

1. LISA Pathfinder Guide	2
1.1 Getting Started with Pathfinder	2
1.1.1 Pathfinder Prerequisites	2
1.1.2 Creating Transactions for Pathfinder	2
1.1.3 Opening the Pathfinder Console	3
1.1.4 Pathfinder Console Layout	3
1.2 Pathfinder Base Features	4
1.2.1 Filtering Paths	4
1.2.2 Pathfinder Console - Paths Tab	5
1.2.3 Pathfinder Console - Component Details Tab	5
1.2.4 Pathfinder Console - Statistics Tab	8
1.2.5 Path Graph	9
1.2.6 Exporting and Importing Paths	11
1.2.7 Viewing Exception Data	11
1.2.8 Viewing Agent Information	12
1.2.9 Stopping and Starting Agent Transaction Capture	12
1.2.10 Integrating Pathfinder Broker and Portal Server	13
1.2.11 Disabling Pathfinder	13
1.3 Creating Cases	13
1.3.1 Email Settings for New Cases	14
1.3.2 Capturing Cases in an Application	14
1.3.3 Pathfinder Console - Cases Tab	15
1.3.4 Searching Cases	16
1.3.5 Editing Case Information	16
1.3.6 Obtaining the Direct URL of a Case	17
1.3.7 Viewing the Transactions for a Case	17
1.4 Creating Data Sets	17
1.5 Creating Baseline Test Cases and Suites	18
1.5.1 Creating EJB Baselines	18
1.5.1.1 Generating the EJB Baseline Assets	18
1.5.1.2 Importing the EJB Baseline Assets	19
1.5.1.3 Viewing and Running EJB Baseline Test Cases	19
1.5.1.4 Viewing and Running EJB Baseline Suites	20
1.5.2 Creating JMS Baselines	21
1.5.2.1 Generating the JMS Baseline Assets	21
1.5.2.1.1 JMS Baseline Nodes	22
1.5.2.2 Importing the JMS Baseline Assets	22
1.5.2.3 Viewing and Running JMS Baseline Test Cases	23
1.5.2.4 Viewing and Running JMS Baseline Suites	24
1.5.3 Creating REST Baselines	26
1.5.3.1 Generating the REST Baseline Assets	26
1.5.3.2 Importing the REST Baseline Assets	26
1.5.3.3 Viewing and Running REST Baseline Test Cases	27
1.5.3.4 Viewing and Running REST Baseline Suites	28
1.5.4 Creating Web Service Baselines	28
1.5.4.1 Generating the Web Service Baseline Assets	29
1.5.4.2 Importing the Web Service Baseline Assets	29
1.5.4.3 Viewing and Running Web Service Baseline Test Cases	29
1.5.4.4 Viewing and Running Web Service Baseline Suites	30
1.5.5 Creating WebSphere MQ Baselines	32
1.5.5.1 Generating the WebSphere MQ Baseline Assets	32
1.5.5.1.1 WebSphere MQ Baseline Scenarios	32
1.5.5.1.2 WebSphere MQ Baseline Drivers	33
1.5.5.1.3 WebSphere MQ Baseline Properties	33
1.5.5.2 Importing the WebSphere MQ Baseline Assets	34
1.5.5.3 Viewing and Running WebSphere MQ Baseline Test Cases	34
1.5.5.4 Viewing and Running WebSphere MQ Baseline Suites	36
1.5.6 Baseline Wizard	37
1.6 Creating Virtual Services from Transactions	41
1.6.1 Creating Virtual Services from JMS Transactions	42
1.6.2 Creating Virtual Services from REST Transactions	43
1.6.3 Creating Virtual Services from Web Service Transactions	43
1.6.4 Creating Virtual Services from WebSphere MQ Transactions	44
1.7 Native MQ Pathfinder	46
1.7.1 Creating the ITKO_PATHFINDER Queue	46
1.7.2 Installing the Native MQ Pathfinder API Exit	46
1.7.3 Configuring the Native MQ Pathfinder API Exit	47
1.7.4 Configuring the Agent Properties for Native MQ Pathfinder	47
1.7.5 Generating Transactions from the ITKO_PATHFINDER Queue	47

LISA Pathfinder Guide

The LISA Pathfinder online documentation consists of the following chapters.

[Getting Started with Pathfinder](#)
[Pathfinder Base Features](#)
[Creating Cases](#)
[Creating Data Sets](#)
[Creating Baseline Test Cases and Suites](#)
[Creating Virtual Services from Transactions](#)
[Native MQ Pathfinder](#)

Pathfinder includes Silk icons created by famfamfam. The author is Mark James and more information can be found on his website at <http://www.famfamfam.com/lab/icons/silk/>.

Getting Started with Pathfinder

For a conceptual overview of Pathfinder, see [LISA Pathfinder](#).

[Pathfinder Prerequisites](#)
[Creating Transactions for Pathfinder](#)
[Opening the Pathfinder Console](#)
[Pathfinder Console Layout](#)

Pathfinder Prerequisites

To run Pathfinder, the LISA Agent is required.

In the LISA Bank application, the Agent is installed by default. For any other application, you will need to [install the Agent](#) and [start the registry](#).

For testing purposes, start the LISA demo server which has the Agent installed or run the LISA Bank application. For running with any other Java application, you will need to configure the Agent. For more information about the Agent, see the [LISA Agent Guide](#).

Pathfinder requires a registry to run. The registry invokes the broker and communicates all transaction-related details to the database. The registry maintains the database of paths that are captured, with other information. The default database used by Pathfinder is located in the **LISA_HOME/database/pathfinder.db** directory.

Pathfinder can be activated only after you have installed the LISA Agent.

In addition, you must do some transactions within the application server, so that they can be seen and traced within the Pathfinder Console.

To use the Pathfinder Console, you must have Flash enabled in the browser.

Creating Transactions for Pathfinder

To use Pathfinder, you must first generate transactions on a computer where the [LISA Agent](#) is running.

You can generate sample transactions for Pathfinder by using the LISA Bank application.

The LISA Bank application needs the Demo Server running.



Some of the tutorials in the [User Guide](#) require you to use the LISA Bank application.

Pathfinder will capture the traffic and generate *paths*, which you can view and manage from the Pathfinder Console.

To create transactions for Pathfinder by using the LISA Bank application

1. Start the Demo Server by running the startup script in the **LISA_HOME/DemoServer/lisa-demo-server** directory.
2. In a web browser, enter <http://localhost:8080/lisabank>. The login page for the LISA Bank application appears.
3. Log in using the following credentials: user name **lisa_simpson** and password **golisa**.
4. Do some transactions within the LISA Bank application. For example:
 - Add a new account
 - Deposit money into the account
 - Withdraw money from the account
5. Log out of the LISA Bank application.

Opening the Pathfinder Console

The Pathfinder Console is a web-based application that lets you view transactions, manage cases, view Agent information, create data sets, create baseline tests, and create virtual services.

You can open the Pathfinder Console from LISA Workstation or from a web browser.

To open the Pathfinder Console from LISA Workstation

- From the main menu, select View > Pathfinder Console.

To open the Pathfinder Console from a web browser

1. Ensure that the [registry](#) is running.
2. Enter <http://localhost:1505/> in a web browser. If the registry is located on a remote computer, replace **localhost** with the name or IP address of the computer.
The LISA Console appears.
3. Click LISA Pathfinder.

Pathfinder Console Layout

The Pathfinder Console is divided into the following areas:

- [Left Panel](#)
- [Right Panel](#)
- [Path Graph](#)
- [Dataset, Baseline, and Virtualize Icons](#)

Left Panel

The left panel includes the following tabs:

- **Paths:** Enables you to restrict the paths that appear in the right panel. For more information, see [Filtering Paths](#).
- **Cases:** Enables you to manage issues that have been captured in an application on an Agent-enabled computer. For more information, see [Creating Cases](#).
- **Agents:** Enables you to view information about Agents in the same subnet. In addition, you can control whether an Agent is capturing transactions. For more information, see [Viewing Agent Information](#) and [Stopping and Starting Agent Transaction Capture](#).

If you do not have a license for creating cases, then the Cases tab does not appear.

Right Panel

The right panel includes the following tabs:

- **Paths:** Lists the available paths. For more information, see [Pathfinder Console - Paths Tab](#).
- **Component Details:** Lists various types of information about a component within a path. For more information, see [Pathfinder Console - Component Details Tab](#).
- **Statistics:** Provides a detailed look into the statistics of the selected transaction, including the Agent name, CPU time, memory, and heap/non heap usage. For more information, see [Pathfinder Console - Statistics Tab](#).

Path Graph

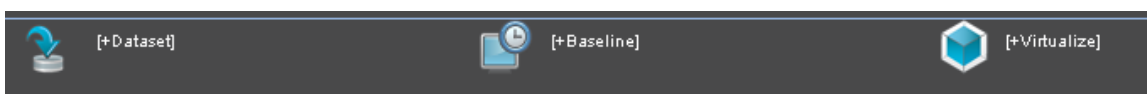
When you select a path in the right panel, the lower portion of the Pathfinder Console displays a *path graph*. The path graph shows the individual components within the path.

For more information, see [Path Graph](#).

Dataset, Baseline, and Virtualize Icons

Depending on your license, one or more of the following icons may appear at the bottom of the Pathfinder Console:

- **Dataset:** Enables you to [create data sets](#).
- **Baseline:** Enables you to [create baseline test cases and suites](#).
- **Virtualize:** Enables you to [create virtual services from transactions](#).



Pathfinder Base Features

The base features of Pathfinder let you view transactions that have been captured by the LISA Agent.

The Pathfinder Console exposes a wealth of information about the transactions and their individual components. This information includes execution times, thread utilization, log messages, SQL statements, request and response data, and stack traces.

You can configure Pathfinder to automatically delete old transactions in the Pathfinder database.

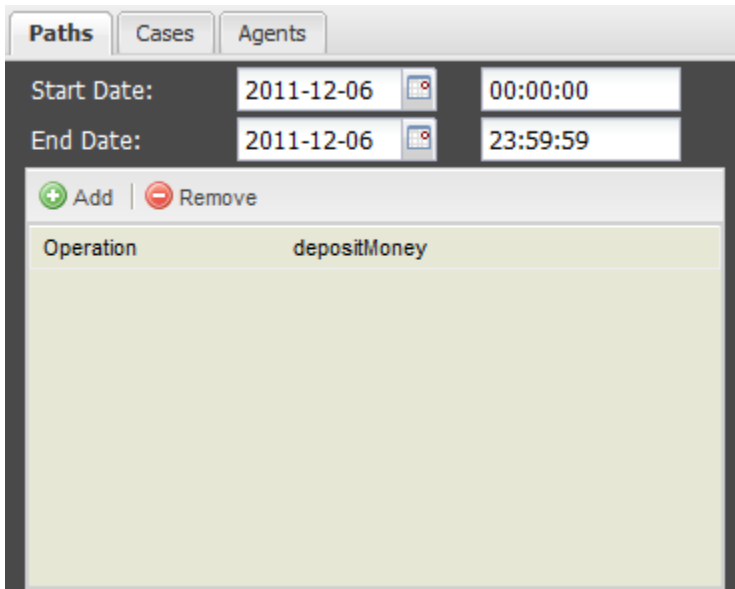
The following topics are available.

- Filtering Paths
- Pathfinder Console - Paths Tab
- Pathfinder Console - Component Details Tab
- Pathfinder Console - Statistics Tab
- Path Graph
- Exporting and Importing Paths
- Viewing Exception Data
- Viewing Agent Information
- Stopping and Starting Agent Transaction Capture
- Automatic Deletion of Transactions
- Integrating Pathfinder Broker and Portal Server
- Disabling Pathfinder

Filtering Paths

You can use the Paths tab in the left panel of the Pathfinder Console to restrict the paths that appear in the right panel.

The following image shows the Paths tab in the left panel. A filter has been added for a specific operation.



You can specify the following date-range criteria:

- Start date and time
- End date and time

By default, the date-range criteria is set to the current day.

You can add one or more of the following filters:

- Agent
- Type: This filter takes a string value that represents the type of node (such as **ejb**, **ibmmq**, **jms**, **rest**, or **ws**). For example, if you add a Type filter and set the subcategory to **ejb**, then the Pathfinder Console displays any path that includes an EJB node.
- Exec Time (ms): This filter takes a numeric value that represents the minimum execution time. For example, if you want to see paths that have an execution time of 500 ms or higher, then enter 500.
- Class Name
- Operation

- Starting IP
- Service IP
- Transaction ID

To filter paths

1. Open the Pathfinder Console.
2. In the left panel, click the Paths tab.
3. (Optional) Change the date-range criteria.
4. (Optional) Add one or more filters.
 - a. Click Add.
 - b. In the left drop-down list, select a category.
 - c. In the right drop-down list, enter a value or select a subcategory.
 - d. Click Save.

Pathfinder Console - Paths Tab

The right panel of the Pathfinder Console includes a Paths tab. Each row in the Paths tab contains information about a transaction that was captured by the LISA Agent.

The following image shows the Paths tab in the right panel. The transactions that appear are based on the LISA Bank application.

Paths							
Component Details							
Statistics							
Time	Category	Starting IP	Service IP	Name	Execution Time	SQL Statements	Elements
2011-12-06 10:45:17 414	http	192.168.16...	192.168.16...	/lisabank/logout.do	301 ms	0	1
2011-12-06 10:45:12 613	http	192.168.16...	192.168.16...	/lisabank/buttonclick.do	3164 ms	1	2
2011-12-06 10:45:08 929	http	192.168.16...	192.168.16...	/lisabank/depositmoney.do	585 ms	11	7
2011-12-06 10:44:45 696	ws	192.168.16...	192.168.16...	/itkoExamples/EJB3AccountControlBean.service	443 ms	0	1
2011-12-06 10:44:45 480	ws	192.168.16...	192.168.16...	/itkoExamples/EJB3AccountControlBean.service	200 ms	0	1
2011-12-06 10:44:45 449	ws	192.168.16...	192.168.16...	/itkoExamples/EJB3AccountControlBean.service	699 ms	0	1
2011-12-06 10:44:45 386	http	192.168.16...	192.168.16...	/lisabank/buttonclick.do	4587 ms	3	4
2011-12-06 10:44:37 649	ws	192.168.16...	192.168.16...	/itkoExamples/EJB3AccountControlBean.service	126 ms	0	1
2011-12-06 10:44:36 931	ws	192.168.16...	192.168.16...	/itkoExamples/EJB3AccountControlBean.service	215 ms	0	1
2011-12-06 10:44:36 354	ws	192.168.16...	192.168.16...	/itkoExamples/EJB3AccountControlBean.service	1665 ms	0	1

The most recent transaction appears at the top.

For each transaction, the following information is displayed:

- **Time:** The date and time when the transaction took place.
- **Category:** The type of transaction (for example, EJB, HTTP, Web Service).
- **Starting IP:** The IP address or hostname where the request came from (that is, the source).
- **Service IP:** The IP address or hostname where the request went to (that is, the target).
- **Name:** The name of the transaction.
- **Execution Time:** The execution time of the transaction (in milliseconds).
- **SQL Statements:** The number of SQL statements executed.
- **Elements:** The number of components in the transaction.

If an exception occurred, then the name appears in red.

When you select a row, the [path graph](#) changes to show details related to the selected transaction.

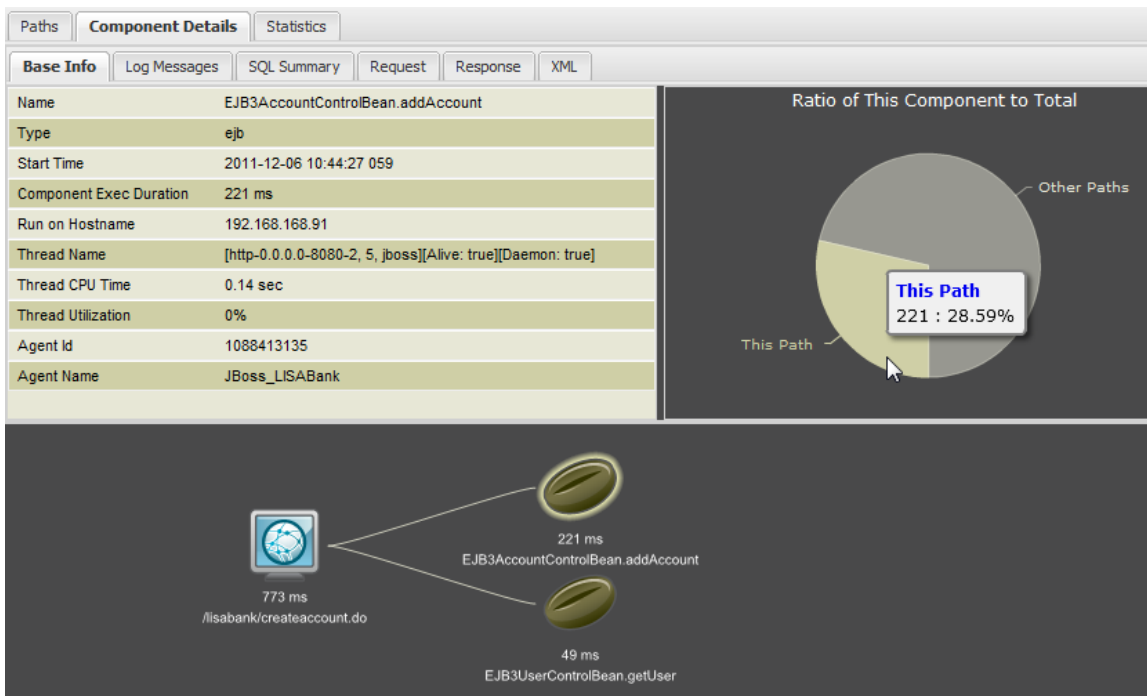
Pathfinder Console - Component Details Tab

When you select a component in the path graph, the Component Details tab lists information about the component.

The Component Details tab has the following subtabs:

- [Base Info Tab](#)
- [Log Messages Tab](#)
- [SQL Summary Tab](#)
- [Request Tab](#)
- [Response Tab](#)
- [XML Tab](#)

The following image shows the Component Details tab. An EJB component is selected in the path graph.



Base Info Tab

Pathfinder shows the basic data of the selected component, from the Java virtual machine (JVM). The following details are listed:

- Component name
- Component type
- Start time
- Execution time of the component
- Name of the host on which the component ran
- Thread name
- Thread CPU time
- Thread utilization
- Agent ID
- Agent name

If the execution time for a component is so low that it rounds down to zero, then the value **0 ms** is displayed.

This tab also includes a pie chart. The pie chart displays the execution time for the component as a percentage of the execution time for the overall transaction.

Log Messages Tab

Pathfinder reports any log messages in this tab. The following details are listed:

- Level of information
- Logger name (if any)
- The text of the log message

Component Details		
Base Info	Log Messages	Statistics
Level	Logger	Message
info	org.apache.struts.action.ActionServlet.service	Web Service end point: http://localhost:8080/itkoExamples/EJB3AccountControlBean?wsdl

To sort the columns, click the drop-down arrow in a column heading and then select Sort Ascending or Sort Descending.

To show or hide columns, click the drop-down arrow in a column heading, point to Columns, and select or clear the check boxes.

SQL Summary Tab

Pathfinder provides a summary of the SQL information in this tab. It displays the relative executions of both the SQL calls and the non database interaction times for this component.

The following details are listed:

- SQL ID
- SQL statement
- Row count
- Invocations
- Average time
- Total time

Paths

Component Details

Statistics

Base Info

Log Messages

SQL Summary

Request

Response

XML

SQL Statements

ID	SQL	Row Count	Invocations	Avg. Time (ms)	Total Time (ms)
[sql id 1]	select account0_account_id as account1_0_0_, account0_name as name0_0_, account0_...	1	1	1	1

Results - [sql id 1]

ACCOUNT1_0_0_	NAME0_0_	TYPE0_0_	VERSION0_0_	AVAILABLE5_0_0_	
18409478675	MySavings	1	1	500.00	

Clicking the SQL statement will display the results.

Double-clicking the SQL statement will open a dialog that displays the entire statement.

To sort the columns, click the drop-down arrow in a column heading and then select Sort Ascending or Sort Descending.

