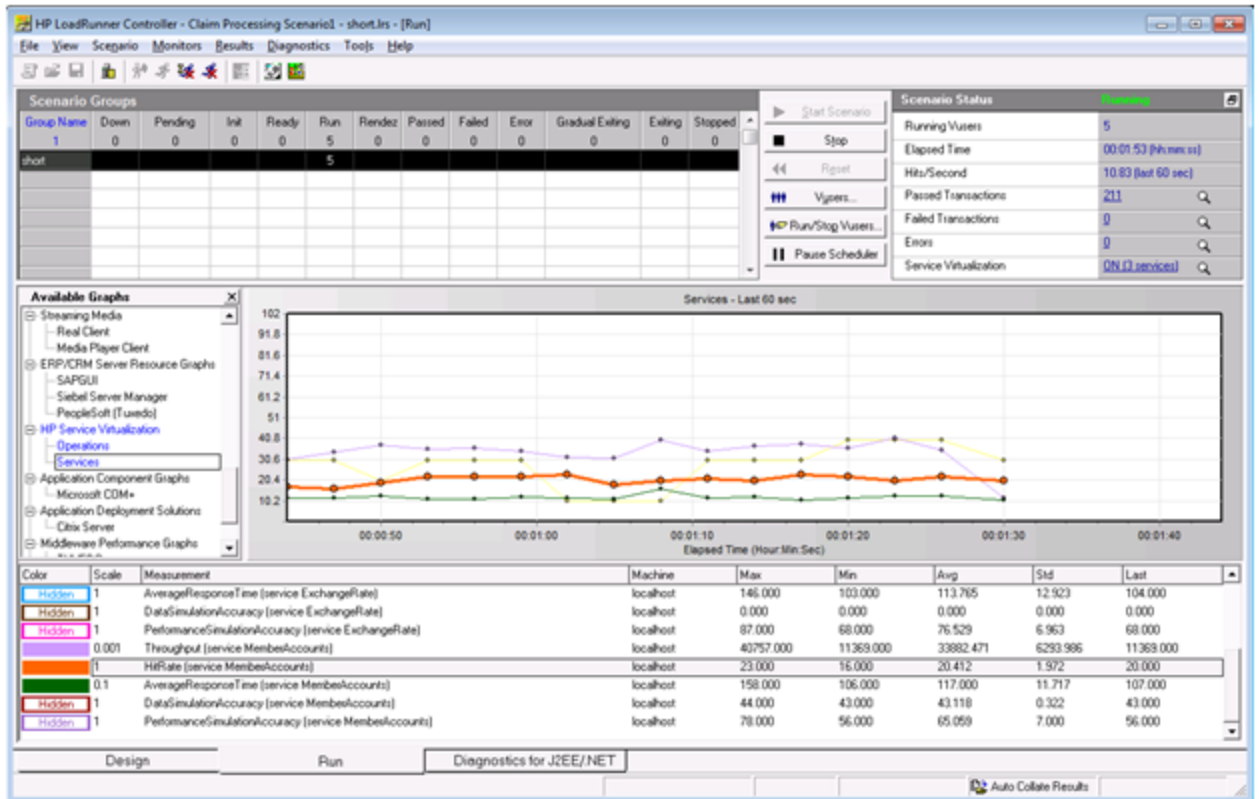
The performance monitors exposed by Service Virtualization are named **Services** and **Operations**.

- The **Services** performance counter provides measurement data per virtual service.

- The **Operations** performance counter provides data per virtual service and service operation.

Troubleshooting

Runtime View

Runtime view doesn't respect selected data model when changing mode.

Problem: User chooses a simulation model in the Service Editor and starts a new learning/simulation from the Runtime View. Instead of using the new simulation, the simulation model from the previous learning/simulation is used again.

Solution: Runtime View is used just to change service modes, not configurations. To change the simulation model, open the Service Editor, select a new simulation model and start a new learning/simulation from the Service Editor.

