Rational Integration Tester

Getting Started Guide

Version 8.0.1

Note

Before using this information and the product it supports, read the information in “Notices” on page 79.

This edition applies to version 8.0.1 of Rational Integration Tester and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

About this Publication

Contents

This guide provides a brief introduction to IBM® Rational® Integration Tester. It demonstrates some of the basic features of Rational Integration Tester, including basic GUI elements, synchronizing with WSDLs, and validating data.

Examples in this guide are described using SOAP/HTTP (it is not possible to show examples using every supported messaging product). Regardless of the messaging used in your environment, the basic principles are the same and you should be able to relate the examples to your own environment.

2.1 Intended Audience

This document is intended to be read by those with a fair understanding and exposure to the concepts involved in both testing and development and in enterprise integration. It is aimed at first-time users of IBM Rational Integration Tester, and those who are interested in its capabilities and ease of use. No prior knowledge of Rational Integration Tester is assumed.

2.2 Scope

This document provides a quick introduction to Rational Integration Tester. (For information about installing Rational Integration Tester, refer to *IBM Rational Integration Tester Installation Guide*.)

Detailed descriptions of Rational Integration Tester features, syntax, and functionality can be found in *IBM Rational Integration Tester Reference Guide*. More detailed examples for specific messaging systems can be found the various plugin guides that are provided with Rational Integration Tester.

2.3 Typographical Conventions

The following typographical conventions are observed throughout this document:

Type	Usage
`Constant Width`	Program output, listings of code examples, file names, commands, options, configuration file parameters, and literal programming elements in running text.
Italic	Document title names in statements that refer you to other documents. Also used to highlight concepts when first introduced.
Bold	Menu items in graphical user interface windows (such as Microsoft Windows-based or UNIX X Window applications) from which you select options or execute macros and functions. Submenus and options of a menu item are indicated with a “greater than” sign, such as **Menu > Submenu** or **Menu > Option**.

2.4 Contacting IBM Support

To contact IBM Support, see: www.ibm.com/contact/us/en/

Getting Started

Contents

This chapter describes how to start using Rational Integration Tester, including some prerequisite tasks.

1.1 Prerequisites

Before you can start using Rational Integration Tester, the product must be installed and the project database must be created – IBM DB2, Microsoft SQL Server, MySQL, and Oracle are supported.

For more information about both of these tasks, refer to *IBM Rational Integration Tester Installation Guide.*

1.2 Starting Rational Integration Tester

On a computer running Microsoft Windows, you can use any of the following methods to launch Rational Integration Tester:

- Click **Start > (All) Programs > IBM Rational Integration Tester**.
- Double-click the **Rational Integration Tester** shortcut on the desktop.
- On the Run dialog box (click **Start > Run**):
  - Enter the full path to the Rational Integration Tester application (for example, `C:\Program Files\IBM\RationalIntegrationTester\ GHTester.exe`) and click **OK**.

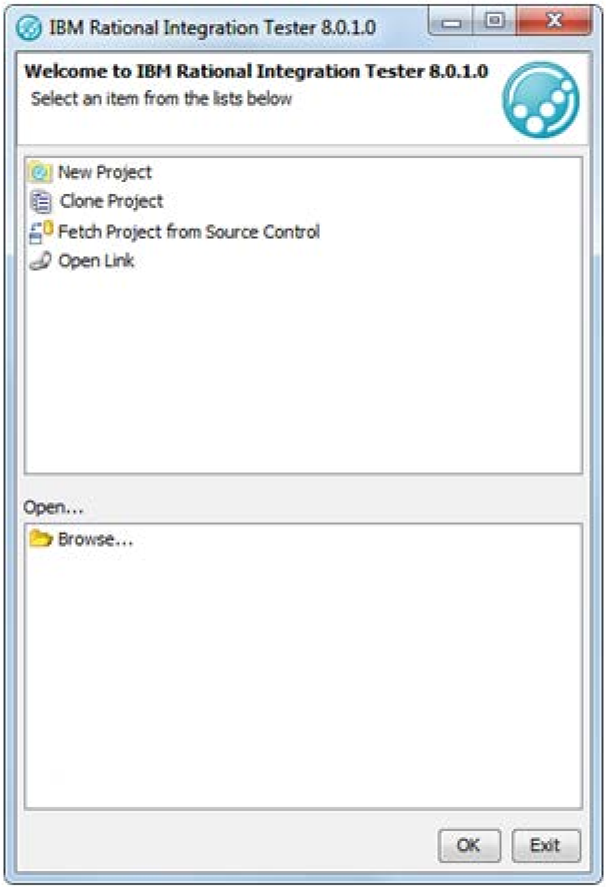
    Or

  - Click **Browse** to locate the Rational Integration Tester application, select it, then click **OK**.
- Open a command prompt and:
  - Enter the full path to the Rational Integration Tester application, for example: `C:\Program Files\IBM\RationalIntegrationTester\ghtester.exe`

    Or

  - Change to the Rational Integration Tester installation directory and enter `ghtester.exe` (or just `ghtester`).
- Using Windows Explorer, browse to the Rational Integration Tester installation directory and double-click **GHTester.exe**.

When Rational Integration Tester starts, the Welcome screen is displayed.



IBM Rational Integration Tester 8.0.1.0

Welcome to IBM Rational Integration Tester 8.0.1.0

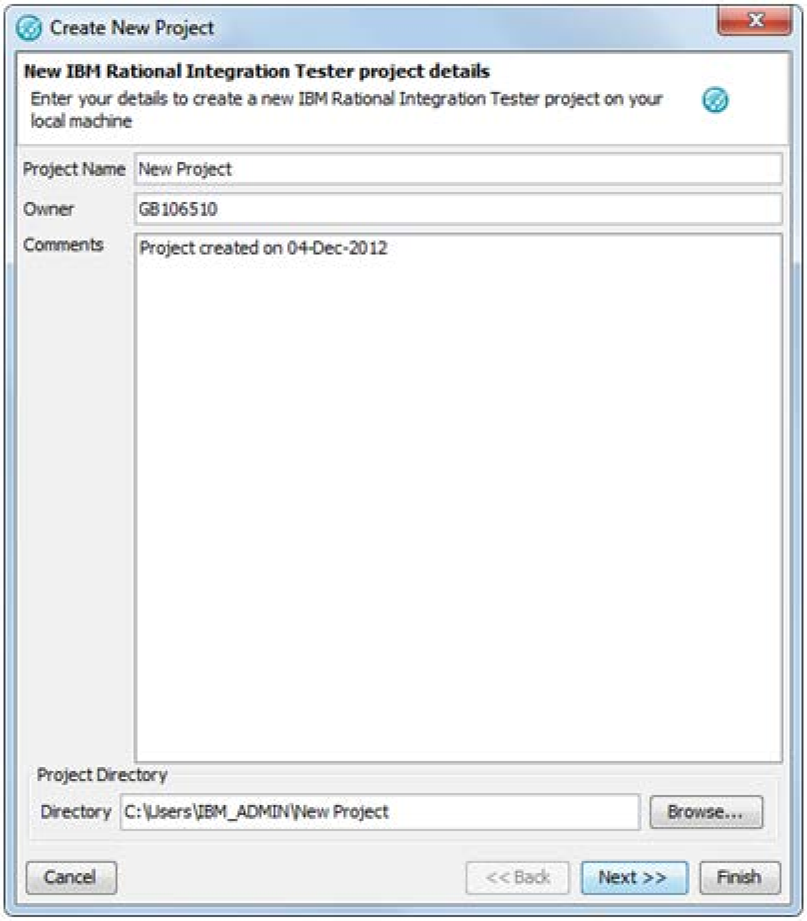
Select an item from the lists below

1.3 Creating a New Project

To create a new Rational Integration Tester project:

1. On the Welcome screen, click **New Project**.
2. Click **OK**.

   The Create New Project wizard is launched.



New IBM Rational Integration Tester project details

Enter your details to create a new IBM Rational Integration Tester project on your local machine

Project Name: New Project

Owner: GB106510

Comments: Project created on 04-Dec-2012

Project Directory

Directory: C:\Users\IBM_ADMIN\New Project

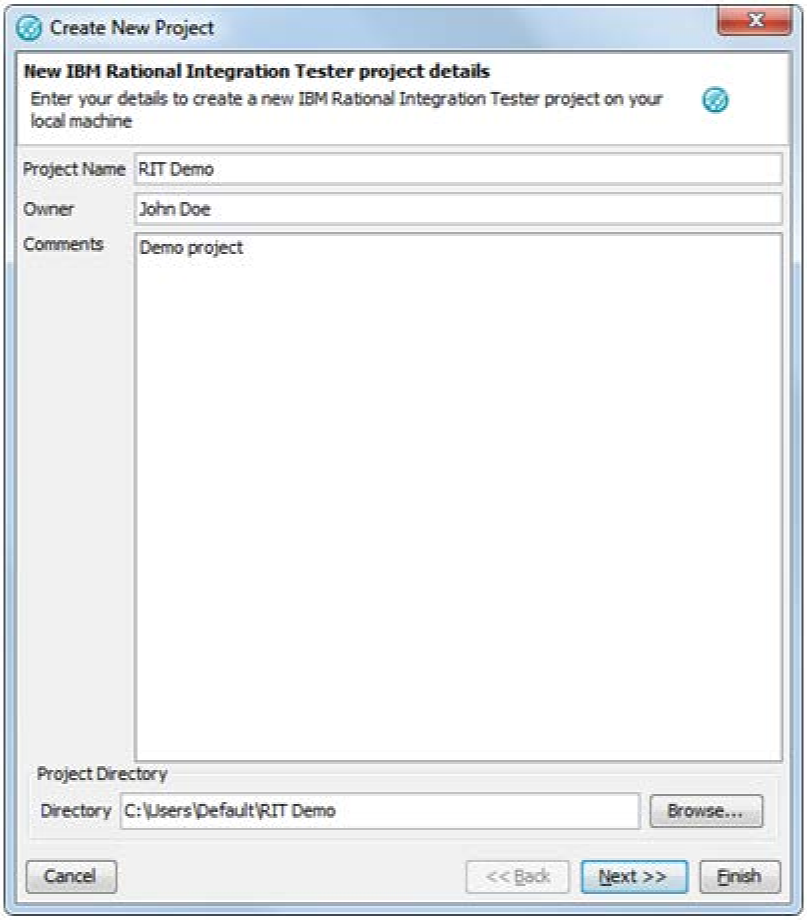
3. In the **Project Name** field, enter a name for the new project.

   For example, for this guided tour of Rational Integration Tester, you could create a project called `RIT Demo`.

4. **Optional:** In the **Owner** field, change the name of the default project owner if desired.
5. **Optional:** In the **Comments** field, enter any desired comments.

The field defaults to showing the creation date of the project.

6. **Optional:** In the **Directory** field, change the project directory, which does not have to have the same name as the project.



Create New Project

New IBM Rational Integration Tester project details

Enter your details to create a new IBM Rational Integration Tester project on your local machine

Project Name: RIT Demo

Owner: John Doe

Comments: Demo project

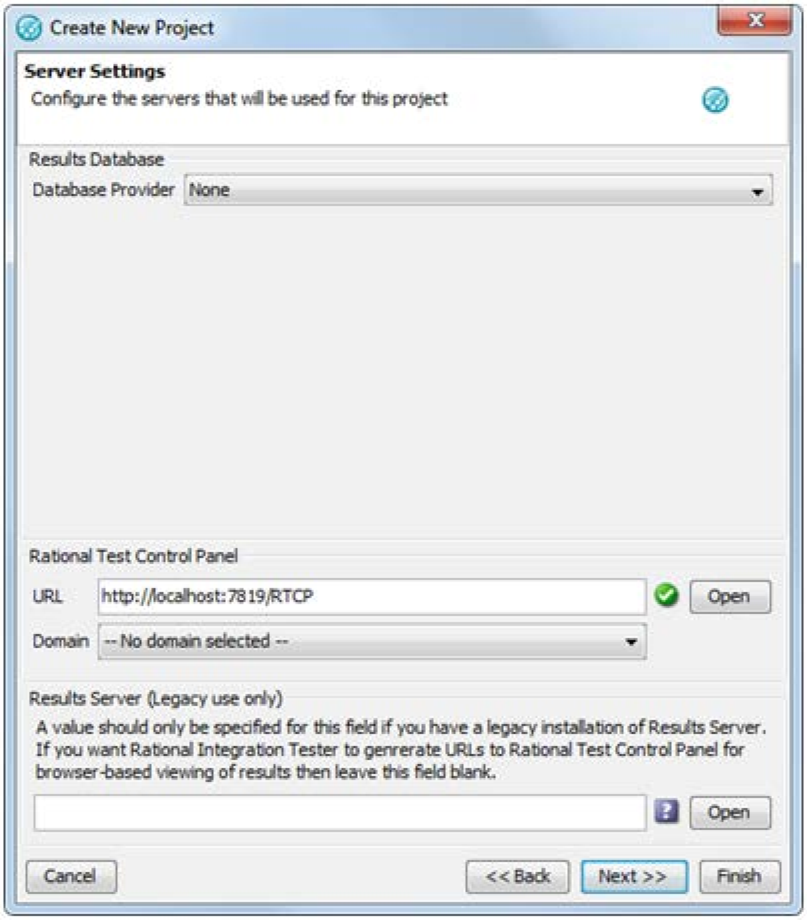
Project Directory

Directory: C:\Users\Default\RIT Demo

NOTE: If the directory does not exist, you will be prompted to create it. It is possible to store the project on a network drive but doing so might affect the overall performance of Rational Integration Tester adversely.

7. Click **Next**.

The Server Settings screen of the Create New Project wizard is displayed.