To show or hide columns, click the drop-down arrow in a column heading, point to Columns, and select or clear the check boxes.

Request Tab

Pathfinder displays the actual request that was sent by the selected component to the component on its right.



JMS transactions are different from other transaction types. The payload, whether a request or a response, is always located in the Request tab. This behavior does not affect the generation of data sets, baselines, and virtual services in Pathfinder.

The following image shows an example of the Request tab.

Paths	Component Details	Statistics
Base Info	Log Messages	SQL Summary
Request	Response	XML
<pre><env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/"> <env:Header /> <env:Body> <ns1:getAccount xmlns:ns1="http://ejb3.examples.itko.com/"> <accountId>1781705749</accountId> </ns1:getAccount> </env:Body> </env:Envelope></pre>		

If a WebSphere MQ transaction includes a binary payload, then the Request tab displays a human-readable version of the payload. Non-text characters are removed. You can view the original binary payload in the XML tab.

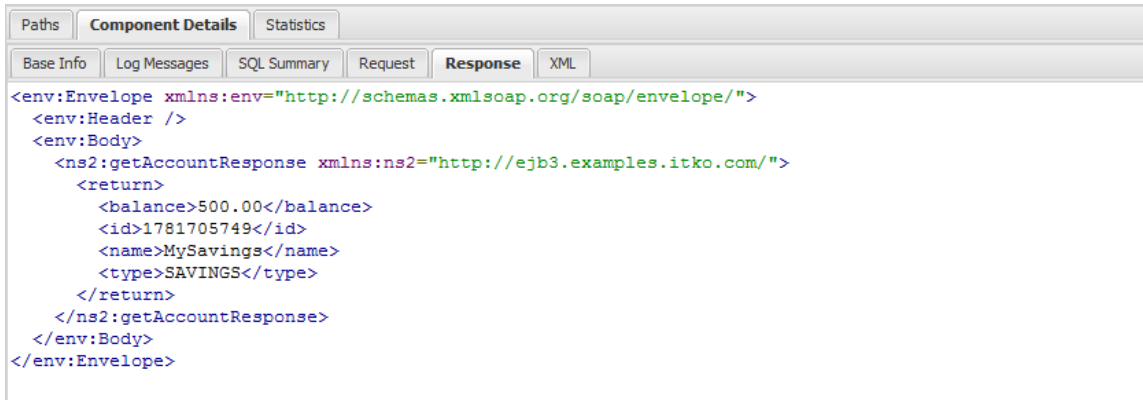
Response Tab

Pathfinder displays the actual response that was received by the highlighted component from the component on its left.



JMS transactions are different from other transaction types. The payload, whether a request or a response, is always located in the Request tab. This behavior does not affect the generation of data sets, baselines, and virtual services in Pathfinder.

The following image shows an example of the Response tab.

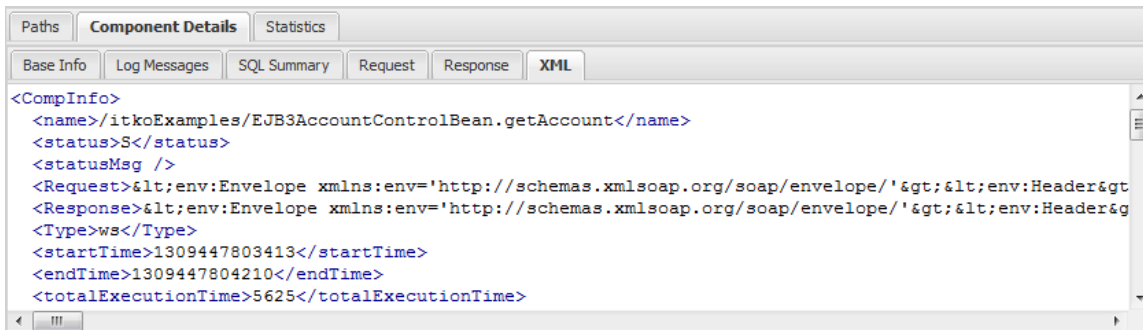


If a WebSphere MQ transaction includes a binary payload, then the Response tab displays a human-readable version of the payload. Non-text characters are removed. You can view the original binary payload in the XML tab.

XML Tab

Pathfinder displays the raw XML of the instrumentation response in the component.

The XML tab is additive. The components to the right contain less information than those on the left. The first component contains the complete response.



Pathfinder Console - Statistics Tab

The Statistics tab in the Pathfinder Console contains VM-level statistics for the window of time in which the selected transaction occurred.

The following statistics are provided:

- **Agent:** The Agent that the frame was generated in.
- **Time:** When the frame was sampled.
- **CPU Usage:** The CPU usage of the VM process hosting the Agent at the time of sampling.
- **Heap Usage:** The Java heap usage of the VM process hosting the Agent at the time of sampling.
- **Non Heap Usage:** The native memory usage of the VM process hosting the Agent at the time of sampling.
- **IO (in):** The number of bytes read by the VM hosting the Agent since the last frame was sampled.
- **IO (out):** The number of bytes written by the VM hosting the Agent since the last frame was sampled.
- **GC Count:** The number of garbage collection events in the VM since the last frame was sampled.
- **GC Time:** The total time spent in garbage collection since the last frame was sampled.

The following image shows the Statistics tab.

Statistics								
Agent	Time	CPU Usage	Heap Usage	Non-Heap Usage	IO (in)	IO (out)	GC Count	GC Time
JBoss_LISABank	2011-12-06 10:44:29	3	212	74	0	0	0	0.000 ms
JBoss_LISABank	2011-12-06 10:44:28	51	211	74	29	179	0	0.000 ms
JBoss_LISABank	2011-12-06 10:44:27	10	193	74	63	44	0	0.000 ms
JBoss_LISABank	2011-12-06 10:44:26	0	190	73	0	0	0	0.000 ms
JBoss_LISABank	2011-12-06 10:44:25	1	190	73	0	0	0	0.000 ms
JBoss_LISABank	2011-12-06 10:44:24	1	189	73	0	0	0	0.000 ms

If multiple Agents are connected, you can choose to select any Agent from the Agents drop-down list.

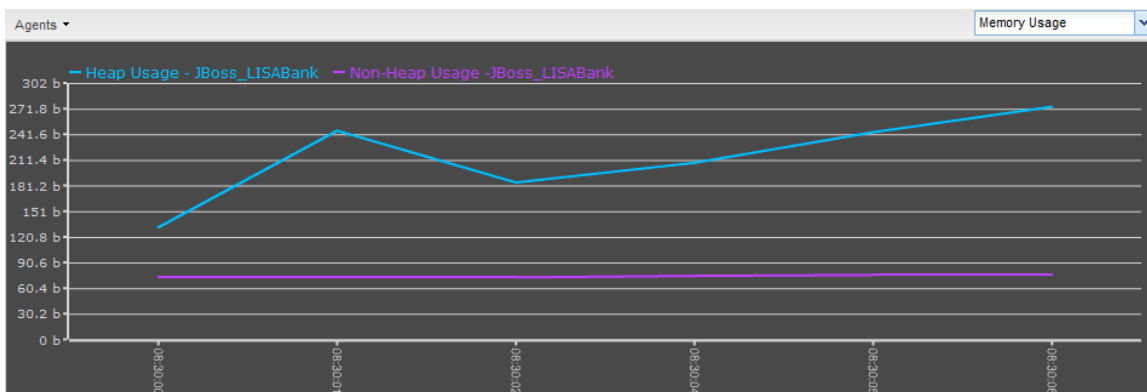
The lower portion of the Pathfinder Console displays a graphical view of the following statistics:

- CPU Usage
- Memory Usage (heap and non heap)
- I/O Usage (in and out)
- Garbage Collection (count and time)

The drop-down list lets you choose among these statistics.

The units on the Y-axis for CPU usage are percentages. The units on the Y-axis for memory usage are bytes. The units on the Y-axis for I/O usage are bytes. The units on the Y-axis for garbage collection time are milliseconds.

The following image shows an example of the memory usage.

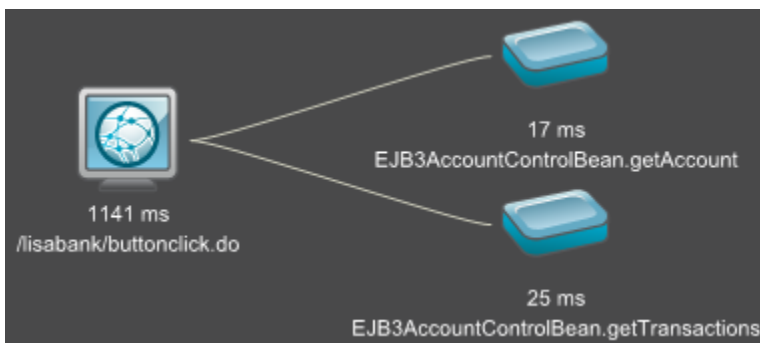


Path Graph

The *path graph* is an area of the Pathfinder Console that contains a visual representation of the components in a path.

When you select a path in the right panel of the Pathfinder Console, the path graph appears.

The following image shows a path graph that has one HTTP component and two web service components.

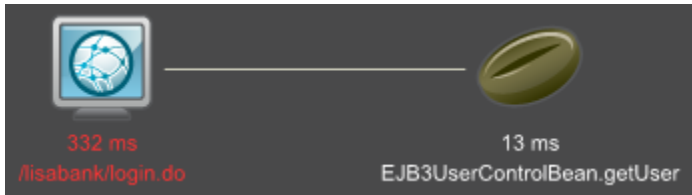


The icons indicate the type of component. Examples of the component types are EJB, HTTP, JMS, REST, web service, LISA, and unknown. The LISA component type represents a step executing inside a test case or virtual service model.

The text below the icon provides the following information:

- Execution time
- Component name

If the path contains an exception, then the appropriate component appears in red. In the following image, the leftmost component has the exception.

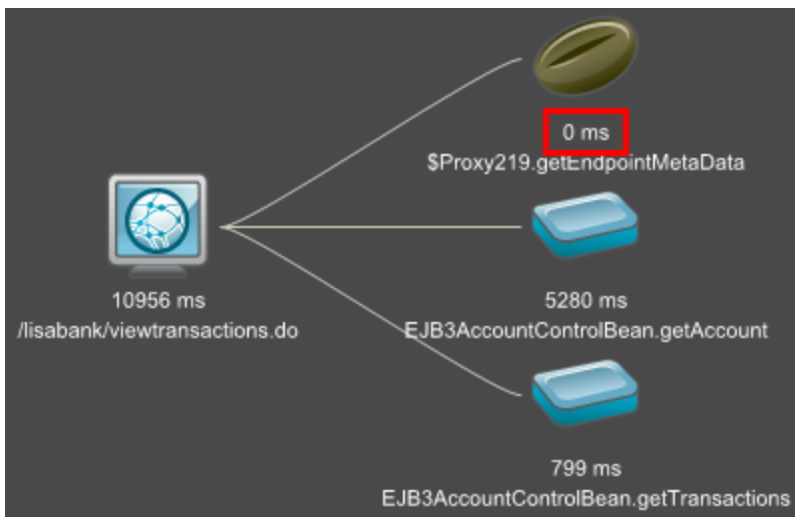


Execution Time

The first line below the component is the execution time.

In certain scenarios, the execution time below the leftmost component is the time for the entire transaction.

If the execution time is so low that it rounds down to zero, then the value **0 ms** is displayed. The following image shows an example of this behavior.



Component Name

The second line below the component is the component name.

The format of the component name depends on the type of component.

The component name for HTTP is the path portion of the URL.

The component name for JMS consists of the following parts:

- A string that identifies the type of operation. If the operation involves the production of a message, then the string is **send:**. If the operation involves the consumption of a message, then the string is **recv:**.
- The name of the queue
- (Optional) The name of the connection factory

An example is **recv:queue/ORDERS.REQUEST@ConnectionFactory**.

The component name for JMS can also be **LISA**. This name is used when a test case receives a JMS message.

The component name for RMI is the Java class and method that was executed.

The component name for webMethods is the flow service name.

The component name for WebSphere MQ consists of the following parts:

- The string **put** or **get**
- The WebSphere MQ host name
- The queue manager name
- The queue name

An example is **put-172.24.255.255-QueueManager-ORDERS.REQUEST**.

The component name for a web service is the path portion of the URL.

Exporting and Importing Paths

The Pathfinder Console lets you export one or more paths. These paths can be imported at a later time.

To export paths

1. In the right panel, select one or more paths that you want to export.
2. Right-click the paths and select Export Selected Paths.
The Path Export Wizard appears.
3. In the Export Name field, enter a name for the export file.
4. Click Next.
5. Click the link and save the export file.
6. Click Finish.

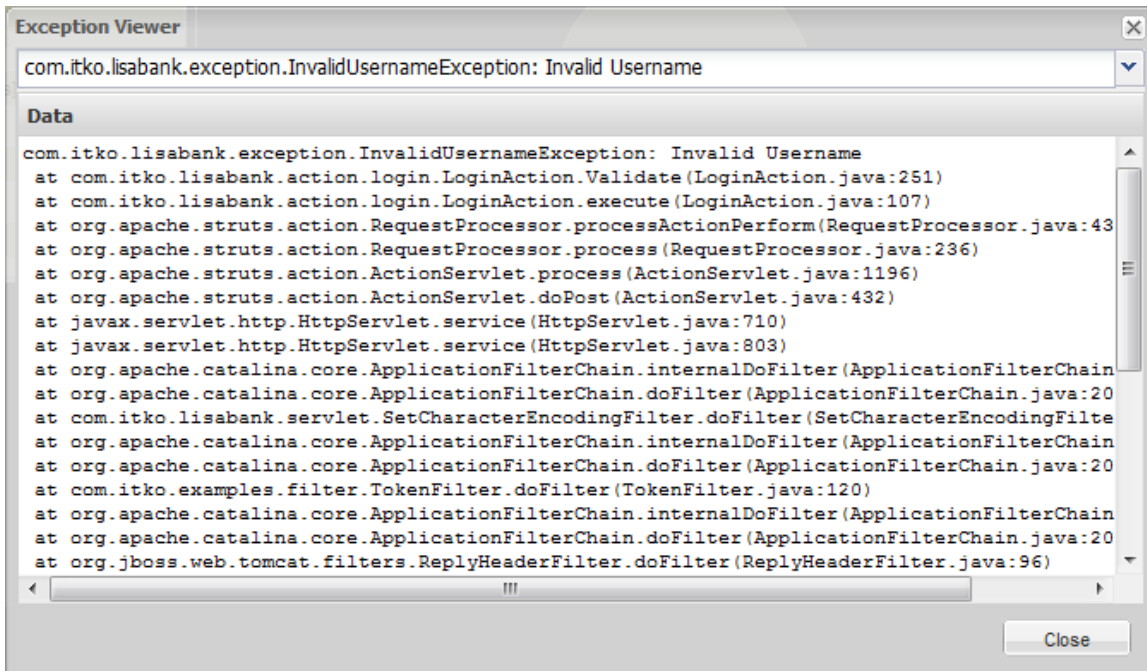
To import paths

1. In the right panel, right-click anywhere in the Paths tab and select Import Paths.
The Path Import Wizard appears.
2. Click Browse and select the export file.
3. Click Next.
4. The Import tab displays the paths that will be imported.
The Skip tab displays any paths that will not be imported because they are already in the Pathfinder database.
The Agents tab displays the ID and names of the Agents.
5. Click Next.
6. Click Finish.

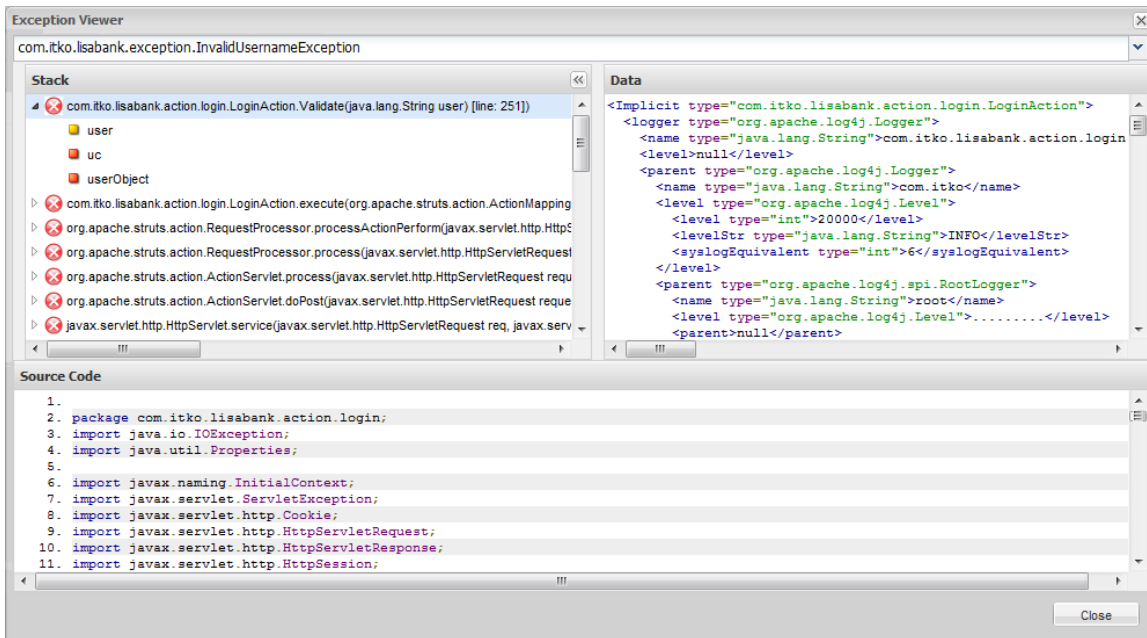
Viewing Exception Data

If a path contains an exception, the path graph displays the appropriate component in red. You can view detailed information about the exception.

For some exceptions, the Pathfinder Console displays the stack trace.



For other exceptions, the Pathfinder Console also displays the relevant XML data and the actual source code. In addition, the stack trace appears in a tree format. You can expand each node in the tree to show one or more subnodes. Each subnode has an icon that indicates whether it represents "this" (green), an argument (yellow), or a normal variable (red).



This enhanced view of the exception data is referred to as a "super stack." You can enable or disable super stacks by editing the **lisa.agent.superstack.enabled** property in the **rules.properties** file.

```
# Enabling superstacks
property key=lisa.agent.superstack.enabled value=true
```

To view exception data

1. In the path graph, right-click the node that appears in red and select View Exceptions. The Exception Viewer appears.
2. If the viewer contains a single panel named Data, then review the stack trace.
3. If the viewer contains three panels (Stack, Data, and Source Code), then do the following:
 - The Stack panel displays the stack trace in a tree format. The Data and Source Code panels are initially empty. You can expand each node in the tree to show one or more subnodes.
 - In the Stack panel, select a node. The Data panel displays the XML data. The Source Code panel displays the source code for the selected node.
4. When you are done, click Close.

Viewing Agent Information

The Agents tab in the Pathfinder Console displays information about Agents in the same subnet.

The left panel contains a graphical view of the network topology.

The right panel contains the following tabs:

- The VM tab displays properties for the Agent and the VM in which it is running. The Agent properties include the version number, which can be different from the LISA version number.
- The Performance tab displays statistics on CPU usage, memory usage, and I/O throughput.
- The J2EE Web Info tab displays information from the web.xml deployment descriptor file. For example, you can view the servlet and servlet-mapping elements.
- The JNDI Info tab covers all the web applications in the Agent.

Stopping and Starting Agent Transaction Capture

The Pathfinder Console lets you control whether an Agent is capturing transactions.

To stop Agent transaction capture

1. In the Pathfinder Console, click the Agents tab.
2. Right-click the Agent and select Stop Dispatching. The Agent stops capturing transactions.

To start Agent transaction capture

1. In the Pathfinder Console, click the Agents tab.
2. Right-click the Agent and select Start Dispatching.
The Agent resumes capturing transactions.

Integrating Pathfinder Broker and Portal Server

Integrating Pathfinder Broker and Portal Server with LISA Registry

When you run the LISA registry, it will attempt to start the Pathfinder Broker by default.