Cannot start learning service 'My Service'.

Problem: Real Data model or Performance model must be selected in service configuration.

Solution: This error may happen when all data and performance models are deleted from the service configuration and the user tries to start a new learning/simulation session from the Runtime View. To solve the problem, create a new Data/Performance Model and start a new learning/simulation session from the Service Editor.

HTTPS Client Connection Aborted

Problem: The client connection to a virtualized service deployed on an HTTPS endpoint is aborted with the error message `SSL_ERROR_RX_RECORD_TOO_LONG` when Service Virtualization is running on a Windows XP or Windows 2003 machine. The client is normally able to connect to a real service without any issue.

Solution: See http://support.microsoft.com/default.aspx?scid=kb;EN-US;933430 for a list of possible solutions. For Windows 2003, any workaround method as described in the Knowledge Base article can be used, however, only the first or second methods work with Windows XP.

Configuring HTTP Proxy on Clients

Problem: A virtual service is created on a proxy agent and the user is unable to record messages.

Solution:Configure the HTTP Proxy on clients.

All examples below of specific client configurations are using the proxy server *HTTP Proxy Agent* listening on address `hostname` with port 6071.

HTTP Proxy in .Net Client

The .Net client can be configured to use a default proxy server or a specific proxy server.

If using the default proxy server take *HTTP Proxy Agent* settings to configure the default proxy server. This is done in MS Windows or Internet Explorer in **Internet Properties > Connections > LAN settings > Proxy server**. The client then must be configured to use the default proxy. This is set in application configuration file either for application in element `<defaultProxy>`:

```
<configuration>

    <system.net>

      <defaultProxy enabled="true">

        <proxy usesystemdefault="true"/>

      </defaultProxy>

    </system.net>

</configuration>
```

, or for a specific binding in a binding element:

```
<configuration>

    <system.serviceModel>

        <bindings>

            <basicHttpBinding>

              <binding name="myHttpBinding" bypassProxyOnLocal="false"
useDefaultWebProxy="true">

              </binding>

            </basicHttpBinding>

        </bindings>

    </system.serviceModel>

</configuration>
```

The same configuration file can be used to set a specific proxy server. This is an example of client configuration for application:

```
<configuration>

    <system.net>

      <defaultProxy enabled="true">

        <proxy proxyaddress="http://hostname:6071"/>

      </defaultProxy>

    </system.net>

</configuration>
```

, or for specific binding:

```
<configuration>

    <system.serviceModel>
```

```
    <bindings>

      <basicHttpBinding>

        <binding name="myHttpBinding" bypassProxyOnLocal="false"
useDefaultWebProxy="false" proxyAddress=" http://hostname:6071">

        </binding>

      </basicHttpBinding>

    </bindings>

  </system.serviceModel>

</configuration>
```

HTTP Proxy in Java Client

The proxy settings for Java client are given to the JVM via command line arguments. This is example of how to run client from command line with proxy configuration:

```
java -Dhttp.proxyHost=hostname -Dhttp.proxyPort=6071 MyJavaClient
```

HTTP Proxy in WebLogic

Add the Java proxy parameters to Java options in `JAVA_OPTIONS` environment variable in the proper section of script `%WL_HOME%\common\bin\commEnv.cmd` for MS Windows or in `${WL_HOME}/common/bin/commEnv.sh` for Unix/Linux. This is example of setting proxy configuration in file `commEnv.cmd` (MS Windows):

```
set JAVA_OPTIONS=%JAVA_OPTIONS% -Dhttp.proxyHost=hostname -
Dhttp.proxyPort=6071
```

, or in file `commEnv.sh` (Unix/Linux):

```
JAVA_OPTIONS="${JAVA_OPTIONS} -Dhttp.proxyHost=hostname -
Dhttp.proxyPort=6071
```

HTTP Proxy in WebSphere

The HTTP proxy on the WebSphere application server can be configured via setting transport properties `http.proxyHost` and `http.proxyPort`. These HTTP transport properties can be set via:

1. Using wsadmin.

2. Using an assembly tool.

3. Using the JVM custom property panel in the administrative console.

To learn more about (1) and (2), see the *Configuring additional HTTP transport properties documentation* in WebSphere. To configure the HTTP proxy properties using (3) use the administrative console with the following steps:

1. Open the administrative console.

2. Click **Servers > Application Servers > server > Java and Process Management > Process definition > Java Virtual Machine > Custom Properties**.