Server Settings

Configure the servers that will be used for this project

Results Database

Database Provider: None

Rational Test Control Panel

URL: http://localhost:7819/RTCP

Domain: -- No domain selected --

Results Server (Legacy use only)

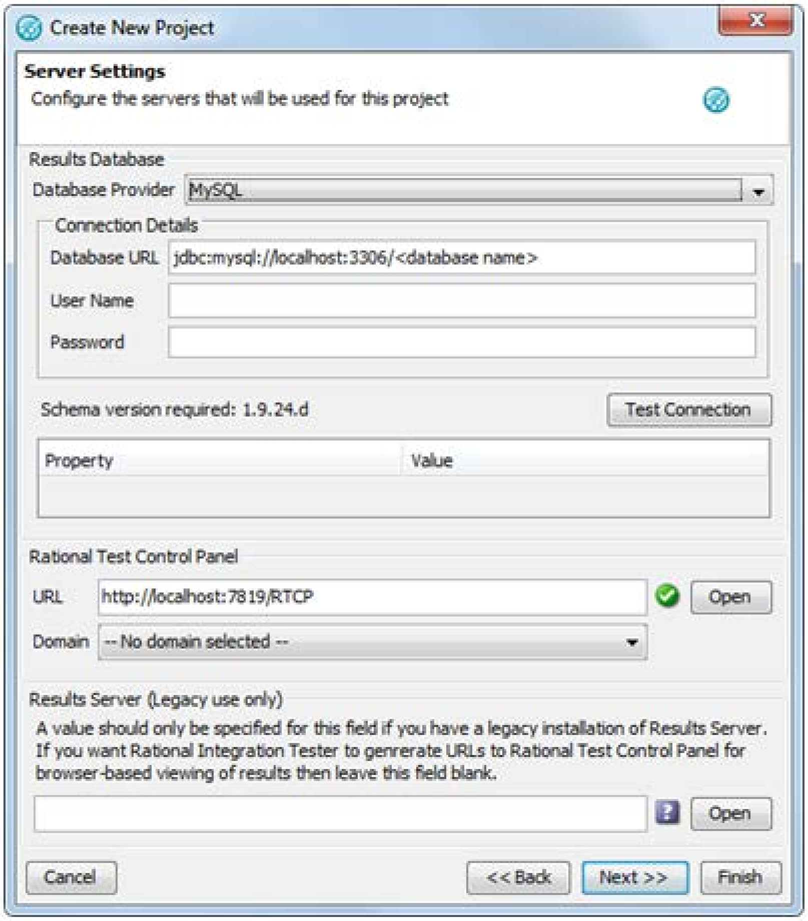
A value should only be specified for this field if you have a legacy installation of Results Server. If you want Rational Integration Tester to genrerate URLs to Rational Test Control Panel for browser-based viewing of results then leave this field blank.

At this point, only the **Results Database** settings on the upper half of the Server Settings screen are important because configuring a connection to the project database is necessary **if** you want to view the output of any tests that you will create and run. Otherwise, you can ignore the **Database Provider** list.

The **Rational Test Control Panel** and **Results Server** settings on the Server Settings screen can be configured later.

8. **Conditional:** In the **Database Provider** list, click the database management system that you want to use.

   Additional fields are displayed on the Server Settings screen.



Create New Project

Server Settings

Configure the servers that will be used for this project

Results Database

Database Provider: MySQL

Connection Details

Database URL: jdbc:mysql://localhost:3306/<database name>

User Name:

Password:

Schema version required: 1.9.24.d

Property	Value

Rational Test Control Panel

URL: http://localhost:7819/RTCP

Domain: -- No domain selected --

Results Server (Legacy use only)

A value should only be specified for this field if you have a legacy installation of Results Server. If you want Rational Integration Tester to genrerate URLs to Rational Test Control Panel for browser-based viewing of results then leave this field blank.

NOTE: For comprehensive information about creating a project results database, refer to *IBM Rational Integration Tester Installation Guide.*

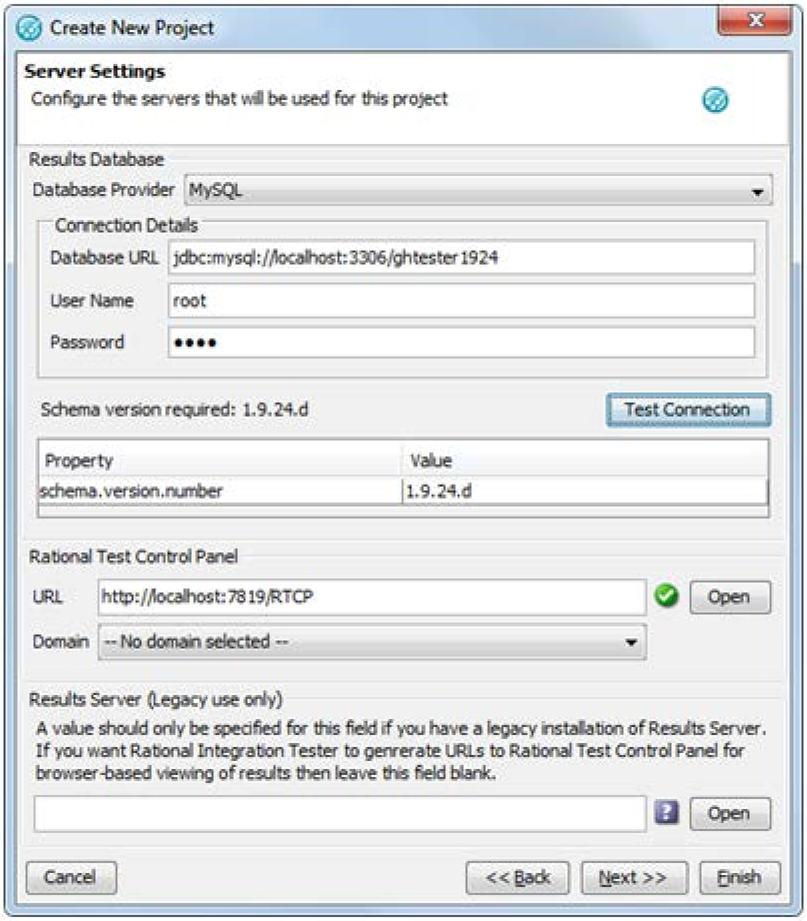
9. **Conditional:** In the **Database URL** field, enter the address of the database.

   For example, if using MySQL, the address might be `jdbc:mysql://localhost:3306/ghtester1924`.

10. **Conditional:** In the **User Name** and **Password** fields, enter the user name and password (if any) for the database.

11. Click **Test Connection** to verify that the connection to the database works correctly.

    If the test is successful, a confirmation message is displayed and the schema version number property and its value are displayed under **Property** and **Value**, respectively.



Create New Project

Server Settings

Configure the servers that will be used for this project

Results Database

Database Provider: MySQL

Connection Details

Database URL: jdbc:mysql://localhost:3306/ghtester1924

User Name: root

Password: ••••

Schema version required: 1.9.24.d

Property	Value
schema.version.number	1.9.24.d

Rational Test Control Panel

URL: http://localhost:7819/RTCP

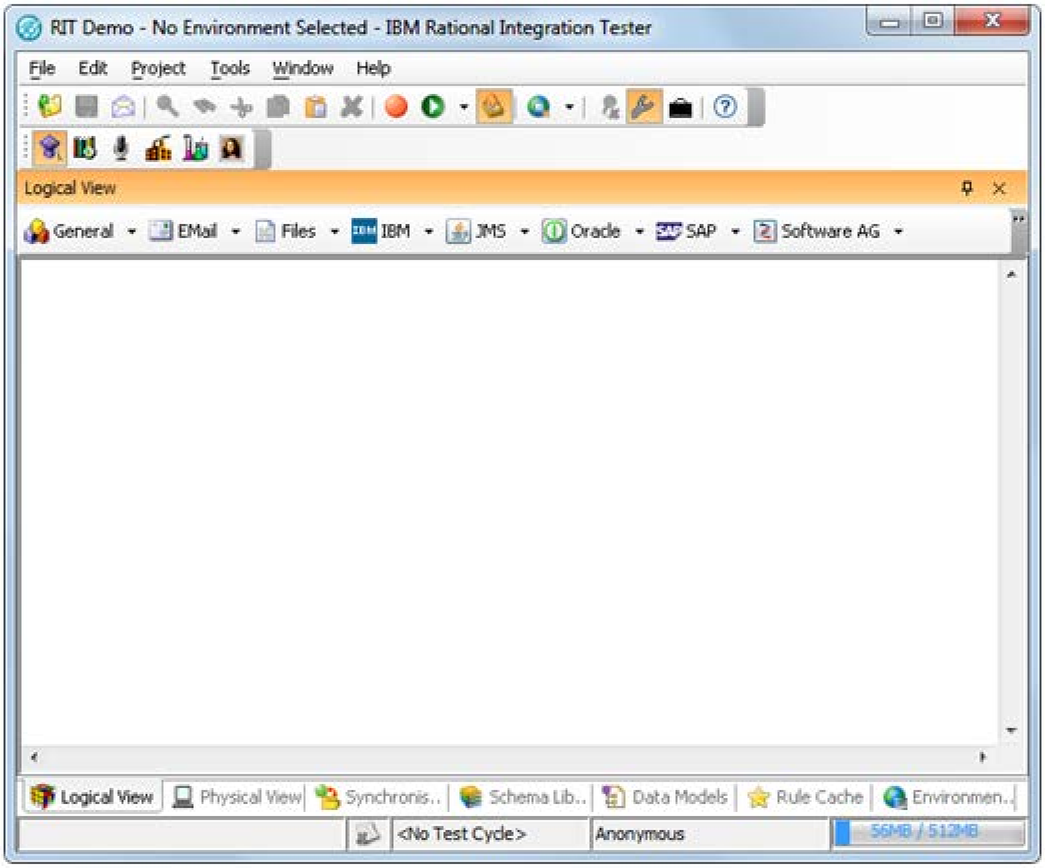
Domain: -- No domain selected --

Results Server (Legacy use only)

A value should only be specified for this field if you have a legacy installation of Results Server. If you want Rational Integration Tester to genrerate URLs to Rational Test Control Panel for browser-based viewing of results then leave this field blank.

12. Click **Finish** to close the Create New Project wizard and to complete creating your new project.

    The Rational Integration Tester application window is displayed.



NOTE: For information about the Rational Integration Tester, refer to Working with Rational Integration Tester.

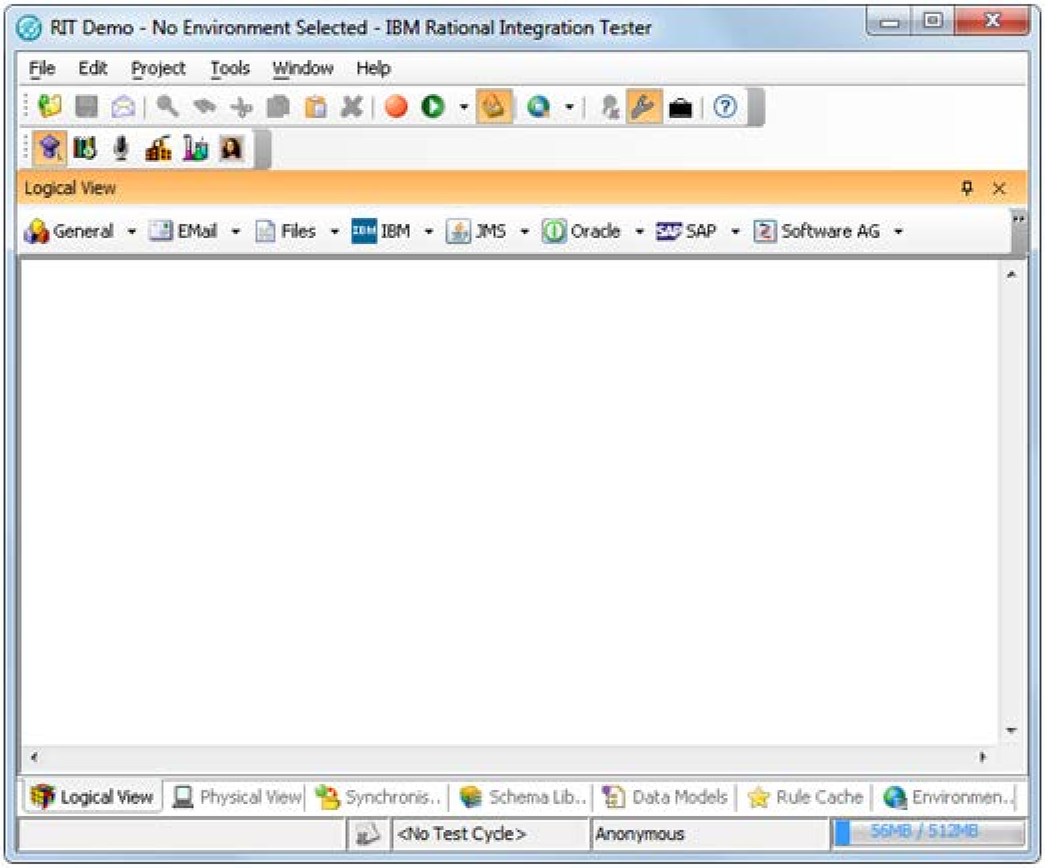
Working with Rational Integration Tester

Contents

This chapter describes the Rational Integration Tester interface (the Workbench) and the perspectives and views that it contains.

2.1 Initial View

After launching Rational Integration Tester for the first time and creating a new project, the following screen is displayed.



This is Architecture School, one of Rational Integration Tester's six perspectives (for information about perspectives, refer to Rational Integration Tester Perspectives).

2.2 GUI Overview

The Rational Integration Tester workspace is a “dockable” user interface, so sub-windows are attached or “docked” to one side of the main application window. These windows can be minimized automatically when not in use. To keep them fully visible, they may be pinned to the workspace using the 📌 button. After a window is pinned, it can be dragged to a different side of the workspace.

As you drag a window around the workspace, a grey outline of the window is displayed to show you where the window will be placed when you release it.

NOTE: If you want to restore the workspace to its default configuration, click **Window > Reset Current Perspective** on the menu bar.

To allow more viewing space in some of the configuration windows, you have the option to hide sections.

2.3 Rational Integration Tester Perspectives

The Rational Integration Tester workspace comprises six perspectives, which can be selected from the Perspectives toolbar.

The following table outlines each perspective.