You will see console output as follows:

```
[TestRegistry] LISA_HOME set to /Users/john/projects/sandbox/lisa/dist/ [TestRegistry] LISA Version :: 0.0 (0.0.0.0) [TestRegistry]
[TestRegistry] Creating a LISA Registry, name=lisa.Registry [TestRegistry] [INFO][9659][main][09:00:40 (667)] LISA AGENT: No
custom rules loaded from /Users/john/projects/sandbox/lisa/rules.properties [TestRegistry] Oct 12, 2009 9:00:41 PM
org.apache.activemq.broker.BrokerService start [TestRegistry] INFO: Using Persistence Adapter:
AMQPersistenceAdapter(activemq-data/localhost) [TestRegistry] Oct 12, 2009 9:00:41 PM
org.apache.activemq.store.amq.AMQPersistenceAdapter start [TestRegistry] INFO: AMQStore starting using directory:
activemq-data/localhost [TestRegistry] Oct 12, 2009 9:00:41 PM org.apache.activemq.kaha.impl.KahaStore initialize [TestRegistry]
INFO: Kaha Store using data directory activemq-data/localhost/kr-store/state [TestRegistry] Oct 12, 2009 9:00:41 PM
org.apache.activemq.store.amq.AMQPersistenceAdapter start [TestRegistry] INFO: Active data files: [] [TestRegistry] Oct 12, 2009
9:00:41 PM org.apache.activemq.broker.BrokerService getBroker [TestRegistry] INFO: ActiveMQ null JMS Message Broker
(localhost) is starting [TestRegistry] Oct 12, 2009 9:00:41 PM org.apache.activemq.broker.BrokerService getBroker [TestRegistry]
INFO: For help or more information please see: http://activemq.apache.org/ [TestRegistry] Oct 12, 2009 9:00:42 PM
org.apache.activemq.kaha.impl.KahaStore initialize [TestRegistry] INFO: Kaha Store using data directory
activemq-data/localhost/kr-store/data [TestRegistry] Oct 12, 2009 9:00:42 PM
org.apache.activemq.transport.TransportServerThreadSupport doStart [TestRegistry] INFO: Listening for connections at:
tcp://Larry.local:61616 [TestRegistry] Oct 12, 2009 9:00:42 PM org.apache.activemq.broker.TransportConnector start [TestRegistry]
INFO: Connector tcp://Larry.local:61616 Started [TestRegistry] Oct 12, 2009 9:00:42 PM
org.apache.activemq.broker.TransportConnector start [TestRegistry] INFO: Connector vm://localhost Started [TestRegistry] Oct 12,
2009 9:00:42 PM org.apache.activemq.broker.BrokerService start [TestRegistry] INFO: ActiveMQ JMS Message Broker (localhost,
ID:Larry.local-56909-1255399242018-0:0) started [TestRegistry] [INFO][9659][main][09:00:42 (579)] LISA AGENT: Started Broker
tcp://localhost:61616 [TestRegistry] LISA Registry Ready.
```

To prevent this, pass "-nobroker" on the command line when you launch the registry.

Configuration of the Broker inside the registry is accomplished with **lisa.properties** entries as follow:

```
# Our Pathfinder Broker can be launched within the registry; these settings are required for that lisa.pathfinder.broker.port=61616
lisa.pathfinder.broker.db=org.apache.derby.jdbc.ClientDriver::jdbc:derby://localhost:1527/reports/pathfinder.db;create=true::pathfinder::
```

The port is used by the agents, consoles, and broker to communicate back and forth (as it is messaging based). The DB entry defines where the broker will store historic data about the transactions it saw and statistics, and other information.

If you use LISA's embedded Derby database, you have nothing to do. To change that database to your own, change this line by creating your own entry in **local.properties** or **site.properties**. The database and user account shown must already exist in your target database instance. The broker will create the tables for you.

Disabling Pathfinder

You can disable Pathfinder by adding the **lisa.pathfinder.on** property to the **local.properties** file.

To disable Pathfinder

1. Open the **local.properties** file.
2. Add the **lisa.pathfinder.on** property and set the value to **false**.

```
lisa.pathfinder.on=false
```

3. Save the **local.properties** file.

Creating Cases



This feature requires a separate license.

LISA Pathfinder lets you capture detailed information about application behavior. The application must be running on an Agent-enabled computer.

A case does not need to represent a defect.

You create a case within the application, and then view it from the Pathfinder Console. The case includes basic information, such as a title, severity level, and description. You can also view the corresponding paths, which expose the actions of the underlying components.

Email Settings for New Cases

When you create a case, Pathfinder sends an email notification to a specified email address. The email includes a link to the case in the Pathfinder Console.

The **rules.properties** file in the **LISA_HOME/bin** directory has properties that control this feature. Be sure to configure these properties before you create a case.

lisa.pathfinder.smtp.server

This property contains the name of the SMTP server to use for sending the email.

lisa.pathfinder.email.from

This property specifies the sender that will appear in the email. The default value is **lisa@itko.com**.

lisa.pathfinder.email.subject

This property specifies the subject line of the email. The default value is **New Pathfinder Case: %1**. The characters **%1** will be replaced with the title of the case.

lisa.pathfinder.email.body

This property specifies the body of the email. The default value is **Your Pathfinder case has been submitted and can be viewed by clicking on the following link:<CR><CR>%1**. The characters **%1** will be replaced with the link to the case in the Pathfinder Console.

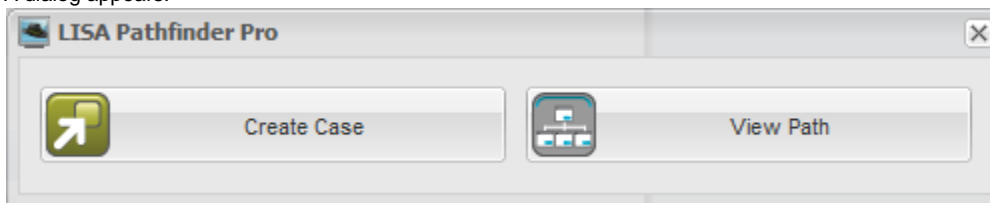
Capturing Cases in an Application

This procedure assumes that the application is running on an Agent-enabled computer.

For the email functionality to work, the [email settings](#) must be configured in the **rules.properties** file.

To capture a case in an application

1. While pressing the Alt key, click anywhere in the application window.
A dialog appears.



2. Click **Create Case**.
The Create Case dialog appears.

LISA Pathfinder Pro - Create Case

Email:

Title:

Defect #:

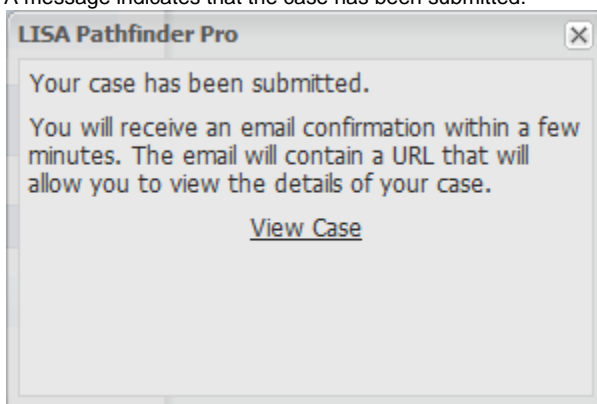
Severity:
 ▼

Description:

☒ Take screenshot

Submit Cancel

3. Enter the following information:
 - **Email:** A message about the new case will be sent to this email address.
 - **Title**
 - **Defect number**
 - **Severity:** The choices are Low, Medium, High, and Critical.
 - **Description:** This field is optional.
4. If you want to include a screen shot of the application at the time of capture, then leave the **Take screenshot** check box selected. Otherwise, clear the check box.
5. Click **Submit**.
 A message indicates that the case has been submitted.

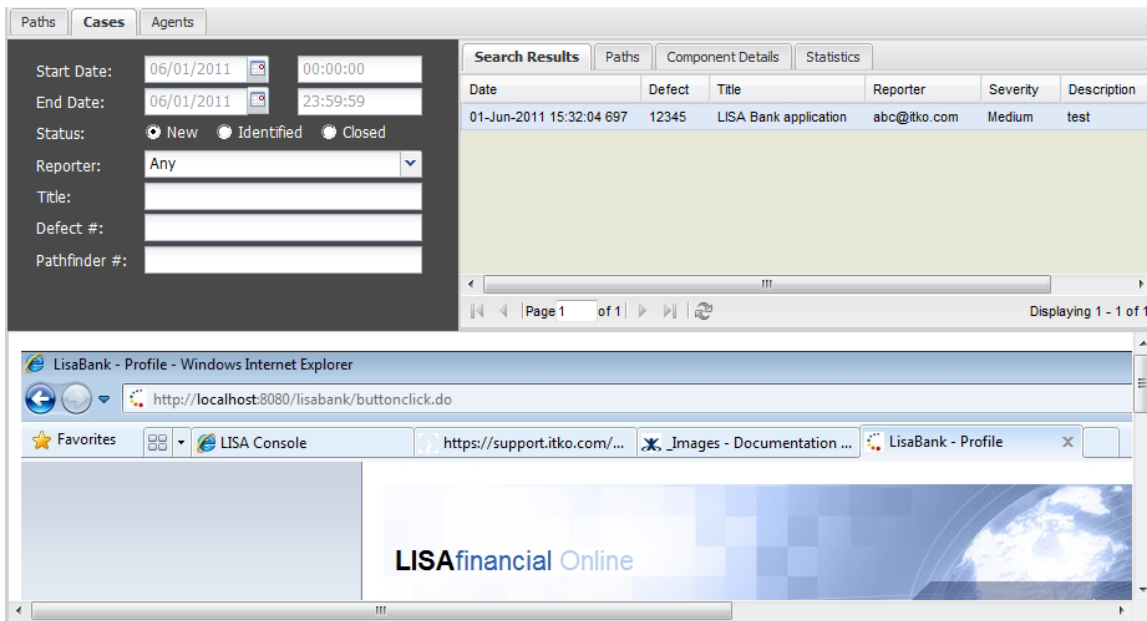


6. If you want to display the case in the [Pathfinder Console](#), then click the **View Case** link. Otherwise, close the message.

Pathfinder Console - Cases Tab

The Cases tab in the Pathfinder Console lets you view all the cases that have been captured and stored in the Pathfinder database.

The following image shows an example of the Cases tab. The upper-left panel contains criteria for [searching the cases](#) in the database. The upper-right panel contains a grid that displays basic information about each case. This information was entered by the user who captured the case. If a screenshot is included with the case, then the screenshot appears in the lower panel.



Searching Cases

The upper-left panel of the **Cases** tab contains criteria for searching the cases in the Pathfinder database.

You can specify the following date-range criteria:

- Start date and time
- End date and time



When the status is set to **New**, the date-range criteria is disabled.

You can specify one of following statuses: **New**, **Identified**, or **Closed**.

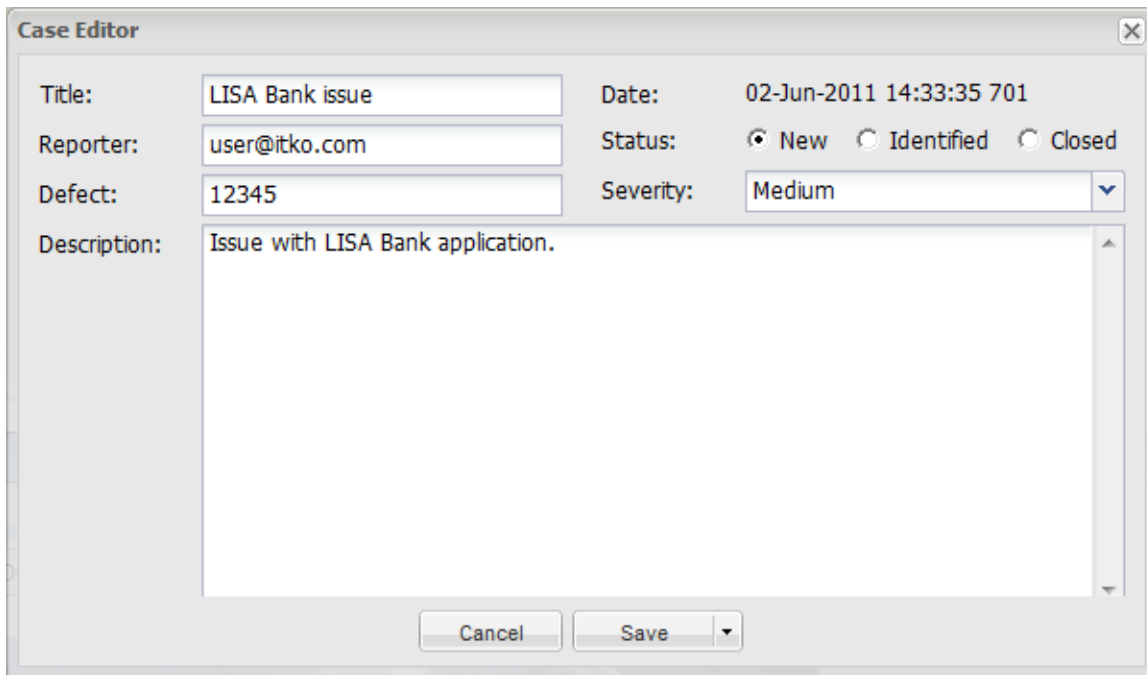
The initial status of a case is **New**. While [editing the case information](#), you can change the status to **Identified** or **Closed**.

Editing Case Information

You can modify the information for an existing case from the Pathfinder Console.

To edit case information

1. In the cases grid of the Pathfinder Console, right-click a case and select Edit. The Case Editor dialog appears.



The Case Editor dialog box contains the following fields and controls:

- Title:** Text field with value "LISA Bank issue".
- Reporter:** Text field with value "user@itko.com".
- Defect:** Text field with value "12345".
- Description:** Text area with value "Issue with LISA Bank application.".
- Date:** Text field with value "02-Jun-2011 14:33:35 701".
- Status:** Radio buttons for "New" (selected), "Identified", and "Closed".
- Severity:** Dropdown menu with value "Medium".
- Buttons:** "Cancel" and "Save" buttons at the bottom.

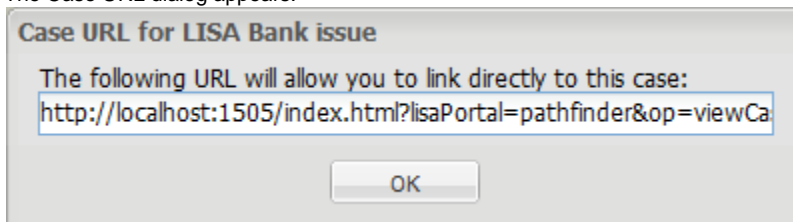
2. Edit one or more of the following fields: title, reporter, defect tracking number, description, status, and severity.
3. Click Save.

Obtaining the Direct URL of a Case

The Pathfinder Console lets you obtain a URL that points directly to an existing case.

To obtain the direct URL of a case

1. In the cases grid of the Pathfinder Console, right-click a case and select Copy Link. The Case URL dialog appears.



The Case URL dialog box displays the following information:

- Title:** Case URL for LISA Bank issue
- Message:** The following URL will allow you to link directly to this case:
- URL:** `http://localhost:1505/index.html?lisaPortal=pathfinder&op=viewCa`
- Button:** "OK" button at the bottom.

2. Copy the URL that appears in the field.
 3. Click OK.
- You can now go directly to the case by entering the URL in a web browser.

Viewing the Transactions for a Case

The Pathfinder Console lets you view the corresponding transactions for each case.

You can mark a node in one of the transactions. For example, you can perform this action to identify the origin of a defect.

To view the transactions for a case

1. In the cases grid of the Pathfinder Console, select a case.
2. Go to the Paths tab.
The transactions for the case appear.
3. Click a path to view the nodes in the path graph.
4. If you want to mark a node, then select the node and click Identify.
The color of the node changes to red. The color of the path changes to purple.
5. If you want to unmark a node, then select the node and click Unidentify.

Creating Data Sets



This feature requires a separate license.

Pathfinder enables you to create data sets by extracting information from transactions.

For a detailed description of data sets, see [Data Sets](#).

To create a data set

1. Open the Pathfinder Console.
2. In the **Paths** tab, select a transaction.
3. In the path graph, select a component.
4. Click the Dataset icon at the bottom of the console. The **Dataset Editor** window opens.
5. In the Request or Response tab, expand the tree, select the target node, and click Add. The Key and Value columns are populated with the node name and its XPath expression, respectively.
6. Select and add additional nodes as necessary. If you want to remove a node that you added, select the check box in the appropriate row and click Delete.
7. When you are done, click Generate. A dialog appears.
8. Enter a name for the data set.
9. Click Create.
10. Click the **Dataset** link to download the data set file.

Creating Baseline Test Cases and Suites



This feature requires a separate license.

Pathfinder enables you to create a "baseline" test case or suite from a set of actual transactions.

The following transaction types are supported:

- [EJB](#)
- [JMS](#)
- [REST](#)
- [Web Service](#)
- [WebSphere MQ](#)

Creating EJB Baselines

The steps for creating EJB baselines are grouped into the following procedures:

- [Generating the EJB Baseline Assets](#)
- [Importing the EJB Baseline Assets](#)
- [Viewing and Running EJB Baseline Test Cases](#)
- [Viewing and Running EJB Baseline Suites](#)

Generating the EJB Baseline Assets

In this procedure, you run the Baseline Wizard on a set of EJB transactions in the Pathfinder Console. The result is a file with the .pfi extension. The .pfi file contains the assets for the EJB baseline test case or suite.

To generate the EJB baseline assets

1. Open the **rules.properties** file in the **LISA_HOME/agent** directory.
2. Locate the **lisa.agent.transaction.weight** property and set the value to 15.
3. Save the **rules.properties** file.
4. Start or restart the application that is using the Agent.
5. Open the Pathfinder Console.
6. Make sure that all the transactions that you want to include in the baseline are viewable in the Paths tab. If necessary, change the [filter criteria](#). It does not matter if there are too many transactions to be viewed on one page.
7. In the Paths tab, select a transaction that includes an EJB component.
8. In the path graph, select an EJB node.
9. Click the Baseline icon at the bottom of the Pathfinder Console. The Baseline Wizard opens.
10. Use the wizard to configure and generate the baseline. If you are generating a baseline test case, then set the creation mode to Consolidated. If you are generating a baseline suite, then set the creation mode to Expanded. For detailed information about each step, see [Baseline Wizard](#).
In the last step of the wizard, you are prompted to download the baseline.

11. Click the link and save the .pfi file.
12. Click Finish. You can now [import the .pfi file](#) into LISA Workstation.

Importing the EJB Baseline Assets

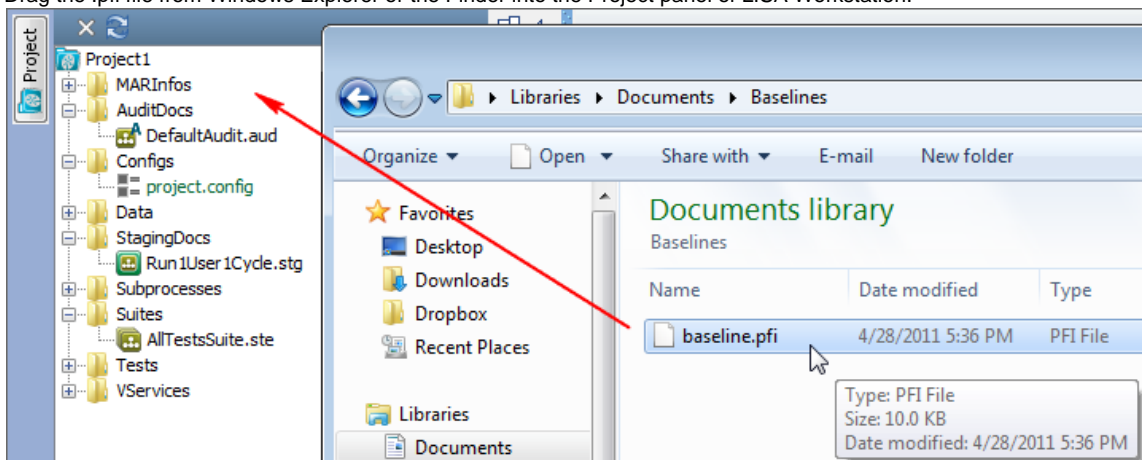
When you [generate the EJB baseline assets](#), the result is a file with the .pfi extension. You can import the file into LISA Workstation by using drag-and-drop.