3. (Optional) If the property is not listed, create a new property name.

4. Enter the name `http.proxyHost` and value `hostname`.

5. Enter the name `http.proxyPort` and value `6071`.

6. Restart the server.

HTTP Proxy in JBoss

Add the Java proxy parameters to Java options in `JAVA_OPTS` environment variable in the startup script `%JBOSS_HOME%\bin\run.bat` or `run.conf.bat` for MS Windows or in `${JBOSS_HOME}/bin/run.sh` or `run.conf` for Unix/Linux. This is an example of setting the proxy configuration in `JAVA_OPTS` environment variable in file `run.conf.bat` (MS Windows):

```
set "JAVA_OPTS=-Dhttp.proxyHost=hostname -Dhttp.proxyPort=6071
```

, or in file run.conf (Unix/Linux):

```
JAVA_OPTS="-Dhttp.proxyHost=hostname -Dhttp.proxyPort=6071
```

Setting HTTP Proxy in Designer

Problem:User is unable to access any remote WSDL or Service Virtualization Server.

Solution:Proxy setting needs to be configured in Designer.

How to Set HTTP Proxy in Designer

In some cases Service Virtualization Designer communicates with external services using the HTTP protocol. The first case is the communication with Service Virtualization Server, where the service is the server management API. The second case is the import of the a real service WSDL. In some situations, the Designer's HTTP communication must be forwarded through an external HTTP Proxy. In that situation, the HTTP Proxy settings must be placed in the Designer's configuration file.

Note: The setting of HTTP Proxy for agents is not done in the Designer configuration file. See "Forwarding HTTP(S) Gateway/Proxy Agent Communication through HTTP Proxy" on page 71 for details.

Setting HTTP Proxy in the Designer configuration file

In order to use an external HTTP Proxy for the Designer HTTP communication with a server and the import of a WSDL from real services, the Designer configuration file must be modified. This file is located at `%[INSTALLLOCATION]%\Designer\bin\VirtualServiceDesigner.exe.config`.

In the configuration file the element `<defaultProxy>` holds the HTTP Proxy configuration. This element is located in the document in elements `<configuration><system.net>`. By default, HTTP Proxy is disabled by `<defaultProxy enabled="false"/>`.

The Designer can be configured to use the system HTTP Proxy or to use a specific HTTP Proxy. Detailed documentation can be found at http://msdn.microsoft.com/library/kd3cf2ex.aspx. The Designer must be restarted in order to apply the changes in the configuration file.

System HTTP Proxy

If using the system HTTP Proxy, ensure that the HTTP Proxy is configured in the system. The settings are available in Windows Internet Explorer® menu **Internet Properties > Connections > LAN settings > Proxy server**. The Designer must be configured to use the same proxy in the configuration file in the element **<defaultProxy>** like this:

```
<configuration>

   <system.net>

      <defaultProxy enabled="true">

        <proxy usesystemdefault="true"/>

      </defaultProxy>

   </system.net>

</configuration>
```

Specific HTTP Proxy

If using a specific HTTP Proxy other than the system one, follow this example of Designer configuration:

```
<configuration>

   <system.net>

      <defaultProxy enabled="true">

        <proxy proxyaddress="http://foo.com:8080"/>

      </defaultProxy>

   </system.net>

</configuration>
```

Setting Credentials for Authenticated HTTP Proxy

If the Designer is configured to use HTTP Proxy with authentication, some additional modifications may be required in the Designer configuration file. This will allow the supplying of credentials for the HTTP Proxy. These modifications are required in the case of HTTP communication with the server management API. In the case of WSDL imports, these modifications are only optional as the Designer will prompt the user for credentials if needed.