Button	Perspective Name	Shortcut Key	Purpose
🎓	Architecture School	F7	Enables you to define the architecture of the system under test, including service components and logical and physical resources.
📚	Requirements Library	F8	Enables you to create requirements, in the form of messages, that will help other users to create tests and test data more quickly and more accurately.
🎙	Recording Studio	F9	Enables you to monitor systems and processes, and to record events that are captured by Rational Integration Tester.
🏭	Test Factory	F10	Enables you to create tests and test resources (tests, test suites, stubs, test data sets).
🧪	Test Lab	F11	Enables you to execute test resources that are created in the Test Factory perspective.
🖼	Results Gallery	F12	Enables you to view historical test data and various reports for any stored test run.

The following sections describe the Architecture School perspective. For detailed information about all of the perspectives, refer to *IBM Rational Integration Tester Reference Guide.*

2.3.1 Architecture School

Architecture School is the default perspective of Rational Integration Tester, and it is the first one displayed when the application is started. In this perspective, you build the system to be tested in a simple, graphical fashion. You define the logical and physical system components and their relationships to one another.

You can do the following in Architecture School:

- Create and edit service and infrastructure components.
- Define a service component's operations and its dependencies to other operations and infrastructure components.
- Create and edit physical resources.
- Define the bindings of infrastructure components to their physical resources for different testing environments.

Architecture School contains seven views, which are accessed by named tabs at the bottom of the perspective:

- Logical View
- Physical View
- Synchronization
- Schema Library
- Data Models
- Rule Cache
- Environment Tasks

The following sections outline these views.

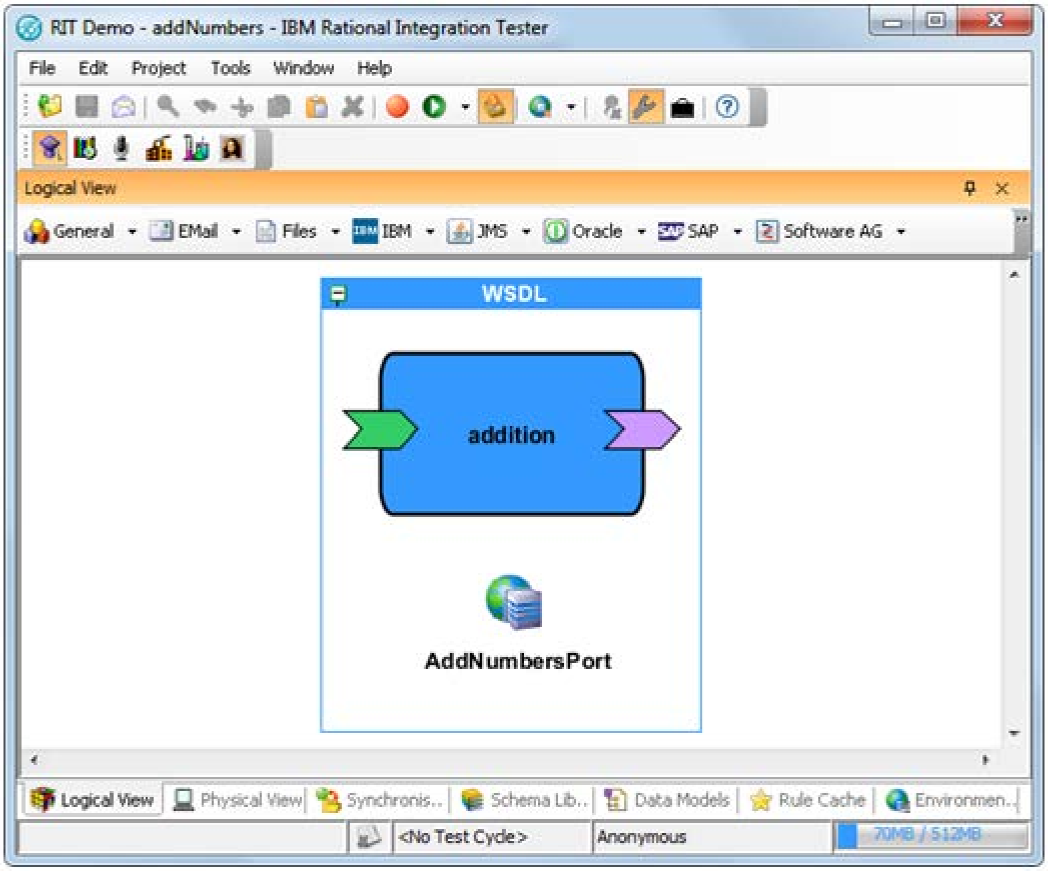
NOTE: The graphics in the following sections are only examples – no items or data are pre-configured in Rational Integration Tester.

selected action includes a rule, it is displayed below. The rule can be edited, and a description can be added if desired.

Click **Delete All** to remove all rules from the selected schema. Click **Delete Orphans** to remove all rules from the selected schema that are associated with fields that no longer exist (due to changes to the schema). This button is only available when "orphaned" rules actually exist for the schema.

Logical View

The Logical View displays the services and components of the system under test, and the dependencies between them. It enables you to look at the system in an abstract manner, without having to worry about the physical locations of the different parts of the system.

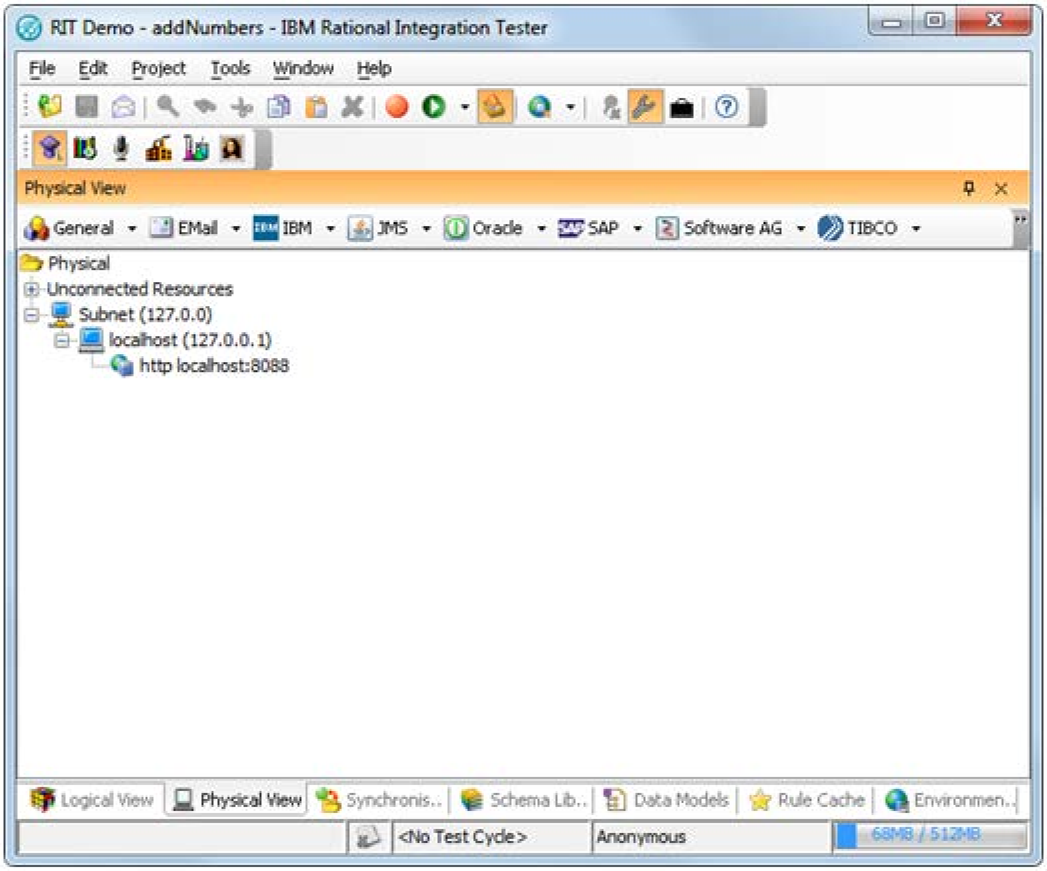


The following table summarizes how to use the Logical View.

To...	Do this...
Navigate the Logical View	Use the horizontal and vertical scroll bars, or click the **Pan** cursor and click the contents of the screen and move them around.
Adjust the zoom level	Use the zoom buttons (and), or hold down CTRL and use your mouse's wheel button to vary the zoom level.
Move any of the components or operations around	Ensure that the **Select** cursor is active, click the item to move and drag it to the desired location.
Force the application organize the layout of all components and operations	Click **Layout All Nodes** .
Set the zoom level so that the entire diagram fits inside the screen	Click **Fit to Contents** .

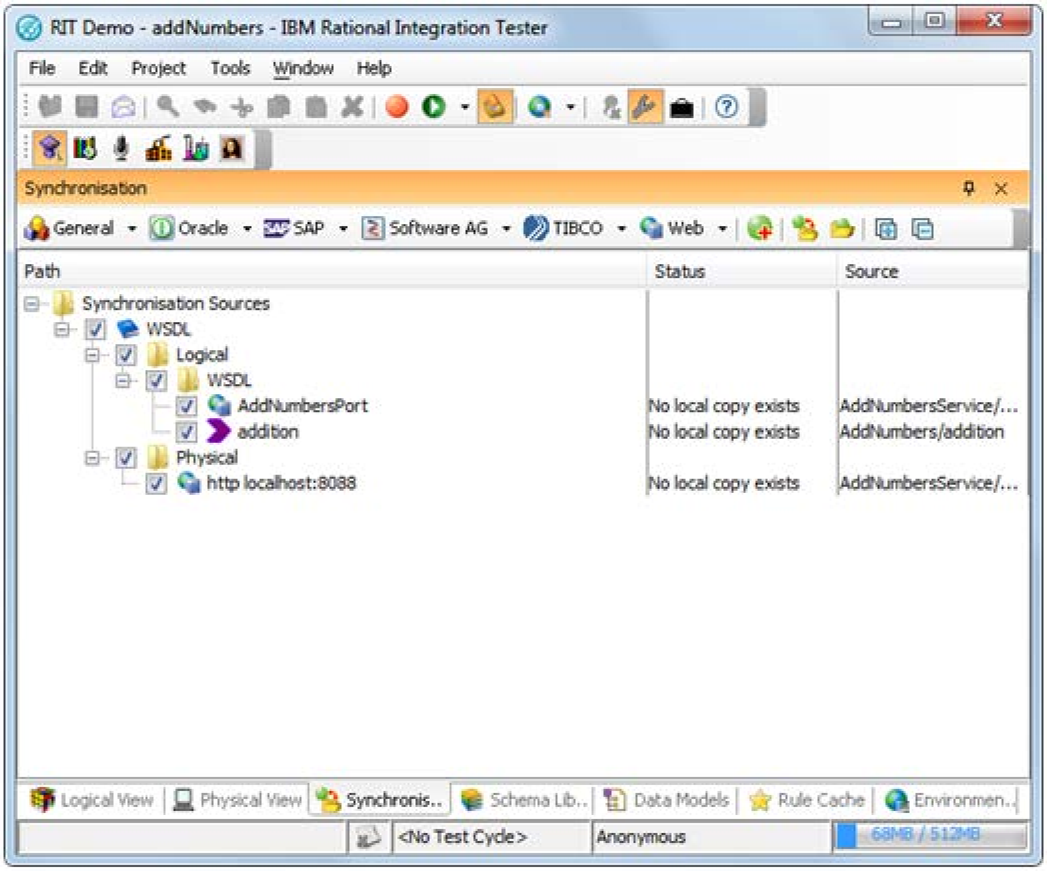
Physical View

The Physical View displays available physical resources and their location within the enterprise. Resources in this view are organized by subnet and host. If a resource is not associated with a subnet or host, it will be displayed under **Unconnected Resources**.



Synchronisation View

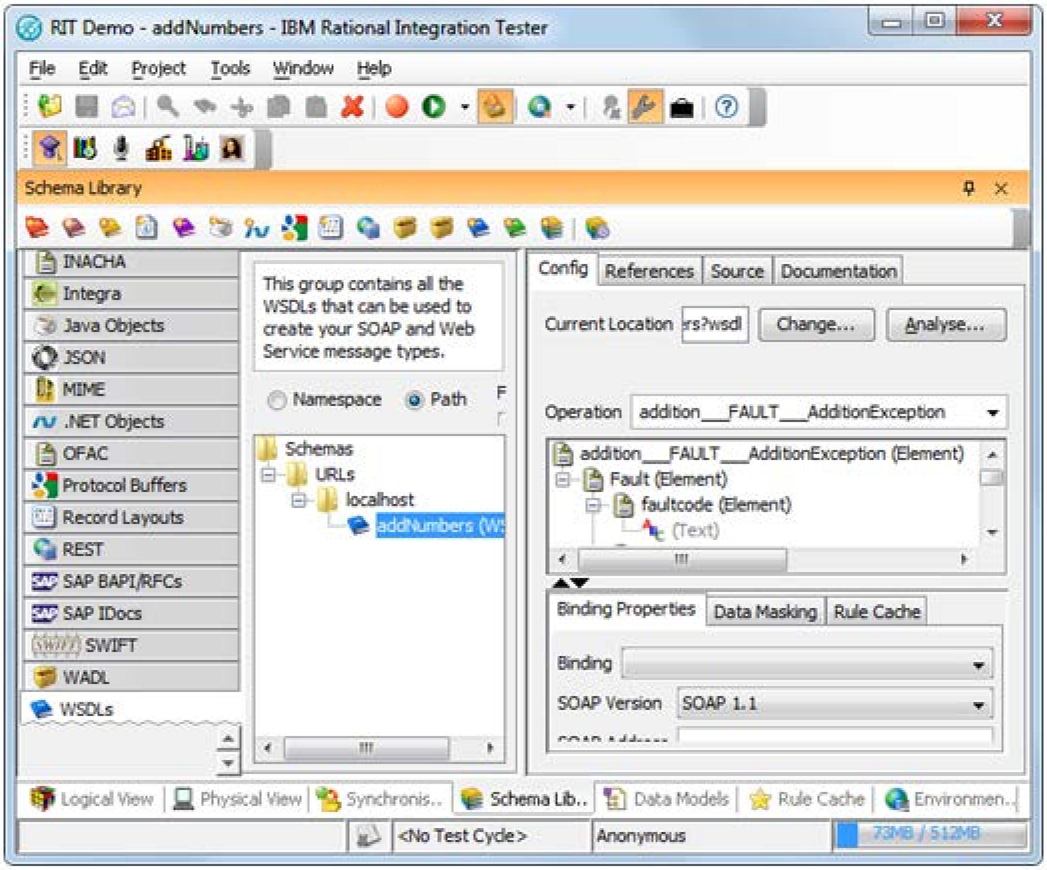
The Synchronisation view enables you to add external resources to your project, view resources that have been added, and synchronize those resources with your local system.