As of release 6.0.6, you can also import the .pfi file by right-clicking the Tests folder and selecting Import Files.

To import the EJB baseline assets

1. In LISA Workstation, open or create a project where you want to place the EJB baseline test case or suite.
2. Open Windows Explorer or the Finder.
3. Locate the directory where you saved the .pfi file.
4. Change the size of Windows Explorer or the Finder so that you can also see the Project panel of LISA Workstation.
5. Drag the .pfi file from Windows Explorer or the Finder into the Project panel of LISA Workstation.



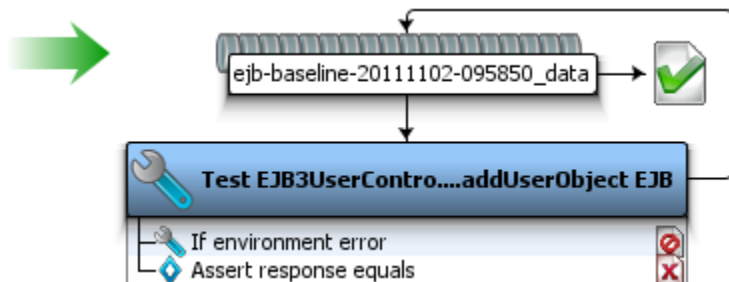
LISA Workstation extracts the assets from the .pfi file and adds them to the appropriate folders.

Viewing and Running EJB Baseline Test Cases

The contents of an EJB baseline test case depend on the validation mode.

Shallow Validation

If you choose shallow validation in the [Baseline Wizard](#), the EJB baseline test case will have a Pathfinder Agent Script step that reads from a Large Data data set.



The Large Data data set contains information that changes for each transaction in the baseline. By default, the Large Data data set is local, rather than global.

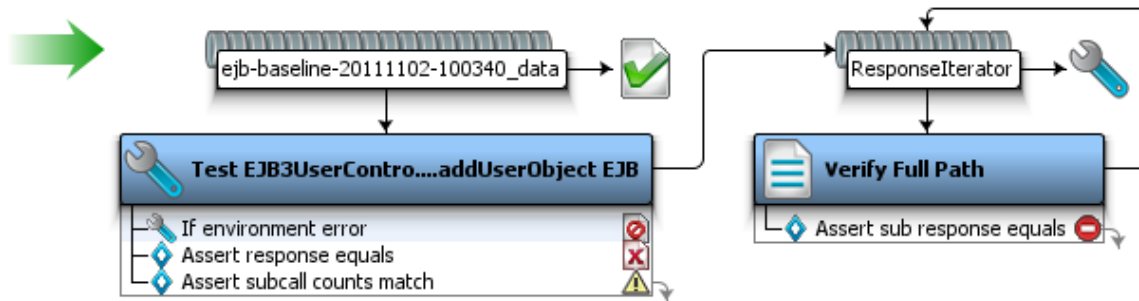
The Pathfinder Agent Script step includes a Graphical XML Side-by-Side Comparison assertion with the name **Assert response equals**. When you run the baseline test case, this assertion verifies that the actual response matches the expected response. If the responses do not match, then the test fails.

Full Validation



With full validation, the service under test must be Agent enabled when you run the baseline test case. Otherwise, the test case cannot gather the downstream element response information to compare to the expected results.

If you choose full validation in the [Baseline Wizard](#), then the EJB baseline test case also has a do-nothing step with the name **Verify Full Path**.



This do-nothing step includes a Graphical XML Side-by-Side Comparison assertion with the name **Assert sub response equals**. When you run the baseline test case, this assertion verifies that the actual response for each subnode matches the expected response. A subnode is any node to the right of the node that was selected in the path graph at baseline creation time. If the responses do not match, then the test generates an error.

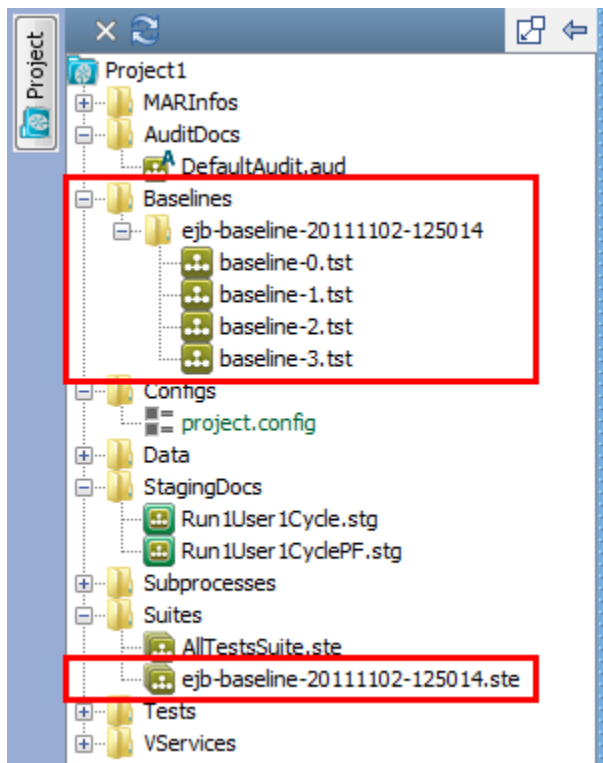
You must run the test case with a staging document in which the Enable LISA Pathfinder check box is selected. The **Run1User1CyclePF.stg** staging document, which is included by default in all projects, has this check box selected.

Viewing and Running EJB Baseline Suites

The test cases in an EJB baseline suite are located in the Baselines folder of the project, rather than the Tests folder.

The suite document is located in the usual place: the Suites folder.

The following image shows a project that contains a baseline suite. The Baselines folder and the suite document are highlighted.



The test cases in an EJB baseline suite are similar to [EJB baseline test cases](#). However, they do not include a Large Data data set. The data is stored in the Pathfinder Agent Script step instead.

If you chose full validation in the [Baseline Wizard](#), then you must run the suite with a staging document in which the Enable LISA Pathfinder check

box is selected. The **Run1User1CyclePF.stg** staging document has this check box selected.

To run the baseline suite, follow the steps in [Running Test Suites](#). The result of each test case is displayed as a comparison between the expected response and the actual response.

The screenshot shows the 'Stage Suite Execution' window. The 'Suite Document' section indicates the location is 'Suites/ejb-baseline-20111102-125014.ste' and the current test is 'Complete'. The 'Baseline Results' panel on the left lists four test cases: 'baseline-0.tst', 'baseline-1.tst', 'baseline-2.tst', and 'baseline-3.tst', all marked with green circles indicating success. The main area displays a side-by-side comparison of 'Expected' and 'Observed' XML responses for a test case. The XML is for a JMS message with fields like login, password, first name, last name, email, phone, SSN, new flag, role key, and account class. The 'Expected' and 'Observed' columns show identical XML structures, with line numbers 1 through 15 visible. At the bottom, there are two groups of buttons: 'Selected Baseline Commands' (Ignore, Save, Update, Undo, Redo, Detect) and 'Whole Suite Commands' (Rerun Suite, View Reports).

The Baseline Results panel indicates whether each test succeeded (green) or failed (red). If a test fails, you can do either of the following actions:

- Select the test and click Ignore. The test will not be executed in future runs of the suite.
- Select the test and click Update. The suite is updated with the actual response that was received.

To run the suite again, click Rerun Suite.

To display the run statistics in the Reporting Console, click View Reports.

Creating JMS Baselines

The steps for creating JMS baselines are grouped into the following procedures:

- [Generating the JMS Baseline Assets](#)
- [Importing the JMS Baseline Assets](#)
- [Viewing and Running JMS Baseline Test Cases](#)
- [Viewing and Running JMS Baseline Suites](#)

JMS baselines are supported for configurations in which JNDI is used to obtain the JMS connection factory.

JMS baselines are also supported for configurations in which a direct API from IBM WebSphere MQ, Java CAPS, SonicMQ, or TIBCO is used to obtain the JMS connection factory.



If JMS message objects are reused and forwarded by a service or client without being modified, then the behavior of Pathfinder is undefined.

Generating the JMS Baseline Assets

In this procedure, you run the Baseline Wizard on a set of JMS transactions in the Pathfinder Console. The result is a file with the .pfi extension. The .pfi file contains the assets for the JMS baseline test case or suite.

To generate the JMS baseline assets

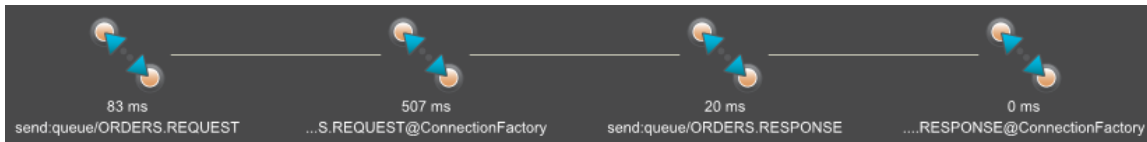
1. Open the Pathfinder Console.
2. Make sure that all the transactions that you want to include in the baseline are viewable in the **Paths** tab. If necessary, change the [filter criteria](#). It does not matter if there are too many transactions to be viewed on one page.
3. In the **Paths** tab, select a transaction that has a value of **jms** in the Category column.
4. In the path graph, select one of the nodes. For more information, see [JMS Baseline Nodes](#).
5. Click the **Baseline** icon at the bottom of the Pathfinder Console. The Baseline Wizard opens.
6. Use the wizard to configure and generate the baseline. If you are generating a baseline test case, then set the creation mode to Consolidated. If you are generating a baseline suite, then set the creation mode to Expanded. For detailed information about each step, see [Baseline Wizard](#).
In the last step of the wizard, you are prompted to download the baseline.
7. Click the link and save the .pfi file.
8. Click Finish.
You can now [import the .pfi file](#) into LISA Workstation.

JMS Baseline Nodes

The procedure for [generating the JMS baseline assets](#) includes a step where you select a node in the path graph.

For a scenario with one request and one response, the path graph should contain four nodes. These nodes represent the following activities:

- The client produces the request message.
- The service consumes the request message.
- The service produces the response message.
- The client consumes the response message.



Select the first node in the path graph. For an exception to this rule, see [Duplicate Nodes](#).

Pathfinder will search the transactions in the **Paths** tab for transactions that contain the same name as the node you select. The baseline will contain all transactions that involve sending a message to or receiving a message from the same queue.

Duplicate Nodes

Under some circumstances, Pathfinder will display two adjacent nodes that represent the same produce or consume operation. The names of the adjacent nodes will both begin with either **send:** or **recv:**.

If the first two nodes in the path graph both begin with **send:**, then select the *first* of the nodes.

If the first two nodes in the path graph both begin with **recv:**, then select the *second* of the nodes.

Importing the JMS Baseline Assets

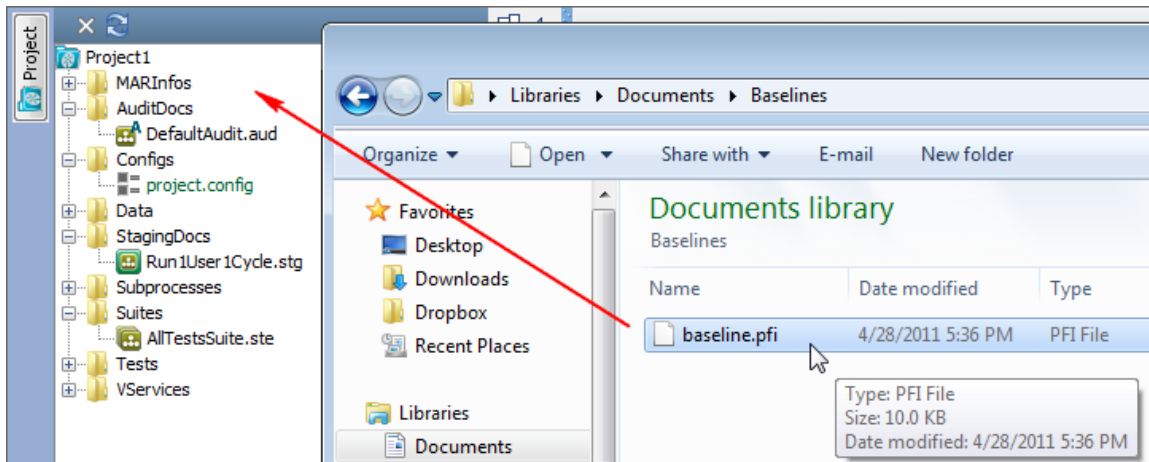
When you [generate the JMS baseline assets](#), the result is a file with the .pfi extension. You can import the file into LISA Workstation by using drag-and-drop.



As of release 6.0.6, you can also import the .pfi file by right-clicking the Tests folder and selecting Import Files.

To import the JMS baseline assets

1. In LISA Workstation, open or create a project where you want to place the JMS baseline test case or suite.
2. Open Windows Explorer or the Finder.
3. Locate the directory where you saved the .pfi file.
4. Change the size of Windows Explorer or the Finder so that you can also see the Project panel of LISA Workstation.
5. Drag the .pfi file from Windows Explorer or the Finder into the Project panel of LISA Workstation.



LISA Workstation extracts the assets from the .pfi file and adds them to the appropriate folders.

Viewing and Running JMS Baseline Test Cases

The contents of a JMS baseline test case depend on the validation mode.

Shallow Validation

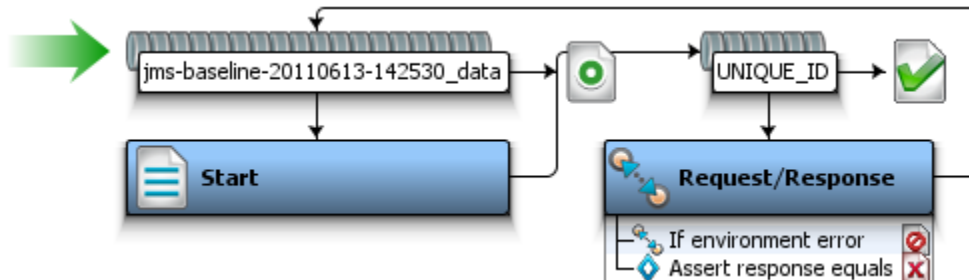
If you select shallow validation in the [Baseline Wizard](#), the steps that appear in the JMS baseline test case depend on the messaging scenario.

The test case for most single-request, single-response scenarios will have two steps:

- A do-nothing step that reads from a Large Data data set
- A messaging step that combines the request and response. The messaging step can be any of the following: JMS Messaging (JNDI), IBM WebSphere MQ, JCAPS Messaging (Native), SonicMQ Messaging (Native), or TIBCO Direct JMS. In the WebSphere MQ step, the client mode is set to JMS.

The test case may also include a Unique Code Generator data set for generating a correlation ID.

The following image shows an example test case. The first step is a do-nothing step. The second step is a messaging step with the name **Request/Response**.



The Large Data data set contains information that changes for each transaction in the baseline. This information includes the request payload, response payload, and JMS message properties. By default, the Large Data data set is local, rather than global.

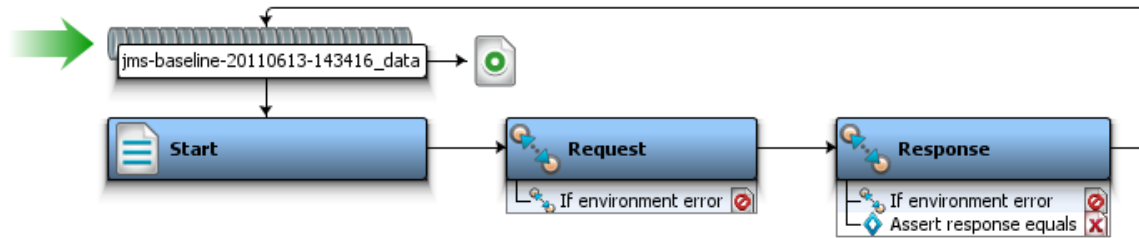
The following image shows a partial set of columns from the Large Data data set.

dest1.request.payload	dest1.request.JMSCorrelationID	dest1.request.JMSExpiration	dest1.request.JMSMessageID	dest1.request.JMSRedelivered	dest1.request.JMSPriority
<request><item><name... a5b497e8-d9f6-468e-b628-3520...	0		ID:EMS-SERVER.1704ED817EA...	false	4
<request><item><name... 946501bf-9b2b-48ed-989d-3a11...	0		ID:EMS-SERVER.1704ED817EA...	false	4
<request><item><name... c38da361-c8d8-426b-93c5-90c76...	0		ID:EMS-SERVER.1704ED817EA...	false	4

The messaging step includes a Graphical XML Side-by-Side Comparison assertion with the name **Assert response equals**. When you run the baseline test case, this assertion verifies that the actual response matches the expected response. If the responses do not match, then the test fails.

Other scenarios will divide the request and response into separate steps.

The following image shows an example test case. The first step is a do-nothing step. The second step is a messaging step with the name **Request**. The third step is a messaging step with the name **Response**.



For information that does not change, Pathfinder generates properties. This information includes destination information, connection information, and XML Diff options.

If the client and service use different JNDI connection settings, then Pathfinder generates two sets of properties for the connection information.

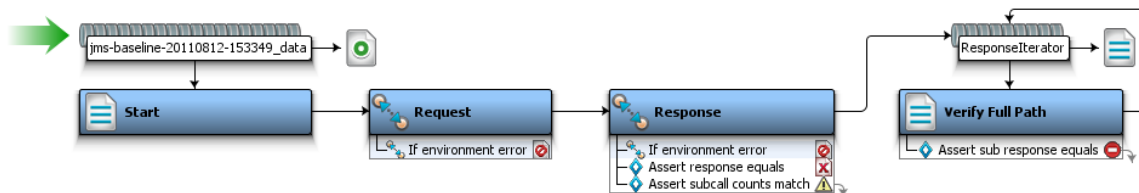
Full Validation

If you select full validation in the [Baseline Wizard](#), the JMS baseline test case also has a do-nothing step with the name **Verify Full Path**.



With full validation, the service under test must be Agent enabled when you run the baseline test case. Otherwise, the test case cannot gather the downstream element response information to compare to the expected results.

The following image shows an example test case. The first step is a do-nothing step. The second step is a messaging step with the name **Request**. The third step is a messaging step with the name **Response**. The fourth step is a do-nothing step with the name **Verify Full Path**.



This do-nothing step includes a Graphical XML Side-by-Side Comparison assertion with the name **Assert sub response equals**. When you run the baseline test case, this assertion verifies that the actual response for each subnode matches the expected response. A subnode is any node to the right of the node that was selected in the path graph at baseline creation time. If the responses do not match, then the test generates an error.

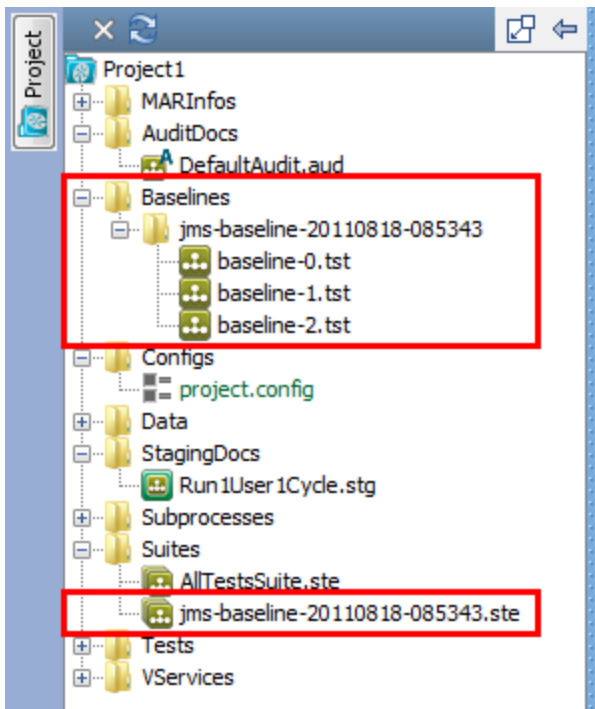
You must run the test case with a staging document in which the Enable LISA Pathfinder check box is selected. The **Run1User1CyclePF.stg** staging document, which is included by default in all projects, has this check box selected.