In order to set credentials for authenticated HTTP Proxy in the Designer, this section must be enabled in the configuration file `VirtualServiceDesigner.exe.config`:

```
<configuration>

   <system.net>

      <defaultProxy enabled="true" useDefaultCredentials="false">

        <module type =
"HP.SOAQ.ServiceVirtualization.ServerManagementClient.Remote.AuthenticatedProxy,
 HP.SV.ServerManagementClient" />

      </defaultProxy>

   </system.net>
```

```
</configuration>
```

The credentials for the authenticated HTTP Proxy are set in application keys section. Example of credential settings follows:

```
<configuration>

   <appSettings>

      <add key="proxyUserName" value="user1" />

      <add key="proxyPassword" value="pass1" />

      <add key="proxyAddress" value="http://foo.com:8080" />

   </appSettings>

</configuration>
```

Where `key`s are defined as:

proxyUserName	User name for authenticated proxy credentials
proxyPassword	Password for authenticated proxy credentials
proxyAddress	Address of authenticated proxy (e.g. `http://foo.com:8080`). If this value is empty then the Proxy Server address in the system proxy is used.

Once the credentials and the proxy settings are configured as described above then the Designer will use them both for HTTP communication with the server management API and WSDL imports from real services.

Chapter 9

⚒ Service Virtualization Demos

Service Virtualization application demos are installed as an option during the installation process. The following six demos are included:

Claim Processing Demo

The simulation of a backend SOAP service with limited accessibility in a simple composite application. The service can optionally use HTTP authentication (see demo readme)

Claim Processing REST Demo

This demo shows a composite application consisting of 2 REST services using JSON and XML data formats.

Claim Processing Faults Demo

The simulation of a backend SOAP service with limited accessibility in a simple composite application. The simulated service returns either regular response or one of 3 different SOAP faults.

Claim Processing Security Demo

This demo shows a composite application consisting of 2 SOAP services. It allows demonstrating how to record and simulate the behavior of one of the SOAP services. Both services authenticate each other using X509 certificates.

Claim Processing Standalone Server Demo

This demo shows a composite application consisting of 2 SOAP services. It allows demonstrating how to record and simulate the behavior of one of the SOAP services on a standalone server.

Claim Approval JMS Demo

This demo shows a composite application consisting of 3 SOAP services. It allows demonstrating how to record and simulate the behavior of two SOAP services. Demo is similar to the Claim processing service simulation demo. One XML over JMS service (TIBCO EMS approval service) has been added to the topology here and is being simulated, too.

Claim Approval WebSphere MQ Demo

This demo shows a composite application consisting of 3 SOAP services. It allows demonstrating

how to record and simulate the behavior of two SOAP services. The demo is similar to the Claim

Processing service virtualization demo. One XML over WebSphere MQ service (WebSphere MQ approval service) has been added to the

topology here, and is being simulated, too.

Request Tracking Service Activity Demo

This demo shows a composite application consisting of 4 SOAP services. It allows demonstrating how to record and simulate the behavior of two SOAP services. In addition, activity can be demonstrated by calling the third SOAP service from a simulated service.

ShoppingCart - No Sessions Demo

This demo shows virtualization of stateful shopping cart service, where only one client is using the stateful service.

ShoppingCart - Sessions by Clients Demo

This demo shows virtualization of stateful shopping cart service, where multiple concurrent clients are using the stateful service and private session is generated for each client.

ShoppingCart - Sessions by Orders

This demo shows virtualization of stateful shopping cart service, where multiple concurrent clients are using the stateful service and sessions are generated per each shopping order. The checkout operation finishing shopping order destroys the client session (the next operation creates a new one).