Path	Status	Source
Synchronisation Sources		
☑ WSDL		
☑ Logical		
☑ WSDL		
☑ AddNumbersPort	No local copy exists	AddNumbersService/...
☑ addition	No local copy exists	AddNumbers/addition
☑ Physical		
☑ http localhost:8088	No local copy exists	AddNumbersService/...

Schema Library View

The Schema Library view enables you to manage and view the schemas and message formats that are available in your project.



RIT Demo - addNumbers - IBM Rational Integration Tester

This group contains all the WSDLs that can be used to create your SOAP and Web Service message types.

○ Namespace ● Path

Current Location rs?wsdl

Operation addition__FAULT__AdditionException

Binding

SOAP Version SOAP 1.1

<No Test Cycle>

Anonymous

73MB / 512MB

Each schema category and format is listed under its own tab on the left side of the view. When a category is selected, all of the schemas available in the project are displayed in a tree format in the view’s center frame.

The schemas can be sorted by clicking the **Namespace** and **Path** option buttons above the tree view, and the types of schemas displayed (files or URLs) can be limited by selecting filtering options under **Filters**.

For information about adding schemas, and viewing and managing schema details, refer to *IBM Rational Integration Tester Reference Guide* or *IBM Rational Integration Tester Getting Started Guide*.

Data Models View

A data model enables you to understand the data held by the system that you want to simulate, and it can provide persistent storage of that data.

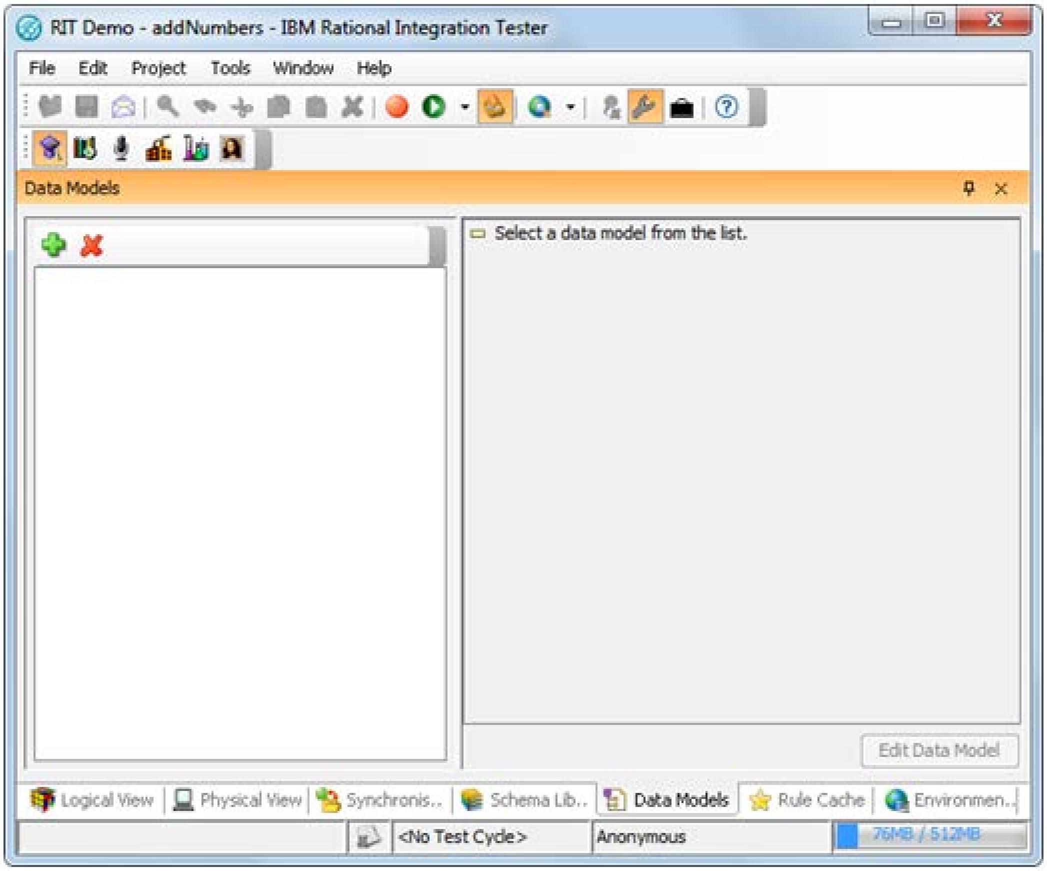
Rational Integration Tester provides two methods of creating a data model:

- You can create a data model while creating a data model-driven stub from recorded events.
- Alternatively, you can use the Data Model Editor to create a data model.

For more information about both of these methods, refer to *IBM Rational Test Virtualization Server Reference Guide*.

The Data Models view enables you to:

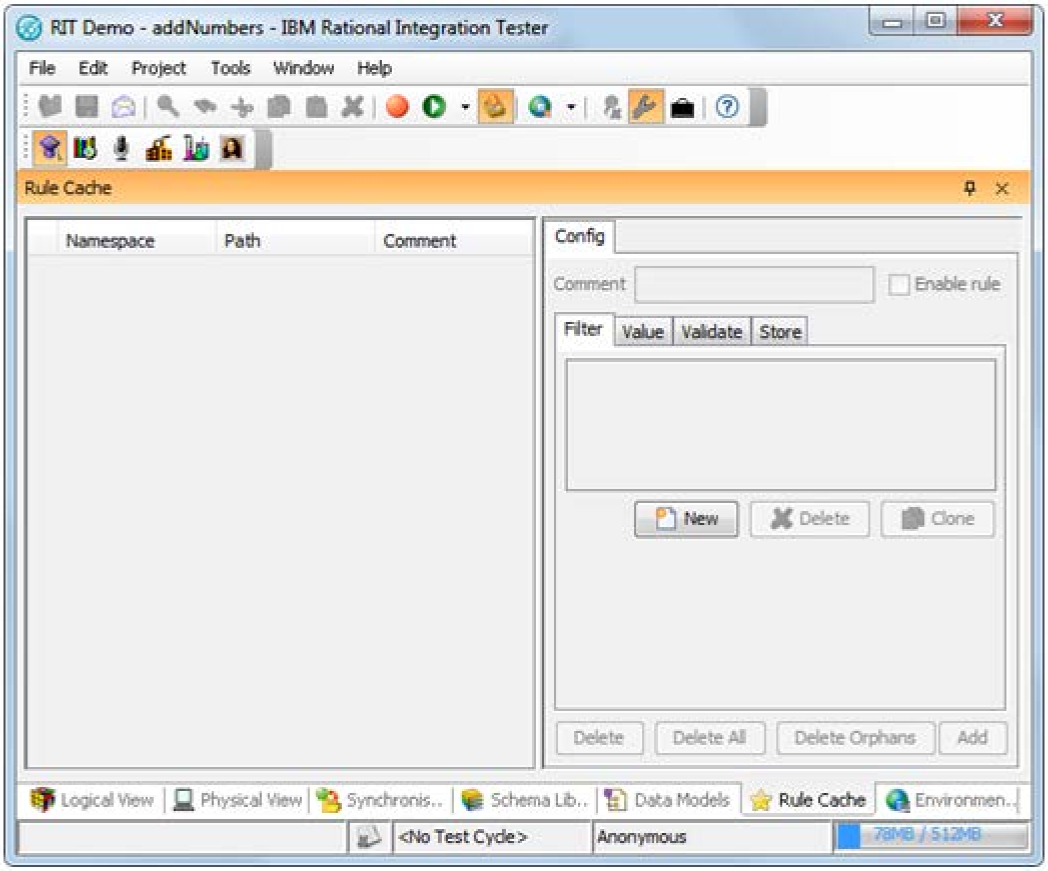
- Access the Data Model Editor to create or modify data models.
- View or delete existing data models.



Select a data model from the list.

Rule Cache View

The Rule Cache view enables you to manage validation rules.



When validation errors are repaired and cached (for information about this, refer to *IBM Rational Integration Tester Reference Guide*), the repair method is saved as a validation rule. These rules will be applied to the same field from which they are created when the same schema is in use.

Existing rules are listed on the left side of the view, and they can be sorted by namespace or path by clicking on the appropriate column heading.

NOTE: The same icons are used on individual fields to indicate where a rule has been applied and whether or not the rule is currently enabled.

Under the **Config** tab on the right side of the view, the selected rule can be enabled or disabled by selecting or clearing the **Enable rule** check box. Validation actions are listed under the **Validate** tab. Actions can be created, modified, cloned, and deleted, the same way as in a Message Editor (for information about this, refer to *IBM Rational*

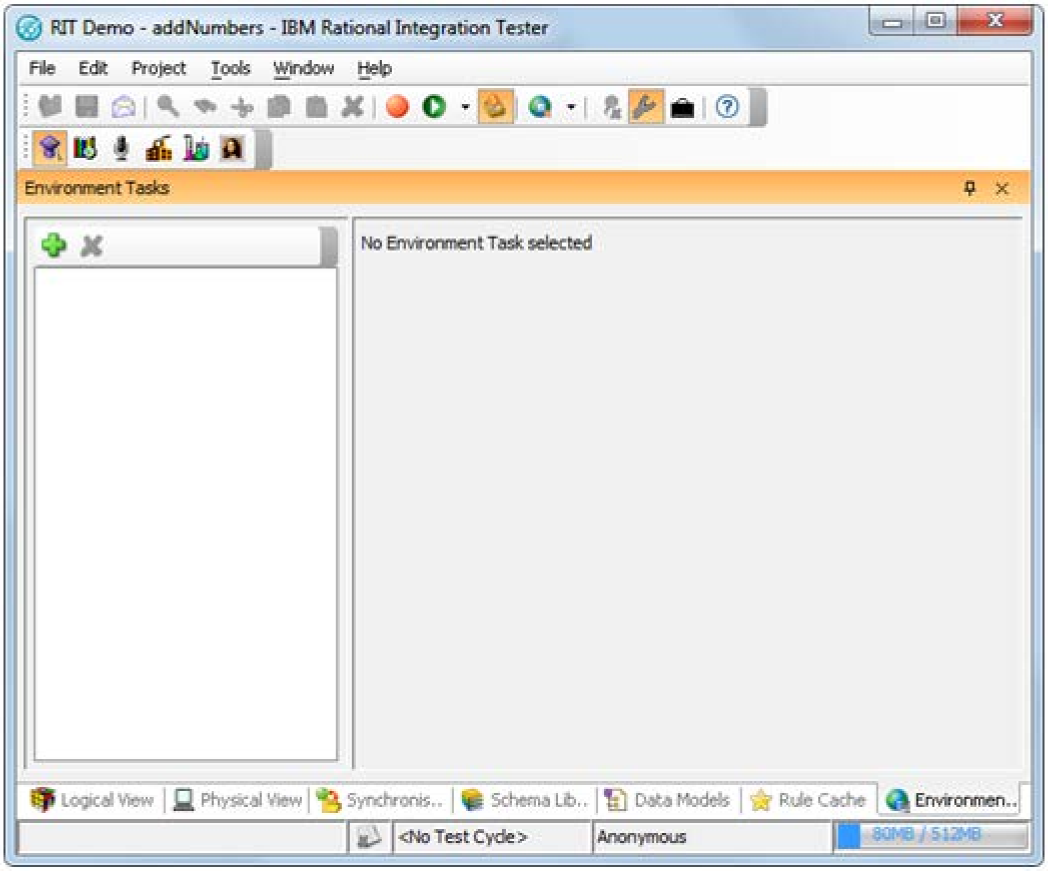
Integration Tester Reference Guide). When the selected action includes a rule, it is displayed below. The rule can be edited, and a description can be added if desired.

Click **Delete All** to remove all rules from the selected schema. Click **Delete Orphans** to remove all rules from the selected schema that are associated with fields that no longer exist (due to changes to the schema). (The **Delete Orphans** button is available only when "orphaned" rules actually exist for the schema.)

For more information about rules, refer to *IBM Rational Integration Tester Reference Guide* or *IBM Rational Integration Tester Getting Started Guide*.

Environment Tasks View

The Environment Tasks view enables you to create, modify, delete, and run any environment tasks that must be run before a stub starts.