Viewing and Running JMS Baseline Suites

The test cases in a JMS baseline suite are located in the Baselines folder of the project, rather than the Tests folder.

The suite document is located in the usual place: the Suites folder.

The following image shows a project that contains a baseline suite. The Baselines folder and the suite document are highlighted.



The test cases in a JMS baseline suite are similar to [JMS baseline test cases](#). However, they do not include the do-nothing step that reads from a Large Data data set. The request and response data are stored in the messaging steps instead.

As with JMS baseline test cases, Pathfinder generates properties for destination information, connection information, and XML Diff options.

If you chose full validation in the [Baseline Wizard](#), then you must run the suite with a staging document in which the Enable LISA Pathfinder check box is selected. The **Run1User1CyclePF.stg** staging document has this check box selected.

To run the baseline suite, follow the steps in [Running Test Suites](#). The result of each test case is displayed as a comparison between the expected response and the actual response.

Stage Suite Execution

Suite Document

Location: Suites/jms-baseline-20110818-085343.ste

Current Test: Complete.

Baseline Results

com.itko.lisa.testing.StagedSuite

- baseline-0.tst (C:\Users\jst)
- baseline-1.tst (C:\Users\jst)
- baseline-2.tst (C:\Users\jst)

Select Contents

Settings

Diff Viewer

Expected

0/0

Observed

1	<response>	1	<response>
2	<item>	2	<item>
3	<name>washer</name>	3	<name>washer</name>
4	<price>636</price>	4	<price>636</price>
5	</item>	5	</item>
6	</response>	6	</response>
7		7	

Selected Baseline Commands:

Ignore

Save

Update

Undo

Redo

Diff

Whole Suite Commands:

Rerun Suite

View Metrics

The Baseline Results panel indicates whether each test succeeded (green) or failed (red). If a test fails, you can do either of the following actions:

- Select the test and click Ignore. The test will not be executed in future runs of the suite.
- Select the test and click Update. The suite is updated with the actual response that was received.

To run the suite again, click Rerun Suite.

To display the run statistics in the Reporting Console, click View Reports.

Creating REST Baselines

The steps for creating REST baselines are grouped into the following procedures:

- [Generating the REST Baseline Assets](#)
- [Importing the REST Baseline Assets](#)
- [Viewing and Running REST Baseline Test Cases](#)
- [Viewing and Running REST Baseline Suites](#)

Generating the REST Baseline Assets

In this procedure, you run the Baseline Wizard on a set of REST transactions in the Pathfinder Console. The result is a file with the .pfi extension. The .pfi file contains the assets for the REST baseline test case or suite.

To generate the REST baseline assets

1. Open the Pathfinder Console.
2. Make sure that all the transactions that you want to include in the baseline are viewable in the **Paths** tab. If necessary, change the [filter criteria](#). It does not matter if there are too many transactions to be viewed on one page.
3. In the **Paths** tab, select a transaction that includes a REST component.
4. In the path graph, select a REST node.
5. Click the **Baseline** icon at the bottom of the Pathfinder Console. The Baseline Wizard opens.
6. Use the wizard to configure and generate the baseline. If you are generating a baseline test case, then set the creation mode to Consolidated. If you are generating a baseline suite, then set the creation mode to Expanded. For detailed information about each step, see [Baseline Wizard](#).
In the last step of the wizard, you are prompted to download the baseline.
7. Click the link and save the .pfi file.
8. Click Finish.
You can now [import the .pfi file](#) into LISA Workstation.

Importing the REST Baseline Assets

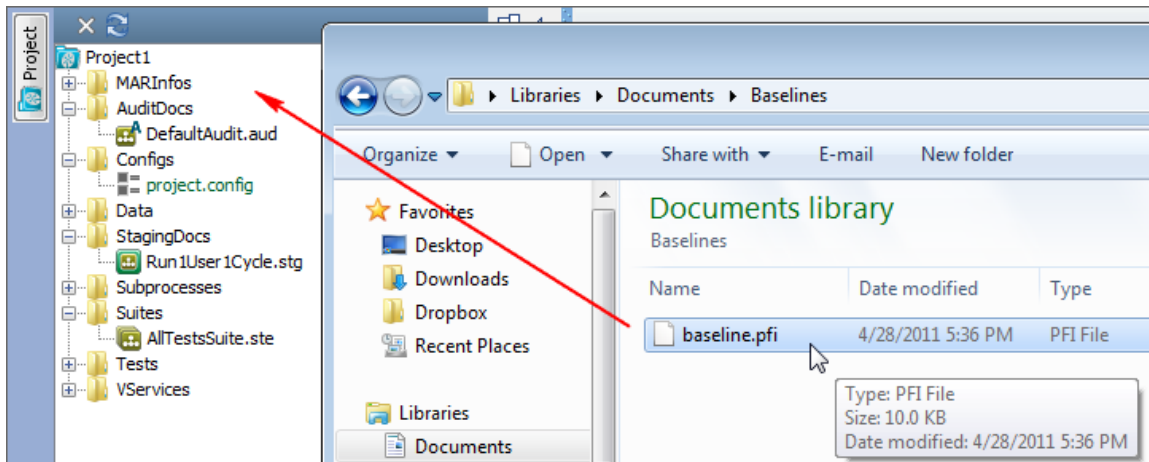
When you [generate the REST baseline assets](#), the result is a file with the .pfi extension. You can import the file into LISA Workstation by using drag-and-drop.



As of release 6.0.6, you can also import the .pfi file by right-clicking the Tests folder and selecting Import Files.

To import the REST baseline assets

1. In LISA Workstation, open or create a project where you want to place the REST baseline test case or suite.
2. Open Windows Explorer or the Finder.
3. Locate the directory where you saved the .pfi file.
4. Change the size of Windows Explorer or the Finder so that you can also see the Project panel of LISA Workstation.
5. Drag the .pfi file from Windows Explorer or the Finder into the Project panel of LISA Workstation.



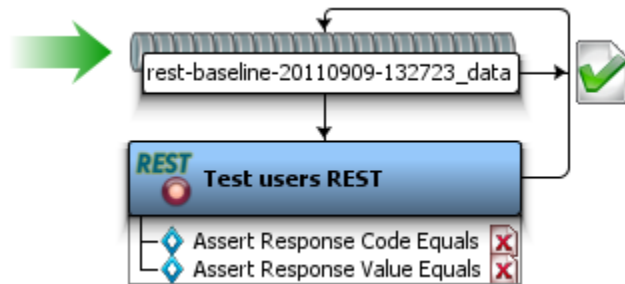
LISA Workstation extracts the assets from the .pfi file and adds them to the appropriate folders.

Viewing and Running REST Baseline Test Cases

The contents of a REST baseline test case depend on the validation mode.

Shallow Validation

If you choose shallow validation in the [Baseline Wizard](#), then the REST baseline test case has a REST step that reads from a Large Data data set.



The Large Data data set contains information that changes for each transaction in the baseline. This information includes the request body and the expected result. By default, the Large Data data set is local, rather than global.

Enabled	url	soap_action	content_type	request_body	expected_result	response_code
<input checked="" type="checkbox"/>	http://{{SERVER}}:{{PORT}}...		text/html	<HTTP method='GET'>... <?xml version="1.0" ... 200		

The REST step includes the following assertions:

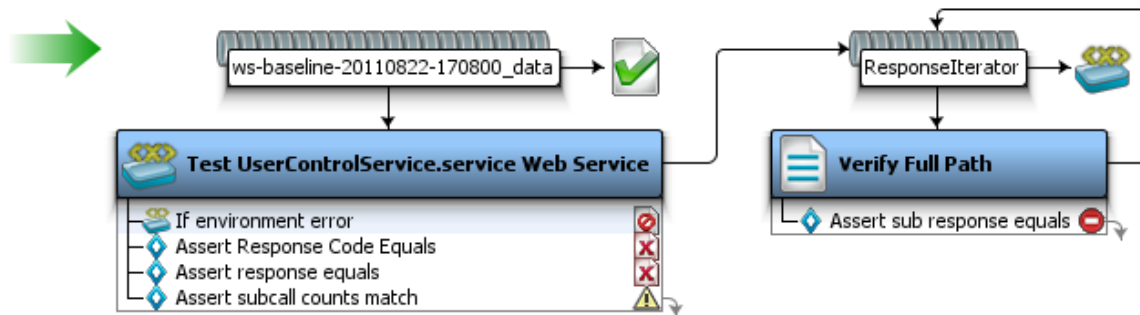
- An Ensure Properties Are Equal assertion with the name **Assert Response Code Equals**
- A Highlight Text Content for Comparison assertion with the name **Assert Response Value Equals**

Full Validation



With full validation, the service under test must be Agent enabled when you run the baseline test case. Otherwise, the test case cannot gather the downstream element response information to compare to the expected results.

If you choose full validation in the [Baseline Wizard](#), then the REST baseline test case also has a do-nothing step with the name **Verify Full Path**.



This do-nothing step includes a Graphical XML Side-by-Side Comparison assertion with the name **Assert sub response equals**. When you run the baseline test case, this assertion verifies that the actual response for each subnode matches the expected response. A subnode is any node to the right of the node that was selected in the path graph at baseline creation time. If the responses do not match, then the test generates an error.

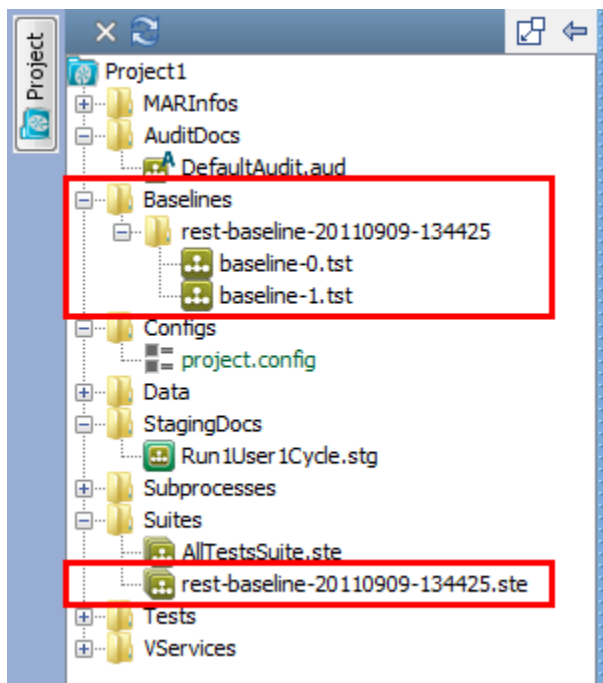
You must run the test case with a staging document in which the Enable LISA Pathfinder check box is selected. The **Run1User1CyclePF.stg** staging document, which is included by default in all projects, has this check box selected.

Viewing and Running REST Baseline Suites

The test cases in a REST baseline suite are located in the Baselines folder of the project, rather than the Tests folder.

The suite document is located in the usual place: the Suites folder.

The following image shows a project that contains a baseline suite. The Baselines folder and the suite document are highlighted.



The test cases in a REST baseline suite are similar to [REST baseline test cases](#). However, they do not include a Large Data data set. The data is stored in the REST step instead.

If you chose full validation in the [Baseline Wizard](#), then you must run the suite with a staging document in which the Enable LISA Pathfinder check box is selected. The **Run1User1CyclePF.stg** staging document has this check box selected.

To run the baseline suite, follow the steps in [Running Test Suites](#).

Creating Web Service Baselines

The steps for creating web service baselines are grouped into the following procedures:


- [Generating the Web Service Baseline Assets](#)
- [Importing the Web Service Baseline Assets](#)

- [Viewing and Running Web Service Baseline Test Cases](#)
- [Viewing and Running Web Service Baseline Suites](#)

Generating the Web Service Baseline Assets


In this procedure, you run the Baseline Wizard on a set of web service transactions in the Pathfinder Console. The result is a file with the .pfi extension. The .pfi file contains the assets for the web service baseline test case or suite.

To generate the web service baseline assets

1. Open the Pathfinder Console.
2. Make sure that all the transactions that you want to include in the baseline are viewable in the **Paths** tab. If necessary, change the [filter criteria](#). It does not matter if there are too many transactions to be viewed on one page.
3. In the **Paths** tab, select a transaction that includes a web service component.
4. In the path graph, select a web service node .
5. Click the Baseline icon at the bottom of the Pathfinder Console. The Baseline Wizard opens.
6. Use the wizard to configure and generate the baseline. If you are generating a baseline test case, then set the creation mode to Consolidated. If you are generating a baseline suite, then set the creation mode to Expanded. For detailed information about each step, see [Baseline Wizard](#).
In the last step of the wizard, you are prompted to download the baseline.
7. Click the link and save the .pfi file.
8. Click Finish.
You can now [import the .pfi file](#) into LISA Workstation.

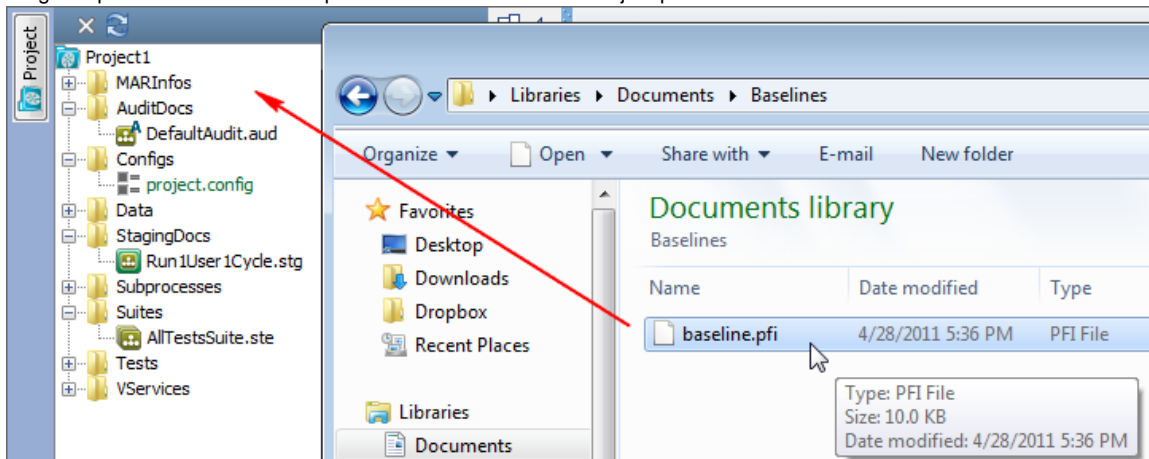
Importing the Web Service Baseline Assets

When you [generate the web service baseline assets](#), the result is a file with the .pfi extension. You can import the file into LISA Workstation by using drag-and-drop.

 As of release 6.0.6, you can also import the .pfi file by right-clicking the Tests folder and selecting Import Files.

To import the web service baseline assets

1. In LISA Workstation, open or create a project where you want to place the web service baseline test case or suite.
2. Open Windows Explorer or the Finder.
3. Locate the directory where you saved the .pfi file.
4. Change the size of Windows Explorer or the Finder so that you can also see the Project panel of LISA Workstation.
5. Drag the .pfi file from Windows Explorer or the Finder into the Project panel of LISA Workstation.



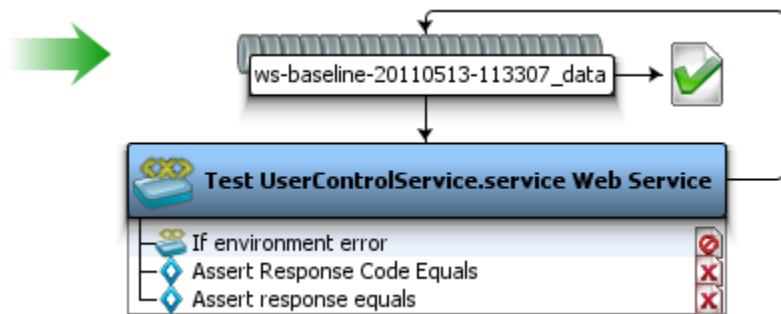
LISA Workstation extracts the assets from the .pfi file and adds them to the appropriate folders.

Viewing and Running Web Service Baseline Test Cases

The contents of a web service baseline test case depend on the validation mode.

Shallow Validation

If you choose shallow validation in the [Baseline Wizard](#), the web service baseline test case will have a Web Service Execution (XML) step that reads from a Large Data data set.



The Large Data data set contains information that changes for each transaction in the baseline. This information includes the request body and the expected result. By default, the Large Data data set is local, rather than global.

Enabled	url	soap_action	content_type	request_body	expected_result	response_code
<input checked="" type="checkbox"/>	http://{{WSSERVER}}:...		text/html	<?xml version="1.0" ...	<?xml version="1.0" ...	200
<input checked="" type="checkbox"/>	http://{{WSSERVER}}:...		text/html	<?xml version="1.0" ...	<?xml version="1.0" ...	200
<input checked="" type="checkbox"/>	http://{{WSSERVER}}:...		text/html	<?xml version="1.0" ...	<?xml version="1.0" ...	200

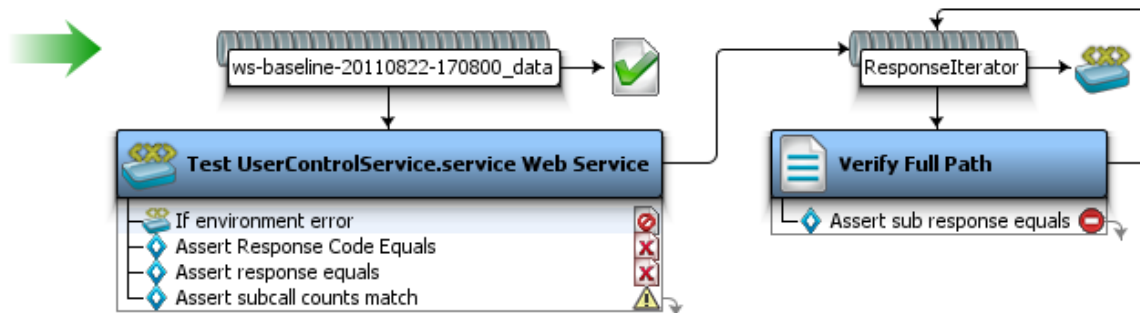
The Web Service Execution (XML) step includes a Graphical XML Side-by-Side Comparison assertion with the name **Assert response equals**. When you run the baseline test case, this assertion verifies that the actual response matches the expected response. If the responses do not match, then the test fails.

Full Validation



With full validation, the service under test must be Agent enabled when you run the baseline test case. Otherwise, the test case cannot gather the downstream element response information to compare to the expected results.

If you choose full validation in the [Baseline Wizard](#), then the web service baseline test case also has a do-nothing step with the name **Verify Full Path**.



This do-nothing step includes a Graphical XML Side-by-Side Comparison assertion with the name **Assert sub response equals**. When you run the baseline test case, this assertion verifies that the actual response for each subnode matches the expected response. A subnode is any node to the right of the node that was selected in the path graph at baseline creation time. If the responses do not match, then the test generates an error.

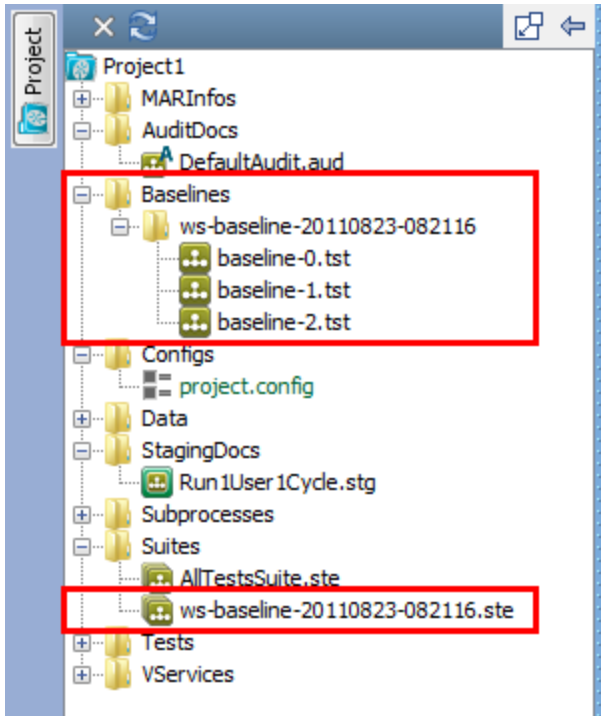
You must run the test case with a staging document in which the Enable LISA Pathfinder check box is selected. The **Run1User1CyclePF.stg** staging document, which is included by default in all projects, has this check box selected.