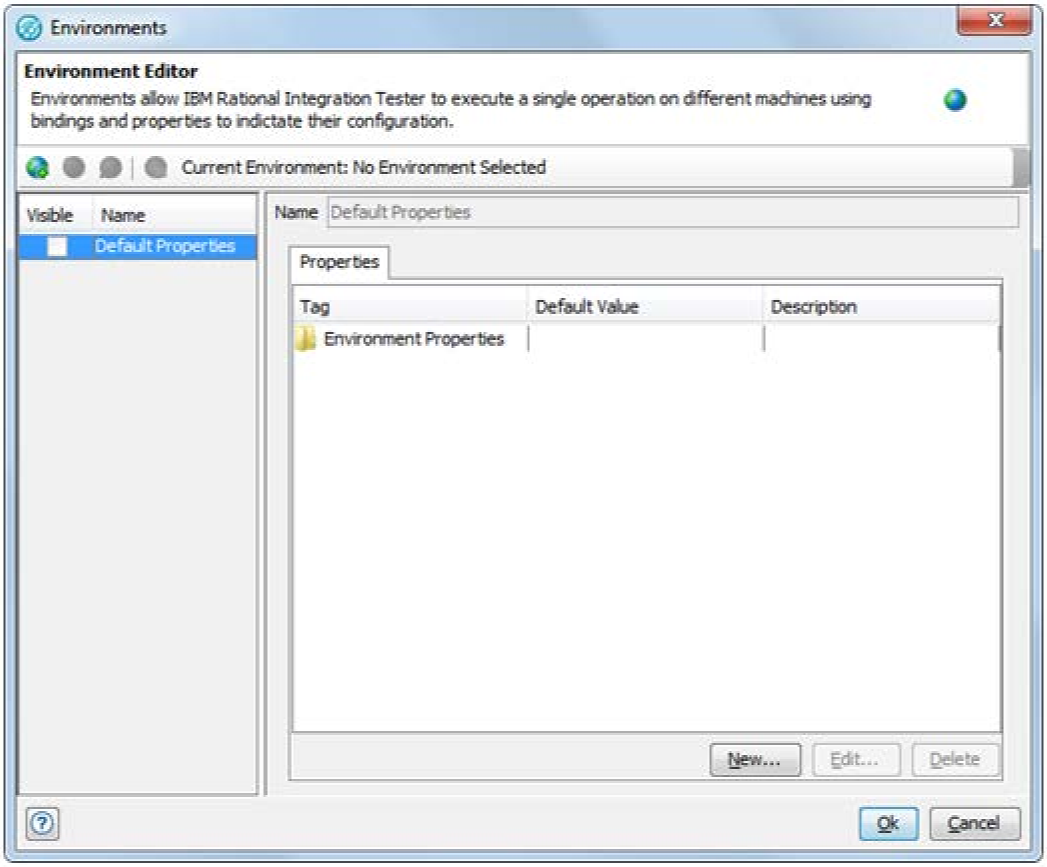


For more information about environments, refer to Environments and to *IBM Rational Test Virtualization Server Reference Guide* and to *IBM Rational Test Control Panel System Administration Guide.*

2.4 Environments

An environment is a set of variables (tags) and bindings that can be used to provide runtime configuration settings to test resources. By creating multiple environments (for example, “System Test”, “UAT”, and so on), you can quickly run the same set of test resources against different phases of your product lifecycle. For example, environment tags can be used to modify messaging settings, such as port numbers.

By default, a new Rational Integration Tester project is not an associated with an environment. Clicking **Project > Edit Environments** on the menu bar displays the Environments dialog box, which enables you to associate your project with an existing environment.



Environments

Environment Editor

Environments allow IBM Rational Integration Tester to execute a single operation on different machines using bindings and properties to indictate their configuration.

Current Environment: No Environment Selected

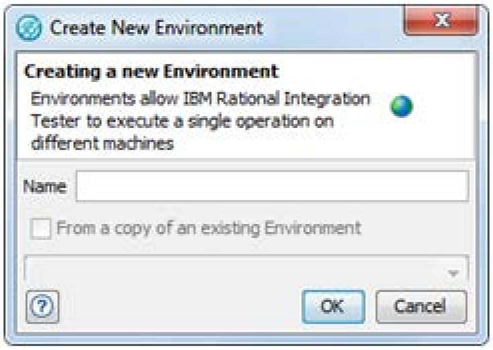
Visible	Name
□	Default Properties

Name: Default Properties

Properties

Tag	Default Value	Description
Environment Properties		

Alternatively, if you want to create a new environment for your project, click **Project > Create New Environment**.



Create New Environment

Creating a new Environment

Environments allow IBM Rational Integration Tester to execute a single operation on different machines

Name

□ From a copy of an existing Environment

Environment variables are referred to as tags, and are accessed by surrounding the name with double percent signs, for example, `%%name%%`. For example, if you were to create an HTTP transport using a port number, you might configure the connection setting to use `%%HTTP_Port%%`. If you refer to a non-existent tag, it is shown underlined: `%UnknownTag%%`.

The configuration of bindings in an environment, which define the relationship between logical and physical components in the project, is described in *IBM Rational Integration Tester Getting Started Guide* and *IBM Rational Integration Tester Reference Guide*.

Testing a Web Service

Contents

This chapter describes how to import a WSDL into Rational Integration Tester, and how to create and run tests based on the operations and input parameters defined in the WSDL.

3.1 Synchronizing With WSDLs

In Rational Integration Tester, synchronization enables you to add external resources to your project and create automatically all of the artefacts that are needed to test the resource.

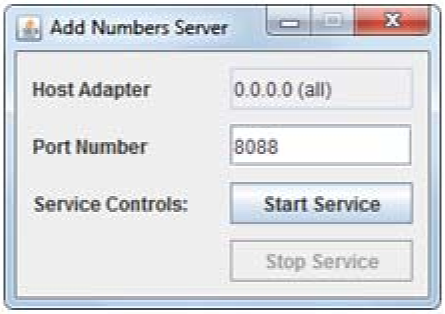
The remainder of this document uses an example web service called “AddNumbers” to show various test creation capabilities of Rational Integration Tester.

To add the AddNumbers web service to your project:

1. In `<`*`Rational Integration Tester Installation Directory`*`>\examples\addnumbers`, run `start.bat`.

   Alternatively, if Rational Integration Tester is running on Linux or Unix, run `start.sh`.

   The Add Numbers Server dialog box is displayed.



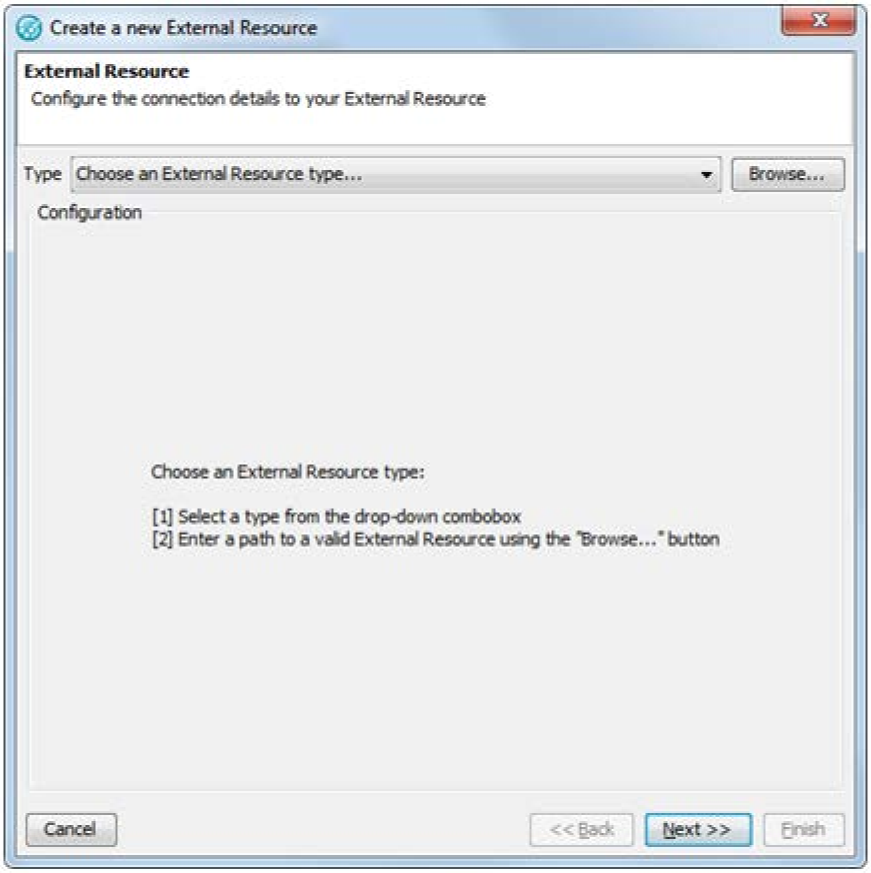
Host Adapter: 0.0.0.0 (all)

Port Number: 8088

2. Click **Start Service**.
3. In Rational Integration Tester, open the Architecture School perspective and click the **Synchronisation** tab.
4. Click ➕ on the toolbar of the Synchronisation view to add a new external resource.

   Alternatively, click **Web > WSDL** on the toolbar of the Synchronisation view.

   The Create a new External Resource wizard is displayed.



Create a new External Resource

External Resource

Configure the connection details to your External Resource

Type
Choose an External Resource type...
Browse...
Configuration

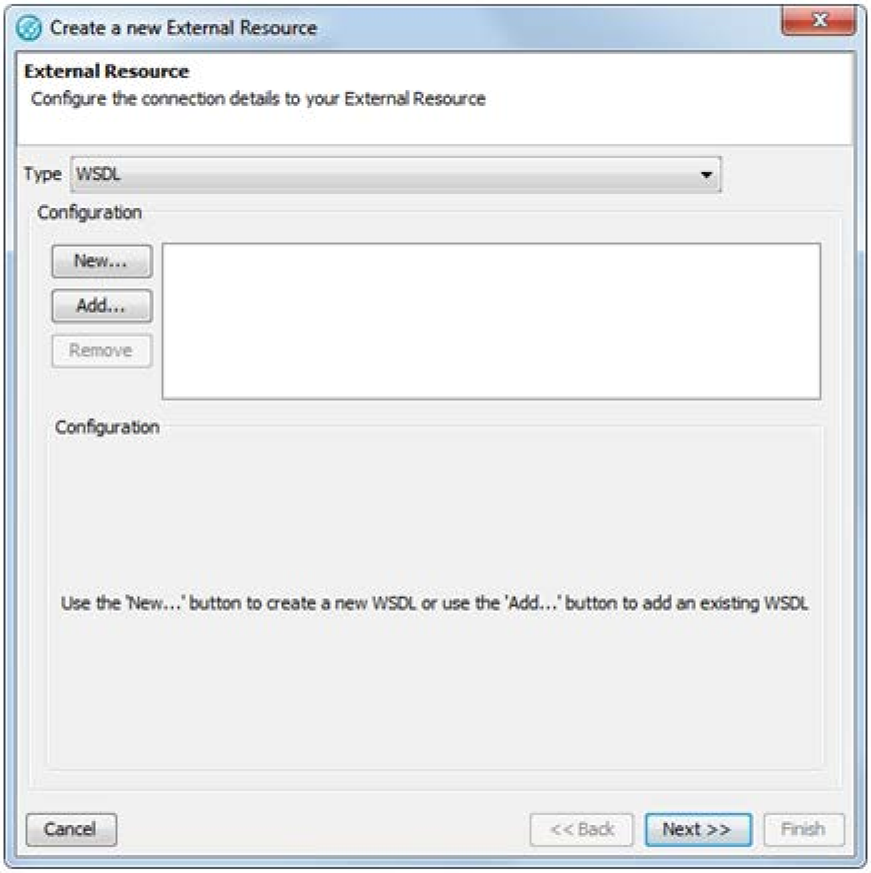
Choose an External Resource type:

[1] Select a type from the drop-down combobox
[2] Enter a path to a valid External Resource using the "Browse..." button

Cancel
<< Back
Next >>
Finish

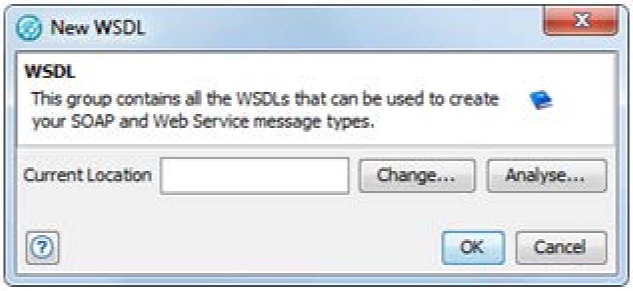
5. In the **Type** list, click **WSDL**.

   The second screen of the wizard is displayed.


Create a new External Resource
External Resource
Configure the connection details to your External Resource
Type WSDL
Configuration
New...
Add...
Remove
Configuration
Use the 'New...' button to create a new WSDL or use the 'Add...' button to add an existing WSDL
Cancel
<< Back
Next >>
Finish

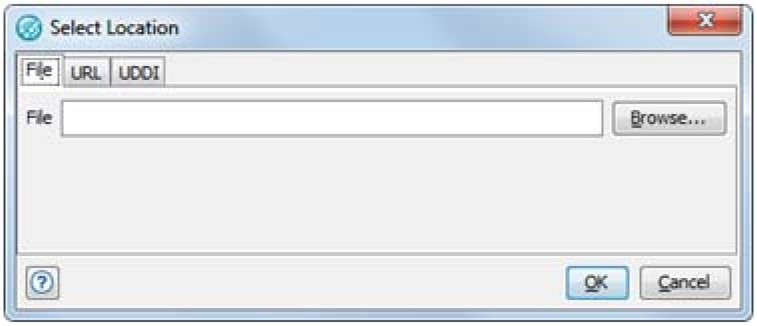
6. Under **Configuration**, click **New**.

   The New WSDL dialog box is displayed.


New WSDL
WSDL
This group contains all the WSDLs that can be used to create your SOAP and Web Service message types.
Current Location
Change...
Analyse...
OK
Cancel

7. Click **Change** to find the WSDL.

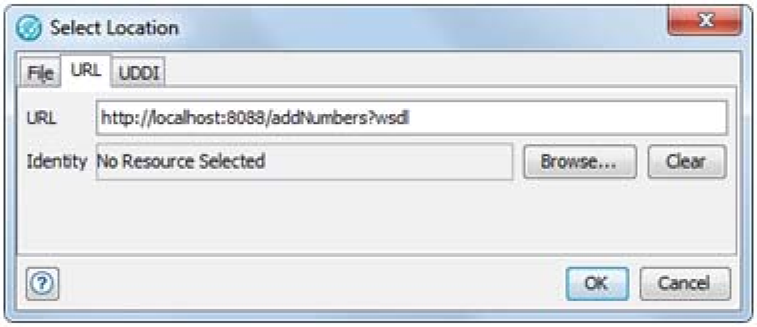
   The Select Location dialog box is displayed.



8. Click the **URL** tab.

9. In the **URL** field, enter the following URL:

```
http://localhost:8088/addNumbers?wsdl
```



URL: http://localhost:8088/addNumbers?wsdl

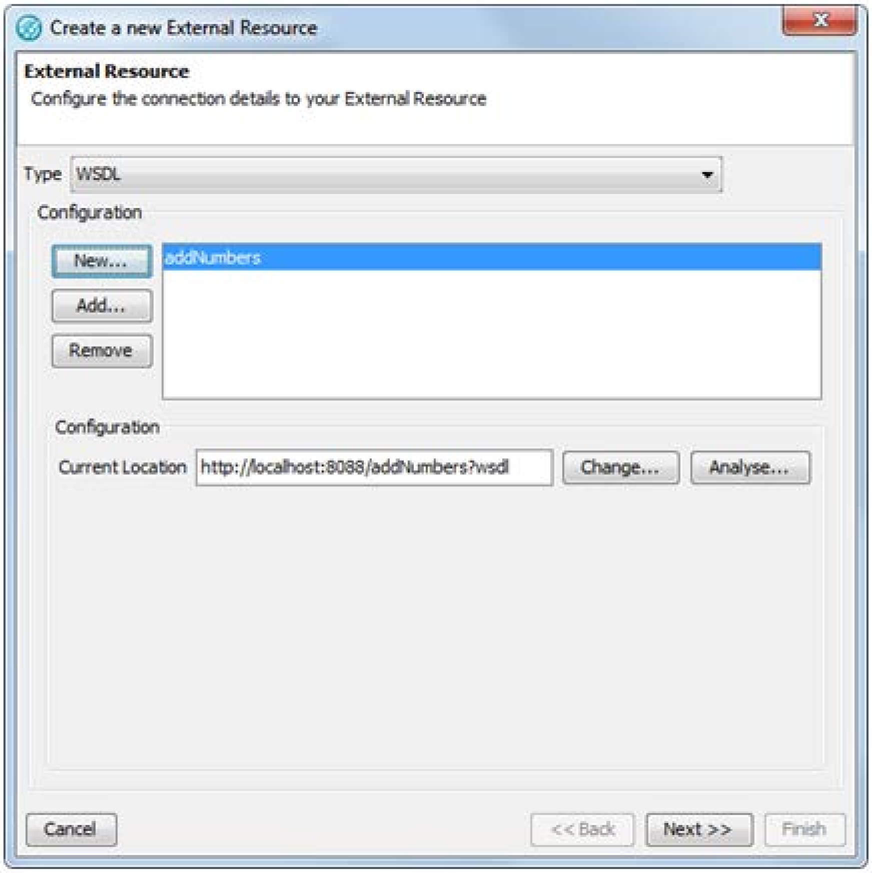
Identity: No Resource Selected

NOTE: If you want to use an alternative WSDL and it is hosted on a password-protected server, you can use an existing Identity resource to specify the login credentials by clicking **Browse** (on the **URL** tab) to open the Identity Selection dialog box (for more information about this, refer to *IBM Rational Integration Tester Reference Guide*).

10. Click **OK** to close the Select Location dialog box.

11. Click **OK** to close the New WSDL dialog box.

12. The information displayed on the second screen of the Create a new External Resource wizard is updated.


Create a new External Resource

External Resource

Configure the connection details to your External Resource

Type: WSDL

Configuration

addNumbers

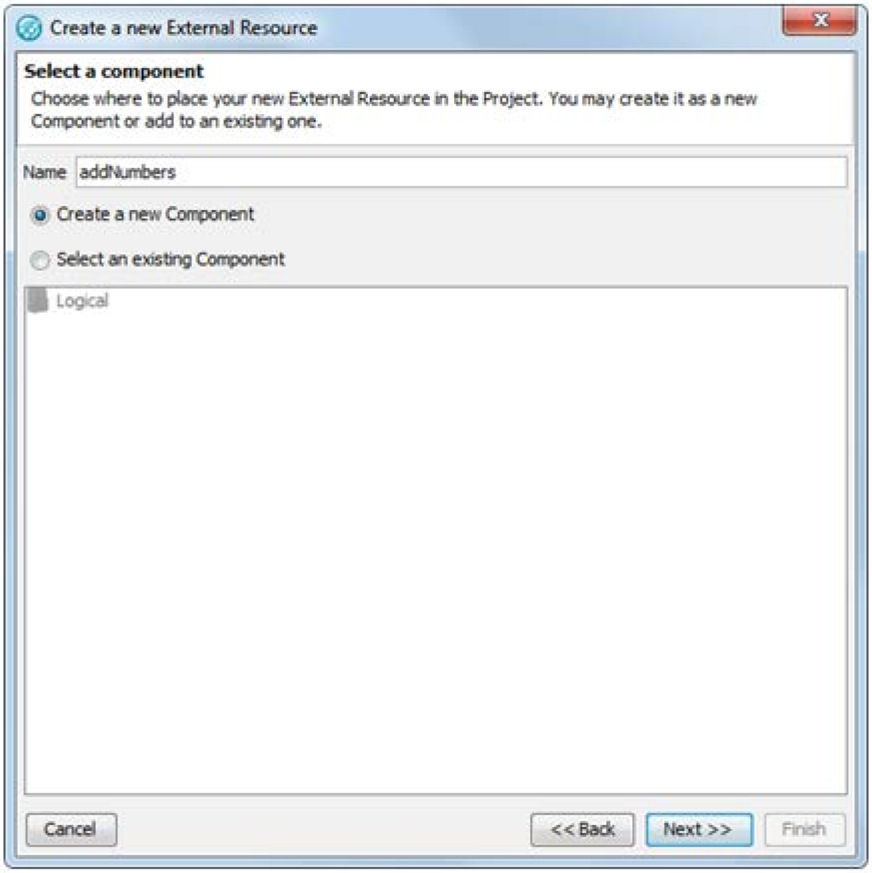
Configuration

Current Location: http://localhost:8088/addNumbers?wsdl

13. Click **Next**.

    The third screen of the Create a new External Resource wizard is displayed.

    `addNumbers` should be displayed in the **Name** field.



Create a new External Resource

Select a component

Choose where to place your new External Resource in the Project. You may create it as a new Component or add to an existing one.

Name addNumbers

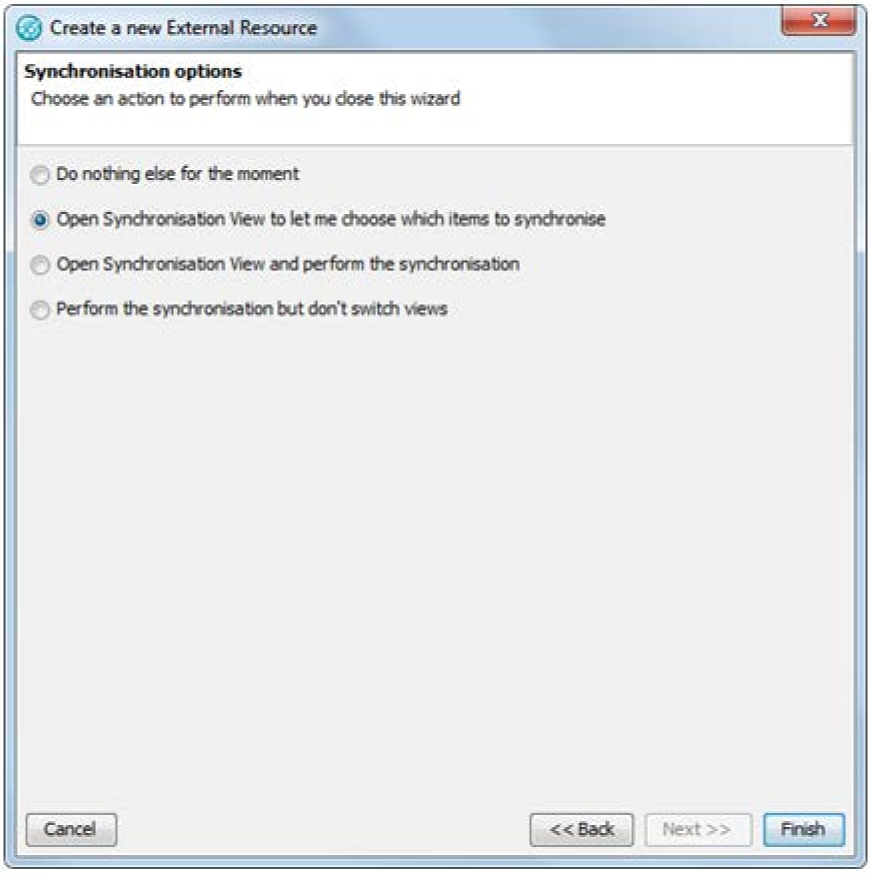
● Create a new Component

○ Select an existing Component

Logical

14. Click the **Create a new Component** option button (if necessary).
15. Click **Next**.

The fourth screen of the Create a new External Resource wizard is displayed.



Create a new External Resource

Synchronisation options

Choose an action to perform when you close this wizard

○ Do nothing else for the moment

● Open Synchronisation View to let me choose which items to synchronise

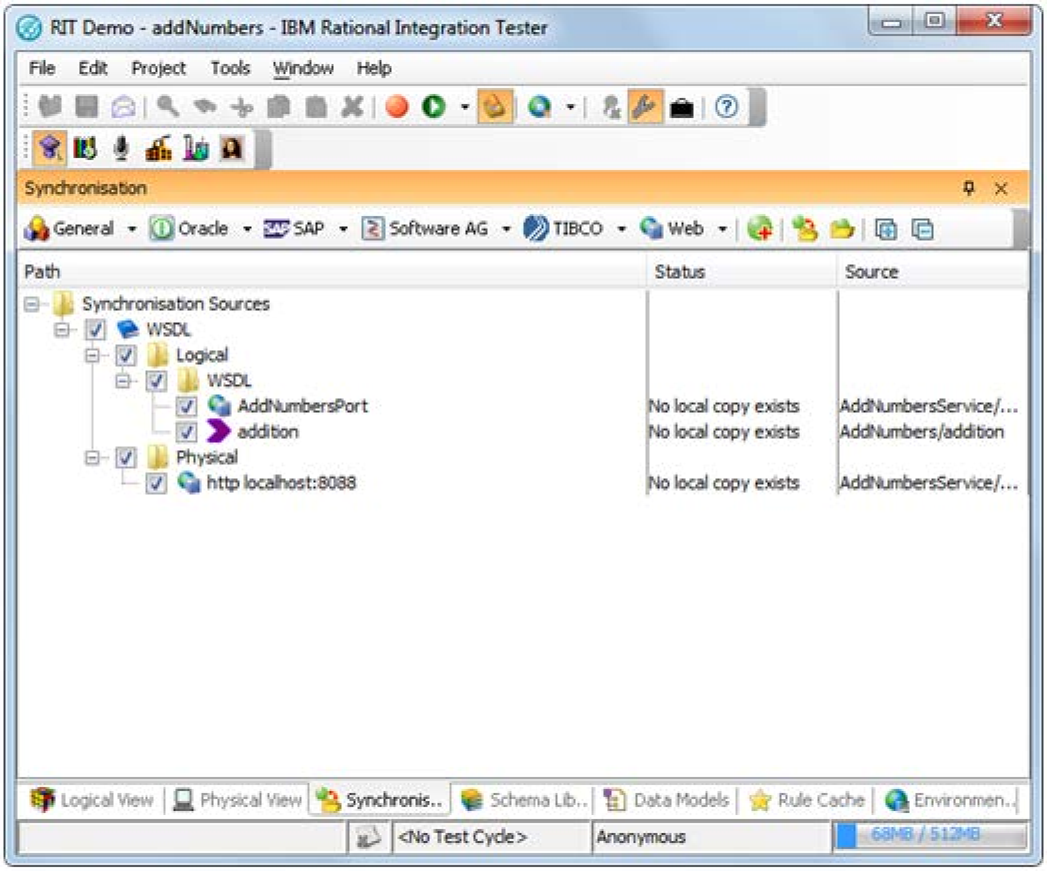
○ Open Synchronisation View and perform the synchronisation

○ Perform the synchronisation but don't switch views

16. Click the **Open Synchronisation View to let me choose which items to synchronise** option button (if necessary).

17. Click **Finish**.

    The Synchronisation view is refreshed and AddNumbers artefacts are displayed.