Viewing and Running Web Service Baseline Suites

The test cases in a web service baseline suite are located in the Baselines folder of the project, rather than the Tests folder.

The suite document is located in the usual place: the Suites folder.

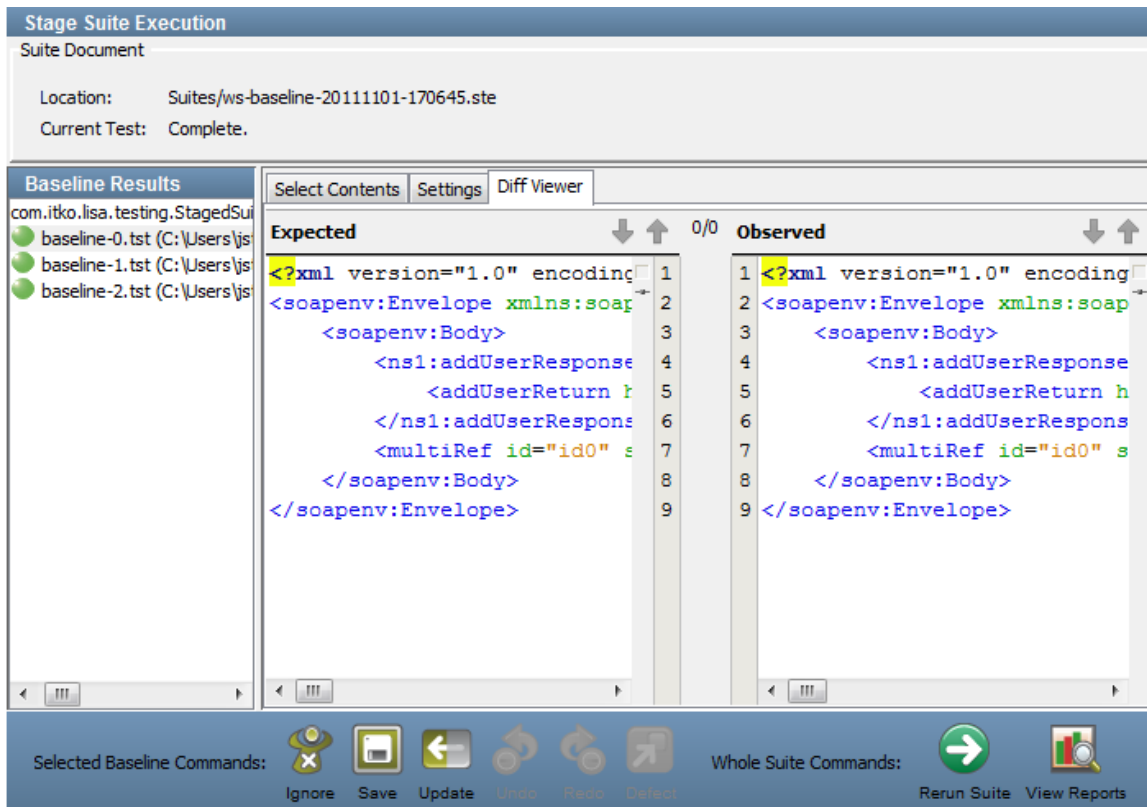
The following image shows a project that contains a baseline suite. The Baselines folder and the suite document are selected.



The test cases in a web service baseline suite are similar to [web service baseline test cases](#). However, they do not include a Large Data data set. The data is stored in the Web Service Execution (XML) step instead.

If you chose full validation in the [Baseline Wizard](#), then you must run the suite with a staging document in which the Enable LISA Pathfinder check box is selected. The **Run1User1CyclePF.stg** staging document has this check box selected.

To run the baseline suite, follow the steps in [Running Test Suites](#). The result of each test case is displayed as a comparison between the expected response and the actual response.



The Baseline Results panel indicates whether each test succeeded (green) or failed (red). If a test fails, you can do either of the following actions:

- Select the test and click Ignore. The test will not be executed in future runs of the suite.
- Select the test and click Update. The suite is updated with the actual response that was received.

To run the suite again, click Rerun Suite.

To display the run statistics in the Reporting Console, click View Reports.

Creating WebSphere MQ Baselines

The steps for creating WebSphere MQ baselines are grouped into the following procedures:

- [Generating the WebSphere MQ Baseline Assets](#)
- [Importing the WebSphere MQ Baseline Assets](#)
- [Viewing and Running WebSphere MQ Baseline Test Cases](#)
- [Viewing and Running WebSphere MQ Baseline Suites](#)

Generating the WebSphere MQ Baseline Assets

In this procedure, you run the Baseline Wizard on a set of WebSphere MQ transactions in the Pathfinder Console. The result is a file with the .pfi extension. The .pfi file contains the assets for the WebSphere MQ baseline test case or suite.

To generate the WebSphere MQ baseline assets

1. Determine which [WebSphere MQ scenario](#) you want to use, and enable the LISA Agent as appropriate.
2. Ensure that the required [WebSphere MQ drivers](#) are located in the **LISA_HOME/lib** directory.
3. (Optional) Configure the [WebSphere MQ properties](#) in the LISA Agent's **rules.properties** file.
4. Open the Pathfinder Console.
5. Make sure that all the transactions that you want to include in the baseline are viewable in the **Paths** tab. If necessary, change the [filter criteria](#). It does not matter if there are too many transactions to be viewed on one page.
6. In the **Paths** tab, select a transaction that has a value of **ibmmq** in the Category column.
7. In the path graph, select the first node.
8. Click the Baseline icon at the bottom of the Pathfinder Console. The Baseline Wizard opens.
9. Use the wizard to configure and generate the baseline. If you are generating a baseline test case, then set the creation mode to Consolidated. If you are generating a baseline suite, then set the creation mode to Expanded. For detailed information about each step, see [Baseline Wizard](#).
In the last step of the wizard, you are prompted to download the baseline.
10. Click the link and save the .pfi file.
11. Click Finish.
You can now [import the .pfi file](#) into LISA Workstation.

WebSphere MQ Baseline Scenarios

WebSphere MQ baselines vary depending on where the LISA Agent is enabled.

Scenario 1: Client Enabled, Service Not Enabled

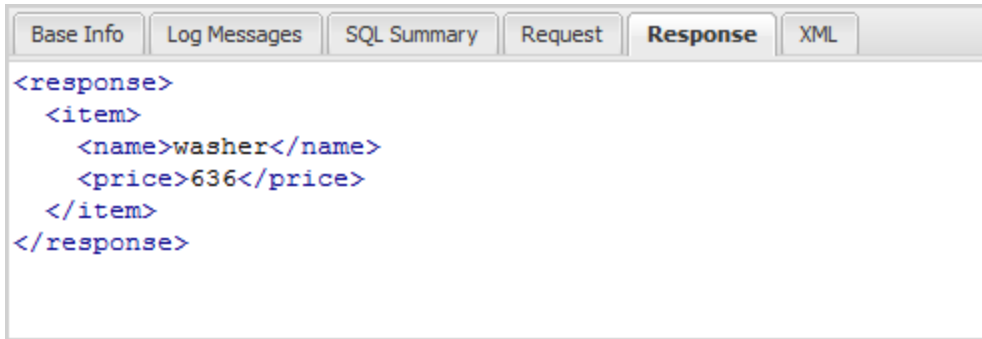
The client is Agent enabled, and the service is not Agent enabled. This scenario simulates a Java-based client running against a Windows or mainframe-based service that cannot be Agent enabled.

When you select a transaction in the Pathfinder Console, the path graph should contain two nodes.

The first node represents the request. You can view the message body by selecting the node and then clicking the Request tab.



The second node represents the response. You can view the message body by selecting the node and then clicking the Response tab.



```
<response>
  <item>
    <name>washer</name>
    <price>636</price>
  </item>
</response>
```

Scenario 2: Client Not Enabled, Service Enabled

The client is not Agent enabled, and the service is Agent enabled. This scenario simulates a Windows or mainframe-based client that cannot be Agent enabled running against a Java-based service.

The first transaction in the Pathfinder Console is a "priming" transaction. The Agent on the service side needs to discover that it is running on the service side and initialize itself.

The subsequent transactions are the same as for scenario 1. The path graph contains a request node and a response node.

Scenario 3: Client Enabled, Service Enabled

Both the client and the service are Agent enabled. This scenario simulates a Java-based client running against a Java-based service.

The first transaction in the Pathfinder Console is a "priming" transaction. The Agent on the service side needs to discover that it is running on the service side and initialize itself.

For the subsequent transactions, the path graph should contain four nodes:

- The first node represents the client's PUT for the request.
- The second node represents the service's GET for the request.
- The third node represents the service's PUT for the response.
- The fourth node represents the client's GET for the response.

WebSphere MQ Baseline Drivers

Before you can create a WebSphere MQ baseline, the **LISA_HOME/lib** directory must contain various WebSphere MQ drivers.

WebSphere MQ 6

If you are using WebSphere MQ 6, copy the following JAR files into the **LISA_HOME/lib** directory:

- MQ_HOME/java/lib/com.ibm.mq.jar
- MQ_HOME/java/lib/connector.jar
- MQ_HOME/java/lib/dhbccore.jar

WebSphere MQ 7

If you are using WebSphere MQ 7, copy the following JAR files into the **LISA_HOME/lib** directory:

- MQ_HOME/java/lib/com.ibm.mq.commonservices.jar
- MQ_HOME/java/lib/com.ibm.mq.headers.jar
- MQ_HOME/java/lib/com.ibm.mq.jar
- MQ_HOME/java/lib/com.ibm.mq.jmqi.jar
- MQ_HOME/java/lib/connector.jar
- MQ_HOME/java/lib/dhbccore.jar

WebSphere MQ Baseline Properties

The LISA Agent enables you to specify configuration parameters in a file named **rules.properties**. For detailed information about this file, see the [LISA Agent Guide](#).

Before you create a WebSphere MQ baseline, you can add the following properties to the **rules.properties** file.

Queue Request Matches and Queue Response Matches



The **QUEUE_REQUEST_MATCHES** and **QUEUE_RESPONSE_MATCHES** properties are optional. They are needed only if the Agent cannot do a good enough job on its own determining whether a message is a request or a response.

The **QUEUE_REQUEST_MATCHES** and **QUEUE_RESPONSE_MATCHES** properties indicate which queues are for requests and which queues are for responses.

If you include these properties, then the property names must be followed by a colon (:) and the actual queue name. The value of each property is a period followed by an asterisk (.*). For example:

```
property key=QUEUE_REQUEST_MATCHES:ORDERS.REQUEST value=.*  
property key=QUEUE_RESPONSE_MATCHES:ORDERS.RESPONSE value=.*
```

Importing the WebSphere MQ Baseline Assets

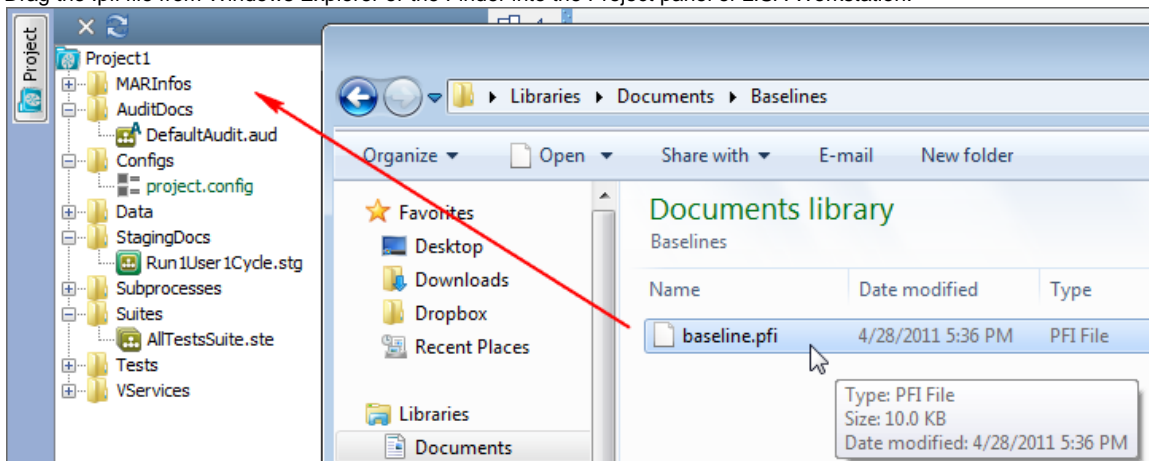
When you generate the WebSphere MQ baseline assets, the result is a file with the .pfi extension. You can import the file into LISA Workstation by using drag-and-drop.



As of release 6.0.6, you can also import the .pfi file by right-clicking the Tests folder and selecting Import Files.

To import the WebSphere MQ baseline assets

1. In LISA Workstation, open or create a project where you want to place the WebSphere MQ baseline test case or suite.
2. Open Windows Explorer or the Finder.
3. Locate the directory where you saved the .pfi file.
4. Change the size of Windows Explorer or the Finder so that you can also see the Project panel of LISA Workstation.
5. Drag the .pfi file from Windows Explorer or the Finder into the Project panel of LISA Workstation.



LISA Workstation extracts the assets from the .pfi file and adds them to the appropriate folders.

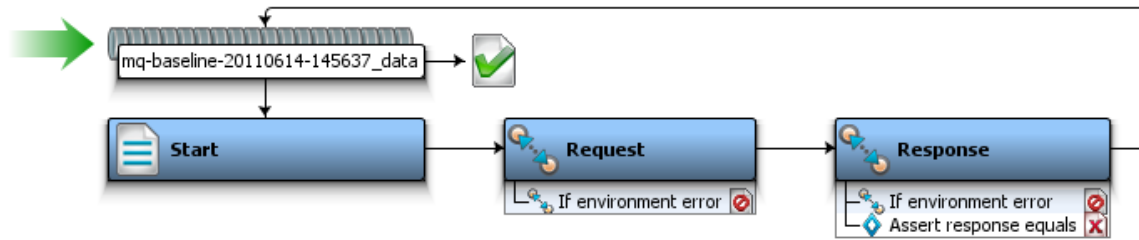
Viewing and Running WebSphere MQ Baseline Test Cases

The contents of a WebSphere MQ baseline test case depend on the validation mode.

Shallow Validation

If you select shallow validation in the [Baseline Wizard](#), then the WebSphere MQ baseline test case has the following steps:

- A do-nothing step that reads from a Large Data data set
- A WebSphere MQ step that sends the request
- A WebSphere MQ step that receives the response



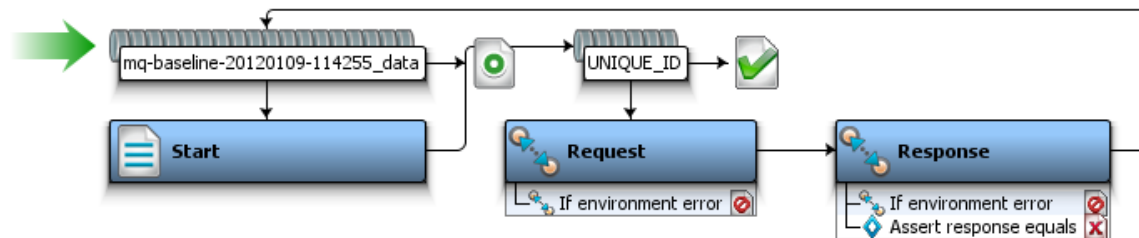
The Large Data data set contains information that changes for each transaction in the baseline. This information includes the request payload and the response payload.

In the WebSphere MQ steps, the client mode is set to native client.

The first WebSphere MQ step sends the request using data from the data set.

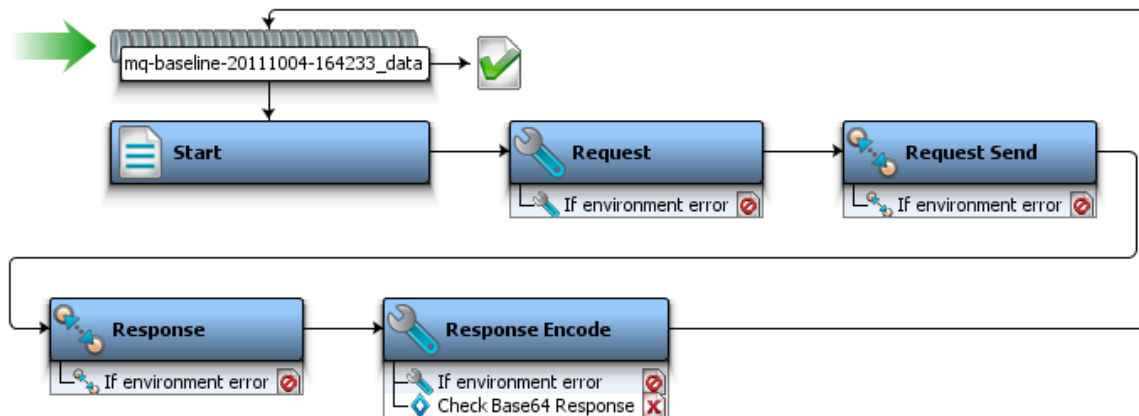
The second WebSphere MQ step receives the response. The step includes a Graphical XML Side-by-Side Comparison assertion with the name **Assert response equals**. When you run the baseline test case, this assertion verifies that the actual response matches the expected response. If the responses do not match, then the test fails.

The test case may also include a Message/Correlation ID Generator data set, which generates a 24-byte code that is used as the correlation ID.



Binary payloads affect the structure of a WebSphere MQ baseline test case. In this scenario, the test case has the following steps:

- A do-nothing step that reads from a Large Data data set
- A Java Script step that decodes the request payload from Base64 notation into an array of bytes
- A WebSphere MQ step that sends the request
- A WebSphere MQ step that receives the response
- A Java Script step that encodes the response payload from an array of bytes into Base64 notation



The Large Data data set includes two versions of the original request payload: a human-readable version and the Base64 representation. The Large Data data set also includes two versions of the original response payload: a human-readable version and the Base64 representation.

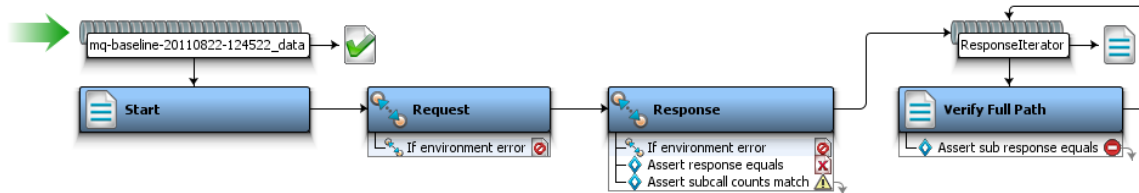
The second Java Script step includes an assertion with the name **Check Base64 Response**. When you run the baseline test case, this assertion verifies that the Base64 representation of the actual response matches the Base64 representation of the expected response. If the responses do not match, then the test fails.

Full Validation



With full validation, the service under test must be Agent enabled when you run the baseline test case. Otherwise, the test case cannot gather the downstream element response information to compare to the expected results.

If you select full validation in the [Baseline Wizard](#), then the WebSphere MQ baseline test case also has a do-nothing step with the name **Verify Full Path**.



This do-nothing step includes a Graphical XML Side-by-Side Comparison assertion with the name **Assert sub response equals**. When you run the baseline test case, this assertion verifies that the actual response for each subnode matches the expected response. A subnode is any node to the right of the node that was selected in the path graph at baseline creation time. If the responses do not match, then the test generates an error.