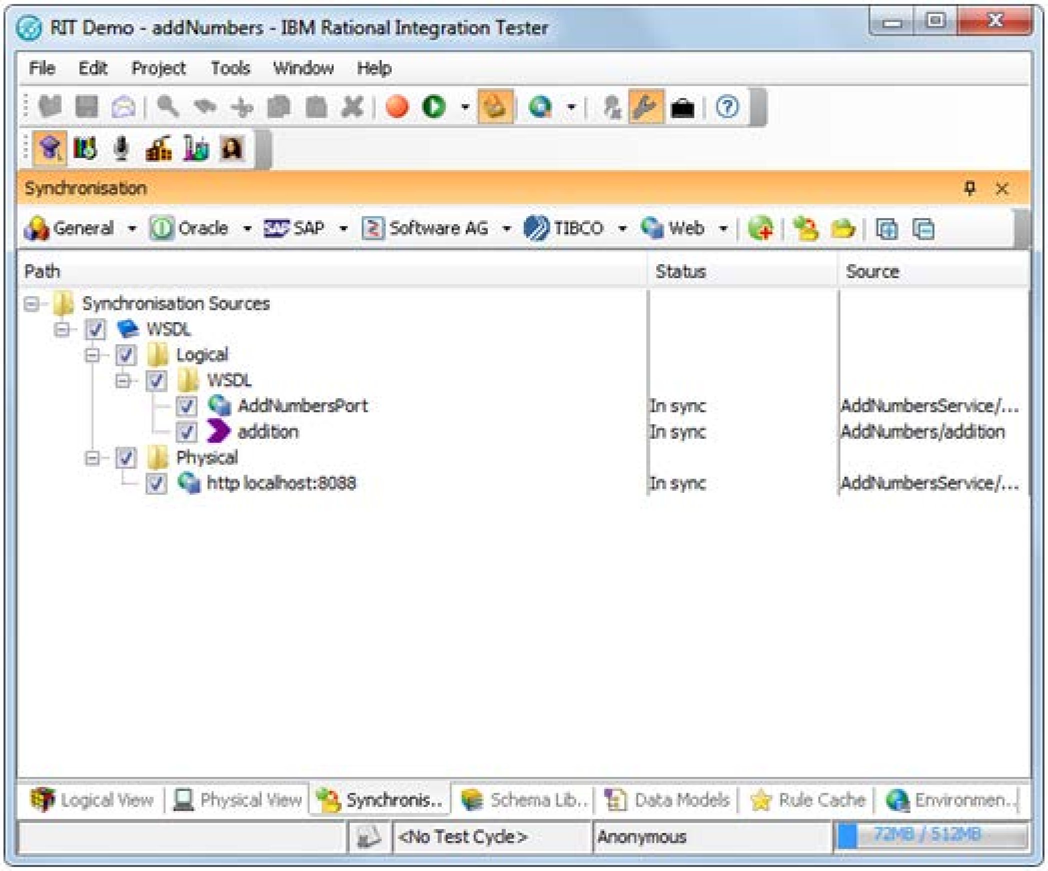


Path	Status	Source
Synchronisation Sources		
☑ WSDL		
☑ Logical		
☑ WSDL		
☑ AddNumbersPort	No local copy exists	AddNumbersService/...
☑ addition	No local copy exists	AddNumbers/addition
☑ Physical		
☑ http localhost:8088	No local copy exists	AddNumbersService/...

18. Ensure that the **addNumbers** check box is selected.

19. Click the 📂 button to create a local copy in Rational Integration Tester of the `AddNumbers` artefacts.

After synchronization is complete, the status of all AddNumbers artefacts should change from `No local copy exists` to `In sync`.



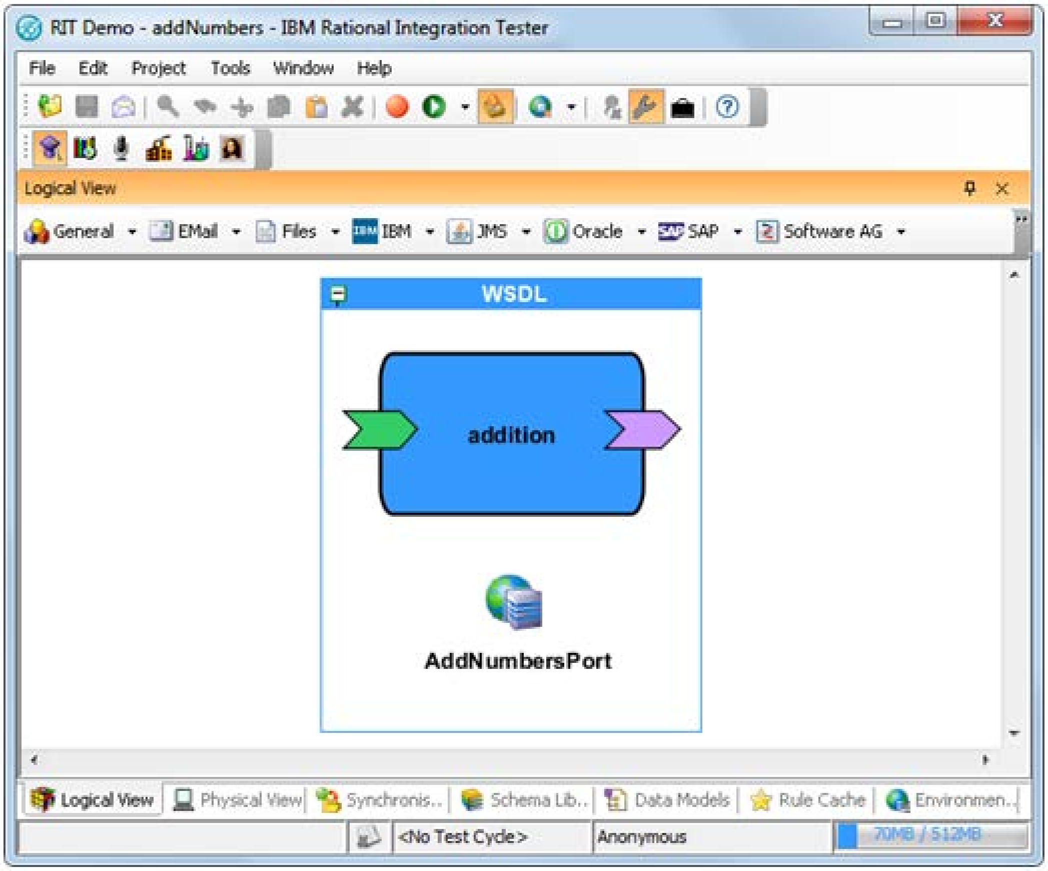
Path	Status	Source
⊟ Synchronisation Sources		
⊟ ☑ WSDL		
⊟ ☑ Logical		
⊟ ☑ WSDL		
☑ AddNumbersPort	In sync	AddNumbersService/...
☑ addition	In sync	AddNumbers/addition
⊟ ☑ Physical		
☑ http localhost:8088	In sync	AddNumbersService/...

3.2 Verifying WSDL Resources

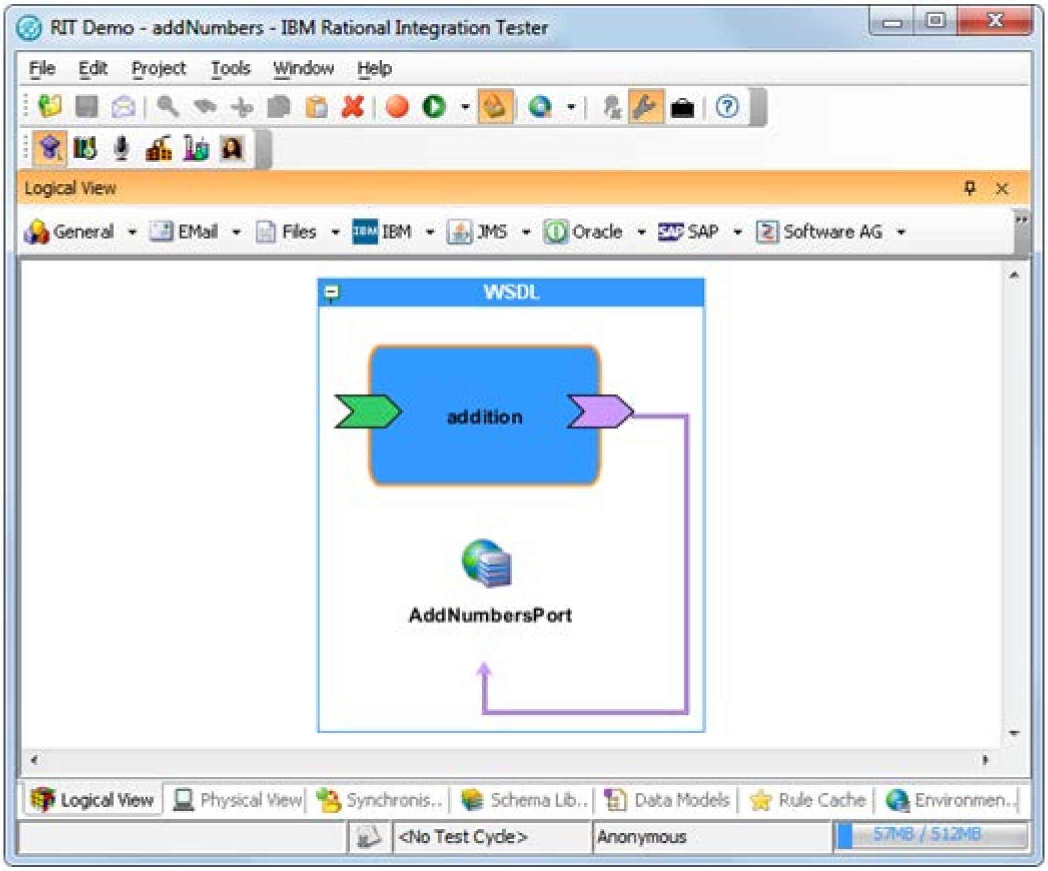
To verify the AddNumbers web service that you have added to your project:

1. Click the **Logical View** tab.

   Rational Integration Tester has created a service called **addNumbers**. Within the service, an AddNumbers service component has been created based on the port type in the WSDL. Within this service component, an operation called **addition** has been created based on the name of the operation in the WSDL.

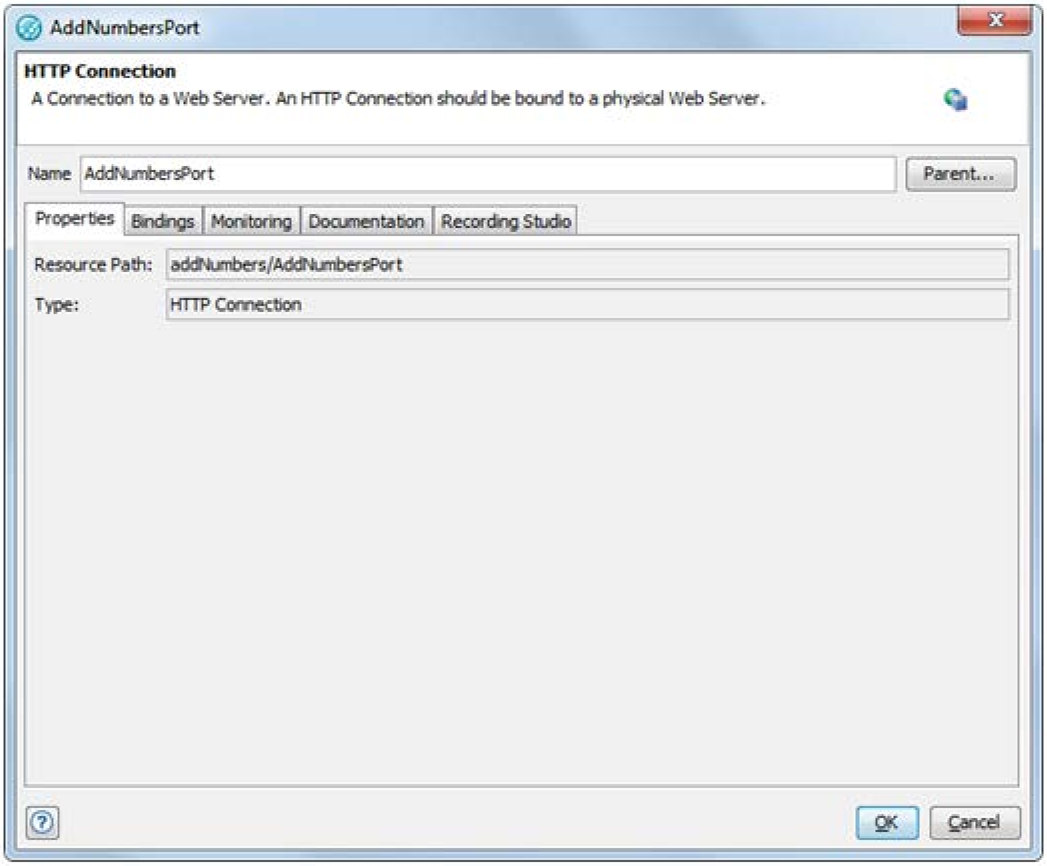


2. Clicking **addition** displays a dependency on the **AddNumbersPort** HTTP connection, which is highlighted by a purple arrow.


RIT Demo - addNumbers - IBM Rational Integration Tester
File Edit Project Tools Window Help
Logical View
General EMail Files IBM JMS Oracle SAP Software AG
WSDL
addition
AddNumbersPort
Logical View Physical View Synchronis.. Schema Lib.. Data Models Rule Cache Environmen..
<No Test Cycle>
Anonymous
57MB / 512MB

3. Double-click **AddNumbersPort**.

   The ports's HTTP Connection dialog box is displayed.



AddNumbersPort

HTTP Connection

A Connection to a Web Server. An HTTP Connection should be bound to a physical Web Server.

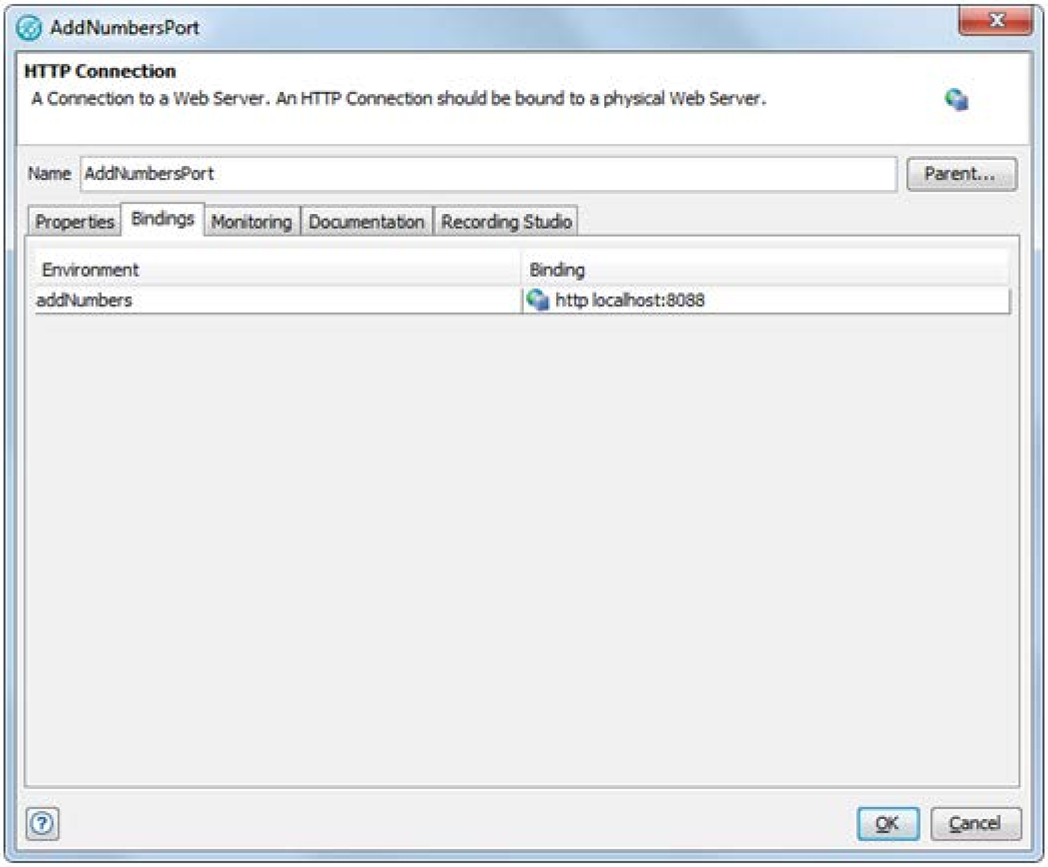
Name: AddNumbersPort

Resource Path: addNumbers/AddNumbersPort

Type: HTTP Connection

4. Click the **Bindings** tab.

   The port should be associated with the environment created by the AddNumbers web service.