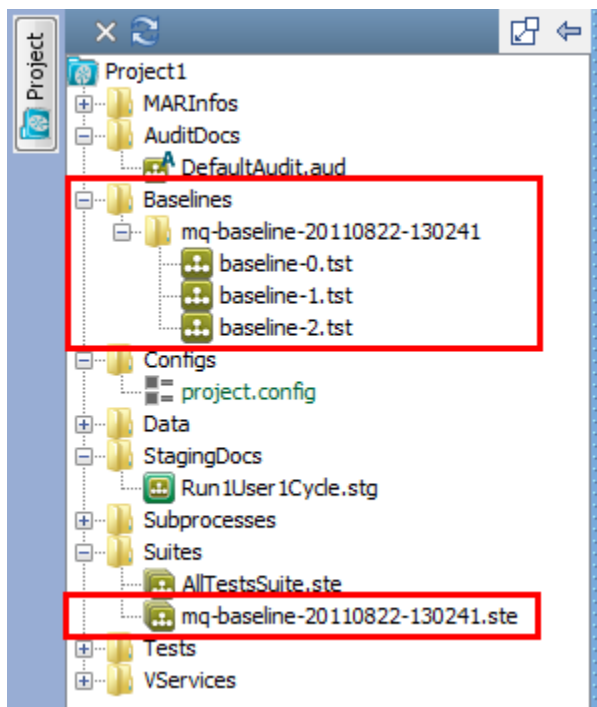
You must run the test case with a staging document in which the Enable LISA Pathfinder check box is selected. The **Run1User1CyclePF.stg** staging document, which is included by default in all projects, has this check box selected.

Viewing and Running WebSphere MQ Baseline Suites

The test cases in a WebSphere MQ baseline suite are located in the Baselines folder of the project, rather than the Tests folder.

The suite document is located in the usual place: the Suites folder.

The following image shows a project that contains a baseline suite. The Baselines folder and the suite document are highlighted.



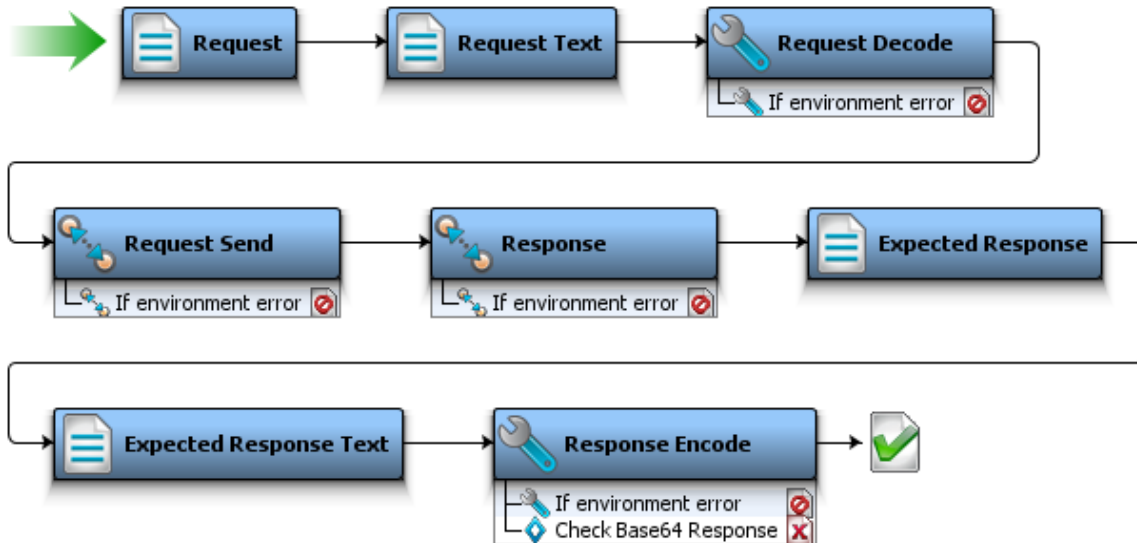
The test cases in a WebSphere MQ baseline suite are similar to [WebSphere MQ baseline test cases](#). However, they do not include the do-nothing step that reads from a Large Data data set. The request and response data are stored in the WebSphere MQ steps instead.

For binary payloads, the test cases have the following steps:

- A Parse Text as Response step that contains a Base64 representation of the original request payload
- An Output Log Message step that writes a human-readable version of the original request payload to the log file
- A Java Script step that decodes the actual request payload from Base64 notation into an array of bytes
- A WebSphere MQ step that sends the request

- A WebSphere MQ step that receives the response
- A Parse Text as Response step that contains a Base 64 representation of the original response payload
- An Output Log Message step that writes a human-readable version of the original response payload to the log file
- A Java Script step that encodes the actual response payload from an array of bytes into Base64 notation

The following image shows the test case for a binary payload.



If you chose full validation in the [Baseline Wizard](#), then you must run the suite with a staging document in which the Enable LISA Pathfinder check box is selected. The **Run1User1CyclePF.stg** staging document has this check box selected.

To run the baseline suite, follow the steps in [Running Test Suites](#). The result of each test case is displayed as a comparison between the expected response and the actual response.

The Baseline Results panel indicates whether each test succeeded (green) or failed (red). If a test fails, you can do either of the following actions:

- Select the test and click Ignore. The test will not be executed in future runs of the suite.
- Select the test and click Update. The suite is updated with the actual response that was received.

To run the suite again, click Rerun Suite.

To display the run statistics in the Reporting Console, click View Reports.

Baseline Wizard

The Baseline Wizard walks you through the process of creating a baseline test case or suite in the Pathfinder Console.

The number of steps in the wizard can vary, depending on the transaction type and on whether you select the Advanced Configuration check box in the initial step.

- [Configuration](#)
- [Upload Test Template](#)
- [Edit Test Template](#)
- [Edit Headers](#)
- [EJB Configuration](#)
- [Review](#)
- [Download Assets](#)

Baseline Wizard Step: Configuration

The Configuration step enables you to specify the baseline name, the validation mode, and the creation mode.

The following image shows the Configuration step.

JMS Baseline Wizard Step 1 of 2 : Configuration

Name:

Validation Mode: ☒ Shallow ☐ Full

Creation Mode: ☐ Consolidated ☒ Expanded

☐ Advanced Configuration

< Previous Next > Cancel

Name. Enter a name for the baseline test case or suite that will be generated. The name cannot include the following characters: less than sign, greater than sign, apostrophe, ampersand, question mark, pipe sign, asterisk, backslash, or forward slash.

Validation Mode. Specify the type of response validation that will be included.

- Shallow. The baseline will include an assertion that validates the response of the selected node.
- Full. The baseline will include an assertion that validates the response of the selected node, and an assertion that validates the response of each subnode. A subnode is any node to the right of the selected node in the path graph.

Creation Mode. Specify the types of assets that will be generated.

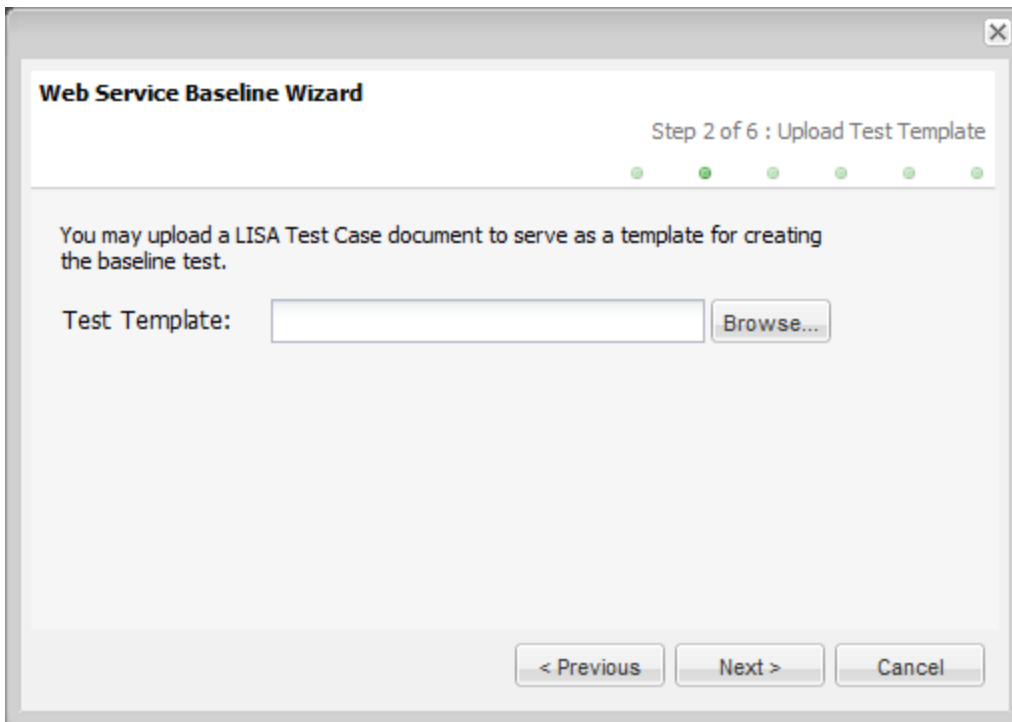
- Consolidated. The wizard will generate a single test case and a data set containing a row for each transaction.
- Expanded. The wizard will generate a suite of tests (one for each transaction) and a suite document for running the tests.

Advanced Configuration. Adds advanced steps to the wizard.

Baseline Wizard Step: Upload Test Template (Advanced)

The Upload Test Template step enables you to specify a test case to be used as a template for the baseline test case or suite.

The following image shows the Upload Test Template step.



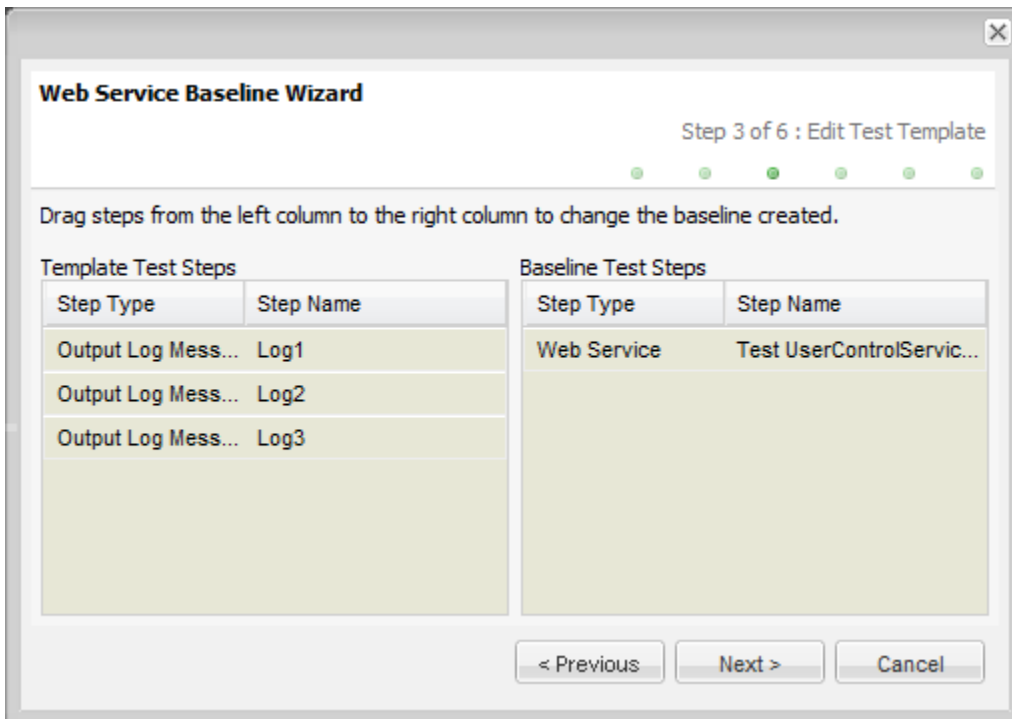
Click Browse to select the test case file.

Baseline Wizard Step: Edit Test Template (Advanced)

The Edit Test Template step enables you to select steps from the template test case and incorporate them into the baseline test case or suite.

The left panel contains the steps in the template test case. The right panel contains the steps in the baseline test case or suite. Move one or more steps from the left panel to the right panel.

The following image shows the Edit Test Template step.



Baseline Wizard Step: Edit Headers (Advanced)

The Edit Headers step enables you to view and edit the HTTP request headers. This step appears only for REST and web service baselines.

The following image shows the Edit Headers step.

Web Service Baseline Wizard

Step 4 of 6 : Edit Headers

Add Header

Name	Value
cache-control	no-cache
host	localhost:8080
lisaframeid	a01ccce0-773c-11e0-8853-0024d6ab5ce3
lisaframeremoteip	192.168.1.67
user-agent	Mozilla/4.0 (compatible; MSIE 6.0; Windo...
lisaframeroot	true
pragma	no-cache

< Previous Next > Cancel

If you want to add a header, click Add Header and enter the name and value.

If you want to delete a header, right-click the row and select Delete Header.

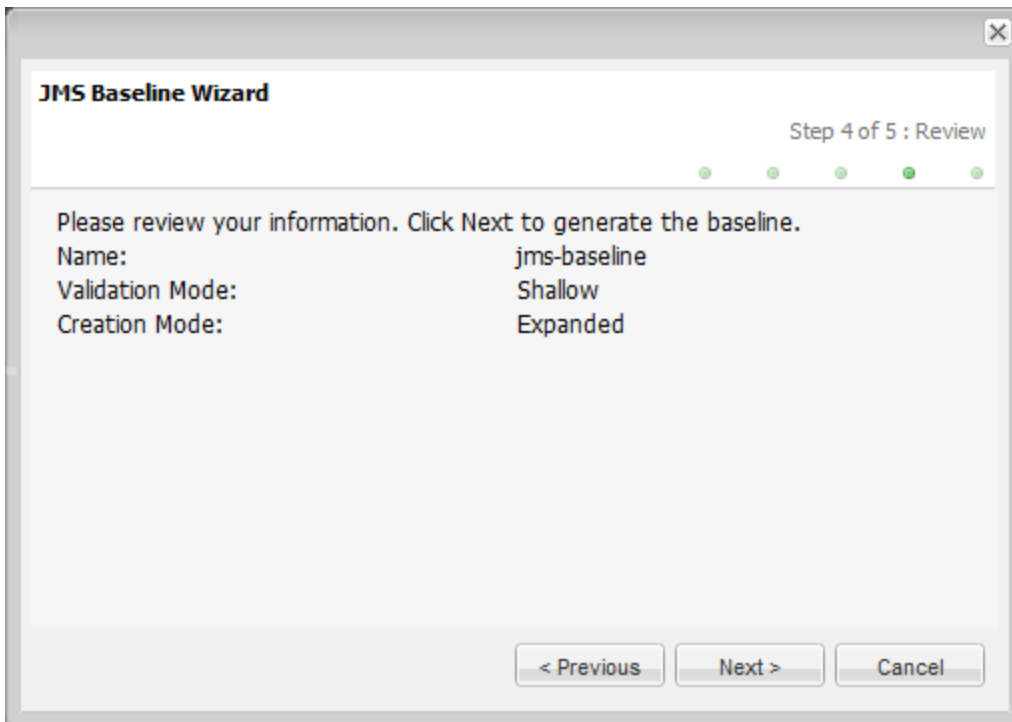
Baseline Wizard Step: EJB Configuration (Advanced)

TO BE DONE

Baseline Wizard Step: Review (Advanced)

The Review step contains a summary of information about the baseline test case or suite.

The following image shows the Review step.



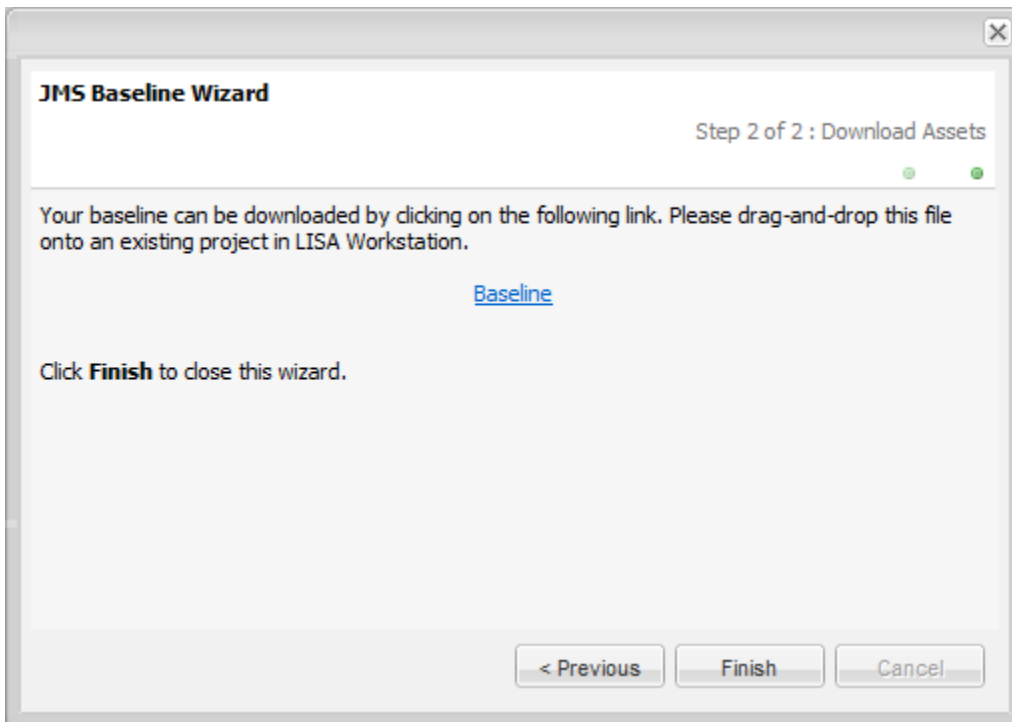
If you want to change any of the settings, click Previous until you reach the settings that you want to change.

When you are ready to generate the test case or suite, click Next.

Baseline Wizard Step: Download Assets

Click the link and download the .pfi file, which contains the assets for the baseline test case or suite.

The following image shows the Download Assets step.



Creating Virtual Services from Transactions



This feature requires a separate license.

Pathfinder enables you to create a virtual service based on a set of actual transactions.

For detailed information about service virtualization, see the [Virtualize Guide](#).

The following transaction types are supported:

- [JMS](#)
- [REST](#)
- [Web Service](#)
- [WebSphere MQ](#)

Creating Virtual Services from JMS Transactions

The result of the first procedure is a raw traffic file that can be imported into the [Virtual Service Image Recorder](#). The benefit of this approach is that it eliminates the need to do proxy recording.

In addition to the request and response bodies, the raw traffic file contains all the connection and queue information in metadata and attributes.

The first procedure also creates a virtual service model. However, you will not use the virtual service model, because it does not have any data protocols. Instead, you will create a new virtual service model with a data protocol in the second procedure.



This feature is supported for configurations in which JNDI is used to obtain the JMS connection factory. This feature is also supported for configurations in which a direct API from IBM WebSphere MQ is used to obtain the JMS connection factory.

To create the raw traffic file from JMS transactions

1. In the Pathfinder Console, make sure that all the transactions that you want to include are viewable in the **Paths** tab. If necessary, change the [filter criteria](#). It does not matter if there are too many transactions to be viewed on one page.
2. In the **Paths** tab, select a transaction that has a value of **jms** in the Category column.
3. In the path graph, select a request node.
4. Click the Virtualize icon at the bottom of the Pathfinder Console. The Create Virtual Service dialog opens.

Create Virtual Service

Please provide the following information:

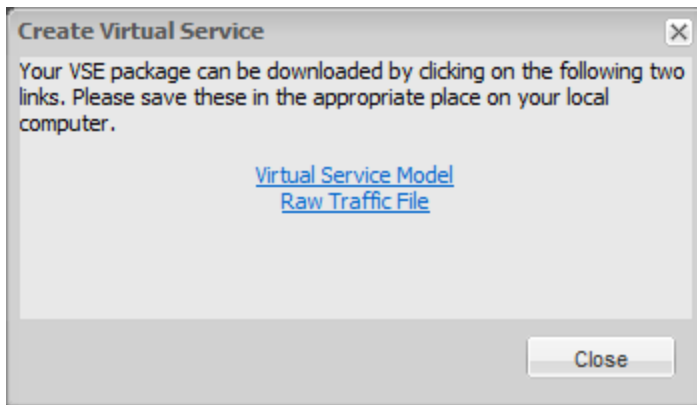
Name:

Create

Close

5. Enter a name for the raw traffic file to be created.
6. Click Create.

The dialog displays two links: Virtual Service Model and Raw Traffic File. You can ignore the Virtual Service Model link.



7. Click the Raw Traffic File link and save the file. You will import this file in the next procedure.
8. Click Close.

To create the service image and virtual service model

1. Go to LISA Workstation.
2. From the main menu, select File > New > VS Image > By recording.
The [Virtual Service Image Recorder](#) appears.
3. Do the following:
 - a. In the Write image to field, enter the fully qualified name of the service image that will be created (for example, **C:\Lisa\Projects\Project1\VSservices\Images\newimage.vsi**).
 - b. In the Import traffic field, browse to and select the raw traffic file.
 - c. In the Transport protocol field, select Standard JMS.
 - d. In the Model file field, enter the fully qualified name of the virtual service model that will be created (for example, **C:\Lisa\Projects\Project1\VSservices\newmodel.vsm**).
 - e. Click Next.
The data protocols step appears.
4. Do the following:
 - a. In the Request Side Data Protocols area, click the plus sign.
 - b. Click the left column of the newly added row and select the appropriate [data protocol](#). For XML-based applications, [Generic XML Payload Parser](#) is a good generic choice.
 - c. If the application's responses are not in an XML or text format, then a response-side data protocol might be required for the VSE engine to perform [magic string](#) substitutions on the response.
 - d. Click Next.
The next step prompts you to select the message recording style.
5. If you want to review the request and response queue information, then do the following:
 - a. Select the Review the destinations and transaction tracking mode check box.
 - b. If the correlation scheme in the Correlation drop-down list is incorrect, change the value.
 - c. Click Next.
The next step contains a Destination Info tab and a Connection Setup tab.
 - d. Pathfinder automatically populates the values in the Destination Info and Connection Setup tabs. The Proxy Destination field is set to N/A because you are not doing proxy recording. You should not need to make changes in either tab, but you can do so if Pathfinder did not set a correct value. Click Next.
The next step contains response information.
 - e. Pathfinder automatically populates the values in this step. The Response Destinations area contains one or more response queues. You should not need to make changes, but you can do so if Pathfinder did not set a correct value. Click Next.
The transactions are imported from the raw traffic file.
6. The next step or steps that appear (if any) depend on the data protocol that you selected. For example, if you selected the Generic XML Payload Parser data protocol, then the next step prompts you to create XPath expressions to form VSE requests. Using the [Virtualize Guide](#) as needed, complete the step or steps.
The last step indicates that the recorder is performing post-processing on what has been recorded.
7. Click Finish.
The transactions are saved to a service image, and the virtual service model is created.

To run the virtual service

1. Shut down the original service.
2. Go to LISA Workstation and [deploy the virtual service model](#) that you created in the second procedure.
3. Run a client application with the virtual service model as the service.

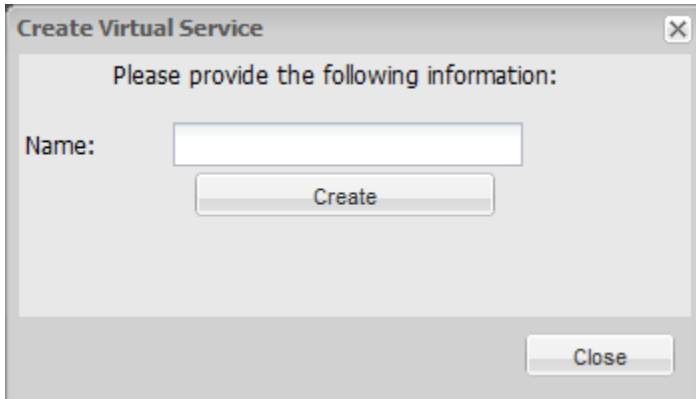
Creating Virtual Services from REST Transactions

TO BE DONE

Creating Virtual Services from Web Service Transactions

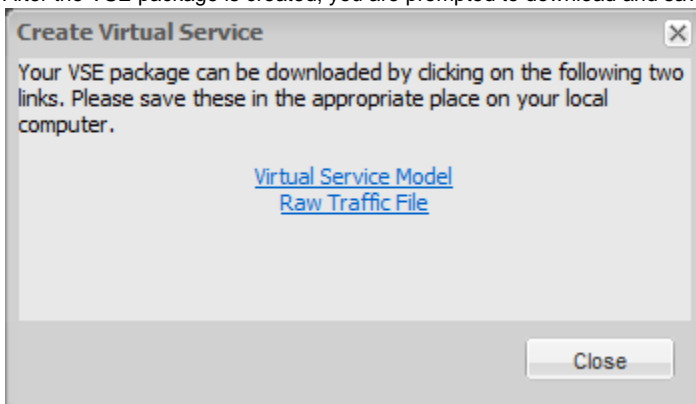
To create a virtual service from a web service transaction

1. Open the Pathfinder Console.
2. In the **Paths** tab, select a transaction.
3. In the path graph, select a node.
4. Click the Virtualize icon at the bottom of the console. The Create Virtual Service dialog opens.



The dialog box is titled "Create Virtual Service" and contains the text "Please provide the following information:". Below this text is a label "Name:" followed by a text input field. Under the input field is a "Create" button. At the bottom right of the dialog is a "Close" button.

5. Enter a name for the virtual service model and raw traffic file to be created.
 6. Click Create.
- After the VSE package is created, you are prompted to download and save the package.



The dialog box is titled "Create Virtual Service" and contains the text "Your VSE package can be downloaded by clicking on the following two links. Please save these in the appropriate place on your local computer." Below this text are two blue, underlined links: "Virtual Service Model" and "Raw Traffic File". At the bottom right of the dialog is a "Close" button.

7. Click the Virtual Service Model link and save the .vsm file to the VServices project directory.
8. Click the Raw Traffic File link and save the .xml file to the VServices project directory.

Creating Virtual Services from WebSphere MQ Transactions

The result of the first procedure is a raw traffic file that can be imported into the [Virtual Service Image Recorder](#). The benefit of this approach is that it eliminates the need to do proxy recording.

In addition to the request and response bodies, the raw traffic file contains all the connection and queue information in metadata and attributes.

The first procedure also creates a virtual service model. However, you will not use the virtual service model, because it does not have any data protocols. Instead, you will create a new virtual service model with a data protocol in the second procedure.

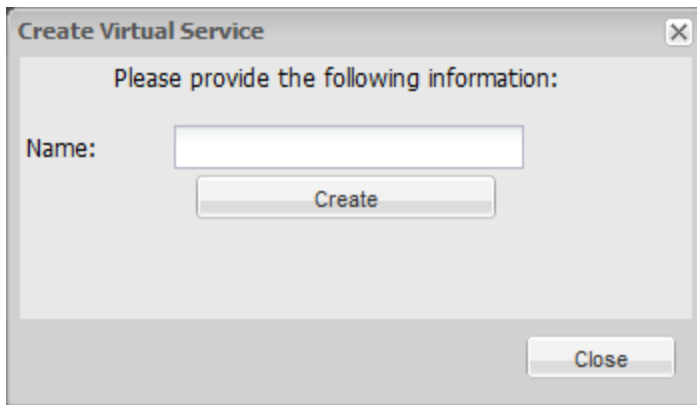
If you are working with binary payloads, then you must use an extension data protocol handler to convert the binary requests into an XML form that the VSE engine can handle. You select this data protocol handler in the Virtual Service Image Recorder.

If the response is also binary, then you have two options:

- Use an extension data protocol handler to parse the binary responses into a text format for the service image. At runtime, the original binary format is reconstructed for each response.
- Do not use an extension data protocol handler for the response. At runtime, the VSE engine returns the original binary response and will not be able to perform magic string substitutions.

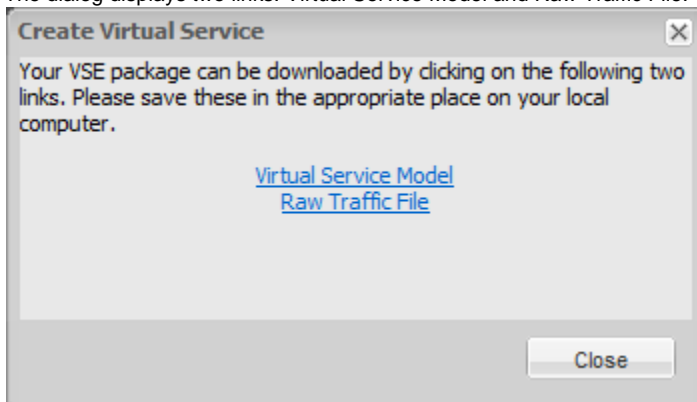
To create the raw traffic file from WebSphere MQ transactions

1. In the Pathfinder Console, make sure that all the transactions that you want to include are viewable in the **Paths** tab. If necessary, change the [filter criteria](#). It does not matter if there are too many transactions to be viewed on one page.
2. In the **Paths** tab, select a transaction that has a value of **ibmmq** in the Category column.
3. In the path graph, select a request node.
4. Click the **Virtualize** icon at the bottom of the Pathfinder Console. The Create Virtual Service dialog opens.



5. Enter a name for the raw traffic file to be created.
6. Click Create.

The dialog displays two links: Virtual Service Model and Raw Traffic File. You can ignore the Virtual Service Model link.



7. Click the Raw Traffic File link and save the file. You will import this file in the next procedure.
8. Click Close.

To create the service image and virtual service model

1. Go to LISA Workstation.
2. From the main menu, select File > New > VS Image > By recording.
The [Virtual Service Image Recorder](#) appears.
3. Do the following:
 - a. In the Write image to field, enter the fully qualified name of the service image that will be created (for example, **C:\Lisa\Projects\Project1\VSImages\newimage.vsi**).
 - b. In the Import traffic field, browse to and select the raw traffic file.
 - c. In the Transport protocol field, select IBM MQ Series.
 - d. In the Model file field, enter the fully qualified name of the virtual service model that will be created (for example, **C:\Lisa\Projects\Project1\VSImages\newmodel.vsm**).
 - e. Click Next.
The data protocols step appears.
4. Do the following:
 - a. In the Request Side Data Protocols area, click the plus sign.
 - b. Click the left column of the newly added row and select the appropriate [data protocol](#). For XML-based applications, [Generic XML Payload Parser](#) is a good generic choice.
 - c. If the application's responses are not in an XML or text format, then a response-side data protocol might be required for the VSE engine to perform [magic string](#) substitutions on the response.
 - d. Click Next.
The next step prompts you to select the message recording style.
5. If you want to review the request and response queue information, then do the following:
 - a. Select the Review the destinations and transaction tracking mode check box.
 - b. If the correlation scheme in the Correlation drop-down list is incorrect, change the value.
 - c. Click Next.
The next step contains a Destination Info tab and a Connection Setup tab.
 - d. Pathfinder automatically populates the values in the Destination Info and Connection Setup tabs. The Proxy Destination field is set to N/A because you are not doing proxy recording. You should not need to make changes in either tab, but you can do so if Pathfinder did not set a correct value. Click Next.
The next step contains response information.
 - e. Pathfinder automatically populates the values in this step. The Response Destinations area contains one or more response queues. You should not need to make changes, but you can do so if Pathfinder did not set a correct value. Click Next.
The transactions are imported from the raw traffic file.
6. The next step or steps that appear (if any) depend on the data protocol that you selected. For example, if you selected the Generic XML Payload Parser data protocol, then the next step prompts you to create XPath's to form VSE requests. Using the [Virtualize Guide](#) as

needed, complete the step or steps.

The last step indicates that the recorder is performing post-processing on what has been recorded.

7. Click Finish.

The transactions are saved to a service image, and the virtual service model is created.

To run the virtual service

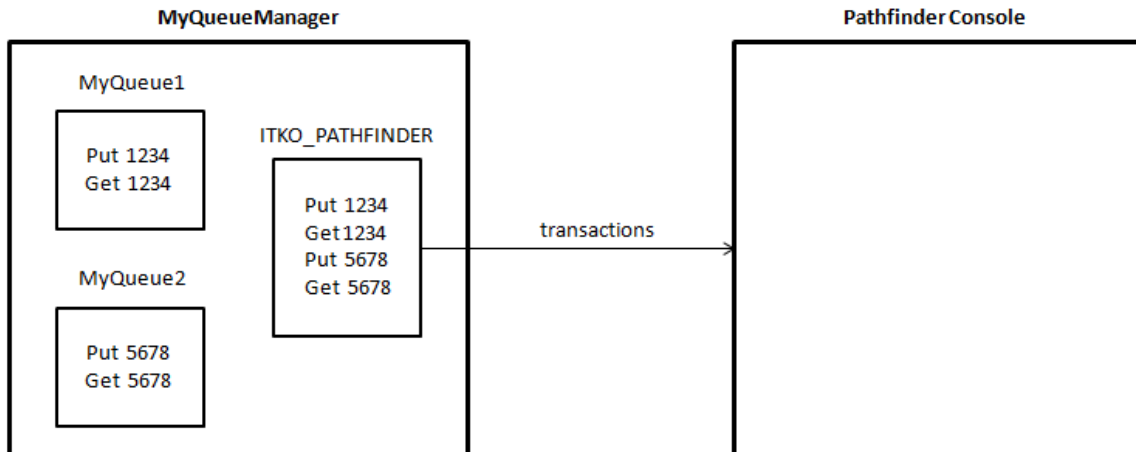
1. Shut down the original service.
2. Go to LISA Workstation and [deploy the virtual service model](#) that you created in the second procedure.
3. Run a client application with the virtual service model as the service.

Native MQ Pathfinder

Native MQ Pathfinder is a WebSphere MQ API exit that monitors the get and put calls for the queues of a queue manager. Native MQ Pathfinder copies the get and put calls to a special queue named ITKO_PATHFINDER. A separate LISA component uses the data in the ITKO_PATHFINDER queue to assemble transactions that can be viewed in the Pathfinder Console. You can then use the transactions to create baselines, raw traffic files, and so on.

This feature is supported on WebSphere MQ 6.0 and 7.0.

The following diagram shows an example of how Native MQ Pathfinder works. The queue manager has three queues: MyQueue1, MyQueue2, and ITKO_PATHFINDER. The ITKO_PATHFINDER queue contains copies of the get and put calls for MyQueue1 and MyQueue2. The data is converted into transactions and sent to the Pathfinder Console.



Native MQ Pathfinder ignores certain types of queues that are not relevant to LISA. For example, any queue that has a name beginning with **SYSTEM.** is ignored.

Both types of put calls are supported: MQPUT and MQPUT1.

The following topics describe how to use Native MQ Pathfinder:

- [Creating the ITKO_PATHFINDER Queue](#)
- [Installing the Native MQ Pathfinder API Exit](#)
- [Configuring the Native MQ Pathfinder API Exit](#)
- [Configuring the Agent Properties for Native MQ Pathfinder](#)
- [Generating Transactions from the ITKO_PATHFINDER Queue](#)

Creating the ITKO_PATHFINDER Queue

Native MQ Pathfinder places copies of get and put calls in a queue called ITKO_PATHFINDER.

If the queue fills up, then the oldest message is deleted.

To create the ITKO_PATHFINDER queue

1. Go to WebSphere MQ Explorer.
2. Create a queue with the name ITKO_PATHFINDER. This queue must be owned by the same queue manager as the queues that you want to monitor. The persistence setting, maximum queue depth, and so on are up to you.

Installing the Native MQ Pathfinder API Exit

The NativeMQ_Pathfinder.zip file contains the API exit files for various operating systems.

To install the Native MQ Pathfinder API exit

1. Shut down WebSphere MQ.
2. Download the NativeMQ_Pathfinder.zip file from [Agent Downloads](#) and extract the contents.
3. (Windows) Copy the iTKO_MQ_Pathfinder.dll file to the MQ_HOME/exits directory.
4. (Linux) Copy the iTKO_MQ_Pathfinder_linux and iTKO_MQ_Pathfinder_linux_r files to the /var/mqm/exits directory. The files must be owned by the mqm user. The files must have the read and execute bits set (chmod a+rx).
5. (Solaris) Copy the iTKO_MQ_Pathfinder_sol_sparc file to the /var/mqm/exits directory. The file must be owned by the mqm user. The file must have the read and execute bits set (chmod a+rx).
6. (Solaris 64) Copy the iTKO_MQ_Pathfinder_sol_sparc.64 file to the /var/mqm/exits64 directory and remove the .64 extension. The file must be owned by the mqm user. The file must have the read and execute bits set (chmod a+rx).
7. Restart WebSphere MQ.

Configuring the Native MQ Pathfinder API Exit

After you install the API exit, you must provide information such as the module and the entry point.

To configure the Native MQ Pathfinder API exit

1. Go to WebSphere MQ Explorer or open the qm.ini configuration file.
2. Configure the queue manager to run an API exit. Set the Name attribute to any descriptive name. Set the Function attribute to **EntryPoint**. This value is case sensitive. Set the Module attribute to the API exit file that you installed (for example, /var/mqm/exits/iTKO_MQ_Pathfinder_linux). If you want Native MQ Pathfinder to generate log files, then set the Data attribute to the directory where the log files will be written.
3. Restart the queue manager.
4. Send a test message to a queue.
5. Browse the iTKO_PATHFINDER queue and verify that the test message was mirrored.

Configuring the Agent Properties for Native MQ Pathfinder

The rules.properties file for the LISA Agent includes a set of Native MQ Pathfinder properties that you must configure.

By default, Native MQ Pathfinder monitors all the queues of a queue manager. You can use the MQMIRROR_INCLUDE_QUEUES and MQMIRROR_EXCLUDE_QUEUES properties to override this behavior. The property values are regular expressions. The MQMIRROR_EXCLUDE_QUEUES property takes precedence over the MQMIRROR_INCLUDE_QUEUES property. If you want to specify a queue name that contains a dot character, then you must place a backslash immediately before the dot character. If you want to use a dot character as a regular expression construct, then you do not need to include a backslash.

If the system under test uses an uncommon pattern, then Native MQ Pathfinder might need help determining which queue/message is a request and which queue/message is a response. In this scenario, try adding the optional QUEUE_REQUEST_MATCHES and QUEUE_RESPONSE_MATCHES properties to the rules.properties file. Set the values to regular expressions. The regular expressions are evaluated against the message payload. For example:

```
property key=QUEUE_REQUEST_MATCHES:SOME_REQUEST_QUEUE value=.*
property key=QUEUE_RESPONSE_MATCHES:SOME_RESPONSE_QUEUE value=.*<response>.*
```

To configure the Agent properties for Native MQ Pathfinder

1. Open the rules.properties file in the LISA_HOME/agent directory.
2. Set the MQMIRROR_HOST property to the IP address or hostname of the server where WebSphere MQ is running.
3. Set the MQMIRROR_PORT property to the port number on which WebSphere MQ is listening for connection requests.
4. Set the MQMIRROR_CHANNEL property to the WebSphere MQ channel for connecting to the queue manager.
5. Set the MQMIRROR_QMNAME property to the queue manager that owns the queues you want to monitor.
6. Ensure that the MQMIRROR_QNAME property is set to iTKO_PATHFINDER. This value is the only valid value.
7. If you need a user and password to access WebSphere MQ, then uncomment the MQMIRROR_USER and MQMIRROR_PASSWD properties and set the values.
8. (Optional) Change the default value of the MQMIRROR_CONNECT_INTERVAL property.
9. If you want to specify which queues to include, then uncomment the MQMIRROR_INCLUDE_QUEUES property and set the value.
10. If you want to specify which queues to exclude, then uncomment the MQMIRROR_EXCLUDE_QUEUES property and set the value.
11. Save the rules.properties file.

Generating Transactions from the iTKO_PATHFINDER Queue

The LisaAgent.jar file includes an option for creating transactions based on the messages that have been mirrored to the iTKO_PATHFINDER queue.

If the broker is running on another computer, then you must also specify the -u option with the URL to the broker.

To generate transactions from the iTKO_PATHFINDER queue

- Run the following command:

```
java -jar LisaAgent.jar -m
```

The messages are retrieved from the ITKO_PATHFINDER queue and assembled into transactions. You can view the transactions in the Pathfinder Console.