HTTP Connection

A Connection to a Web Server. An HTTP Connection should be bound to a physical Web Server.

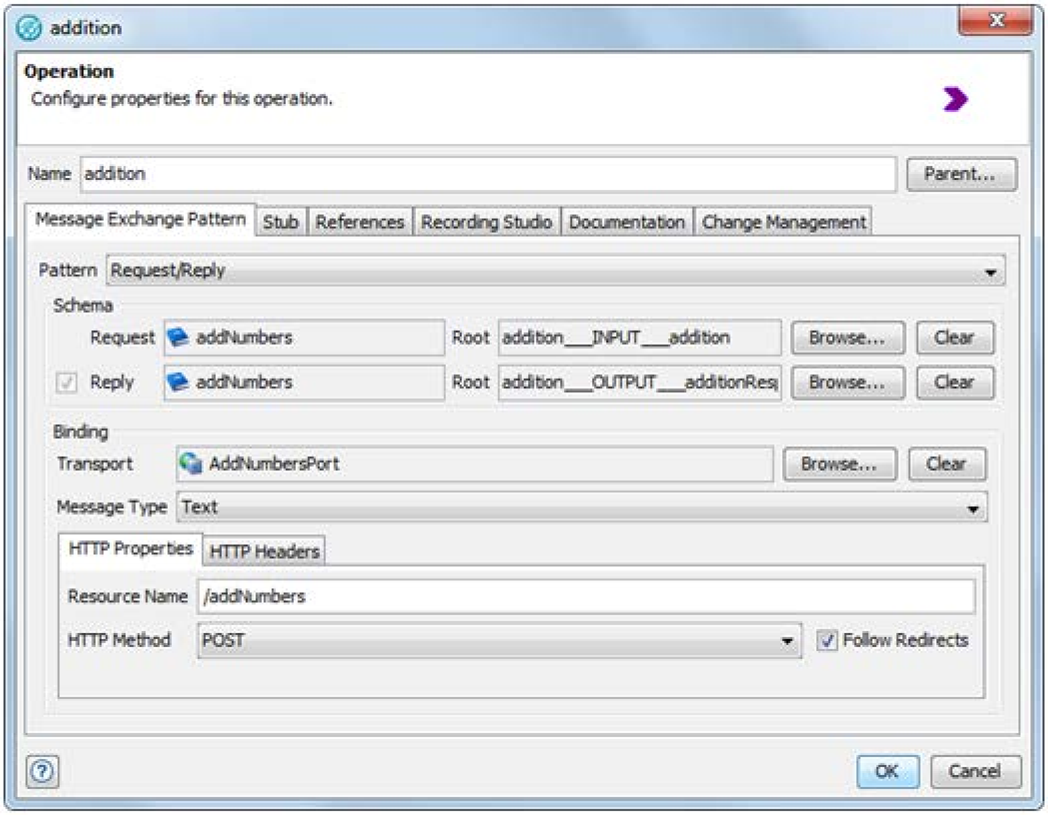
Name AddNumbersPort

Environment	Binding
addNumbers	http localhost:8088

NOTE: If you are using a different web service or if you want to use a different environment, you will have to associate the port to the desired environment manually. For information about this, refer to *IBM Rational Integration Tester Reference Guide*.

5. Click **OK**.
6. Double-click **addition**.

The **addition** operation's Operation dialog box is displayed.



Name: addition

Pattern: Request/Reply

Request: addNumbers — Root: addition__INPUT__addition

☑ Reply: addNumbers — Root: addition__OUTPUT__additionRes

Transport: AddNumbersPort

Message Type: Text

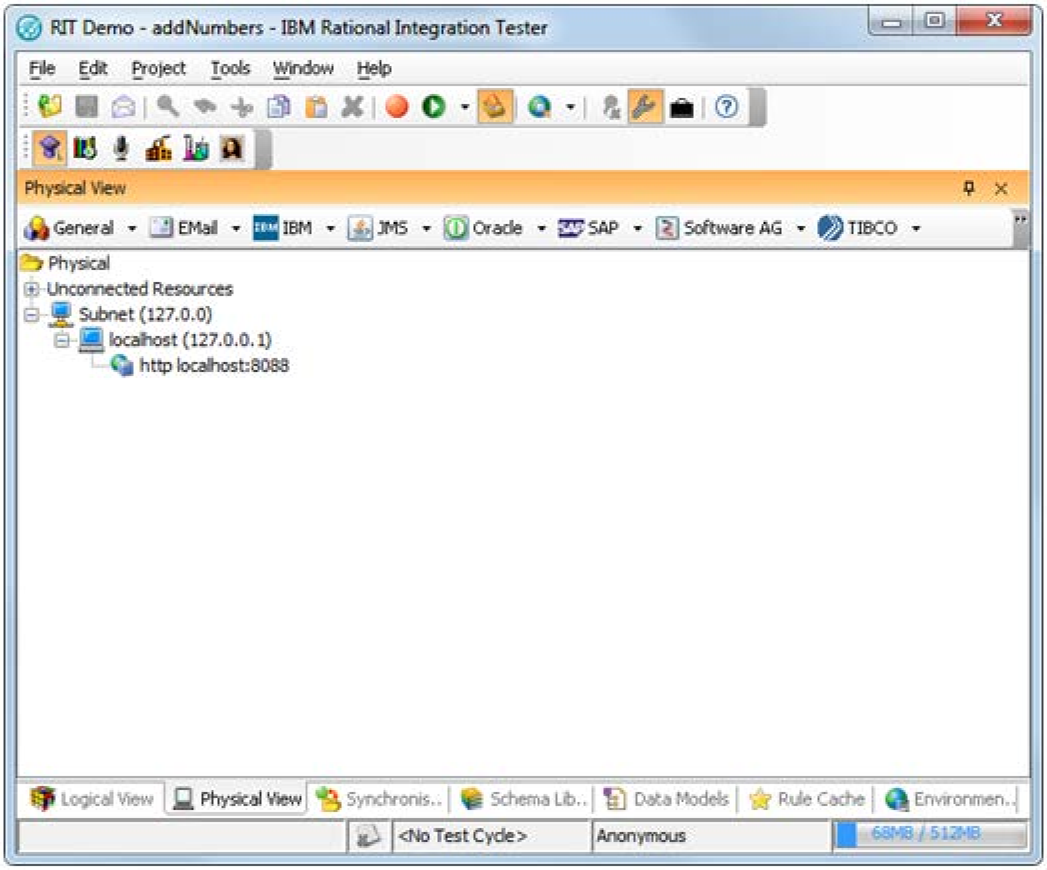
Resource Name: /addNumbers

HTTP Method: POST — ☑ Follow Redirects

The dialog box enables you to view and modify the operation's properties. For example, the **Message Exchange Pattern** tab defines the operation as a request/reply and defines the schemas for the request and reply messages and the transport used to send the data.

7. Click **OK**.
8. Click the **Physical View** tab.

   The Physical View is displayed.



An artefact called **http localhost:8088** should be displayed. This is the HTTP transport that Rational Integration Tester will use to send and receive messages to and from the AddNumbers web service.

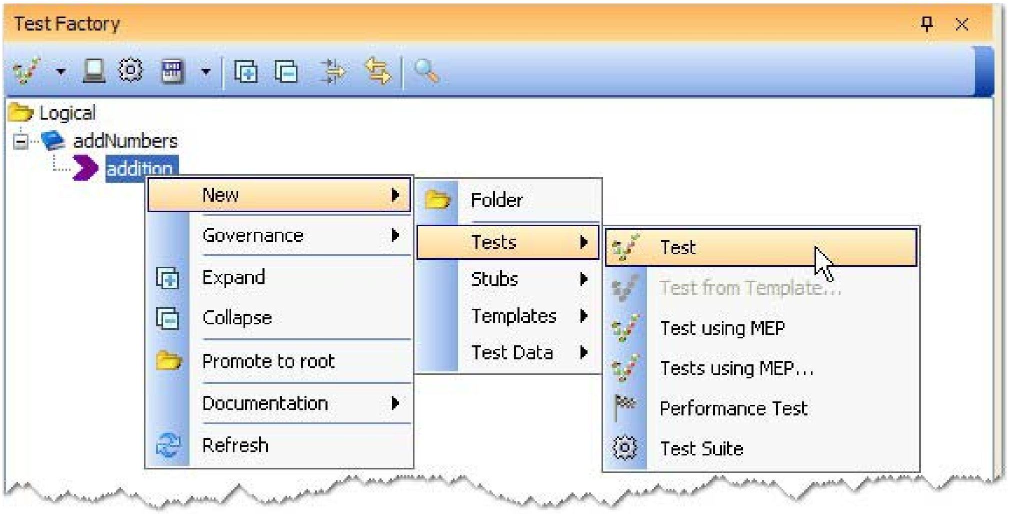
3.3 Creating Tests

Tests can be created in two ways – from scratch or with a wizard that automatically creates tests for you based on the containing operation's message exchange pattern (MEP). Both options will be illustrated in the following sections.

3.3.1 Creating Tests from Scratch

Follow the steps below to create (empty) tests from scratch:

1. Open the Test Factory perspective (**F10**).
2. Right-click on the **addition** operation and select **New > Tests > Test**.



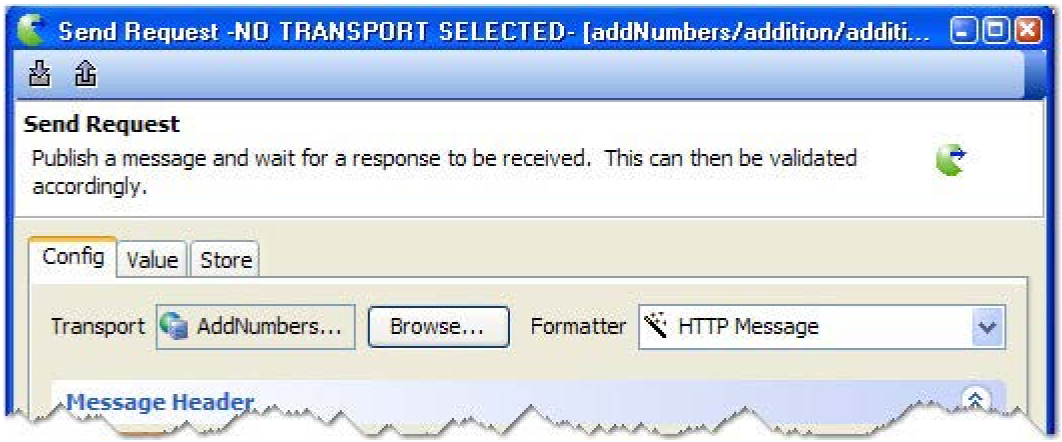
3. When prompted, name the test "additionCheck", then click **OK**.
4. Double-click the new test in the Test Factory tree to open it for editing.
5. In the additionCheck test editor on the right-hand side of the workspace, right-click the **Test Steps** phase and select **New > Messaging > Send Request**.

   A **Send Request** and **Receive Reply** action should be created.

NOTE: Actions placed in the **Initialise** and **Tear-down** sections of a test will run, but these phases are only executed once when running multiple iterations of a test (at test startup and completion). These sections are typically used for one-time test operations, such as clearing database tables or moving files.

6. Double-click the **Send Request** action to open it.

7. Ensure that the Transport is set to **AddNumbersPort** and the Formatter to **HTTP Message**.



Send Request -NO TRANSPORT SELECTED- [addNumbers/addition/additi...

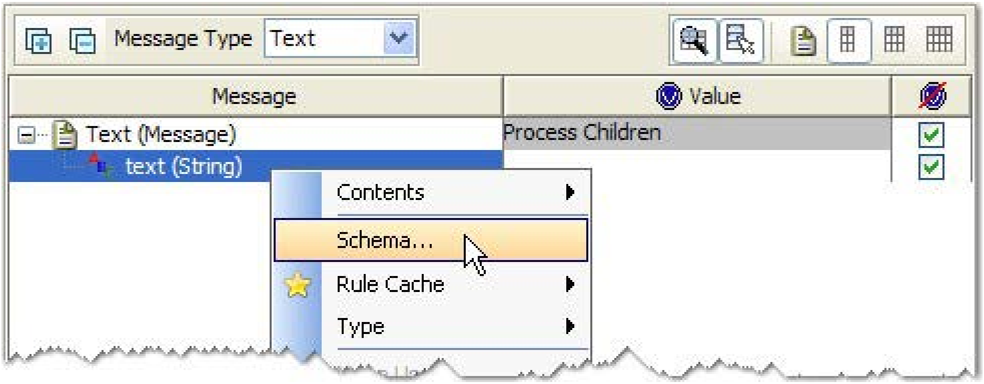
Send Request

Publish a message and wait for a response to be received. This can then be validated accordingly.

Transport AddNumbers...

Formatter HTTP Message

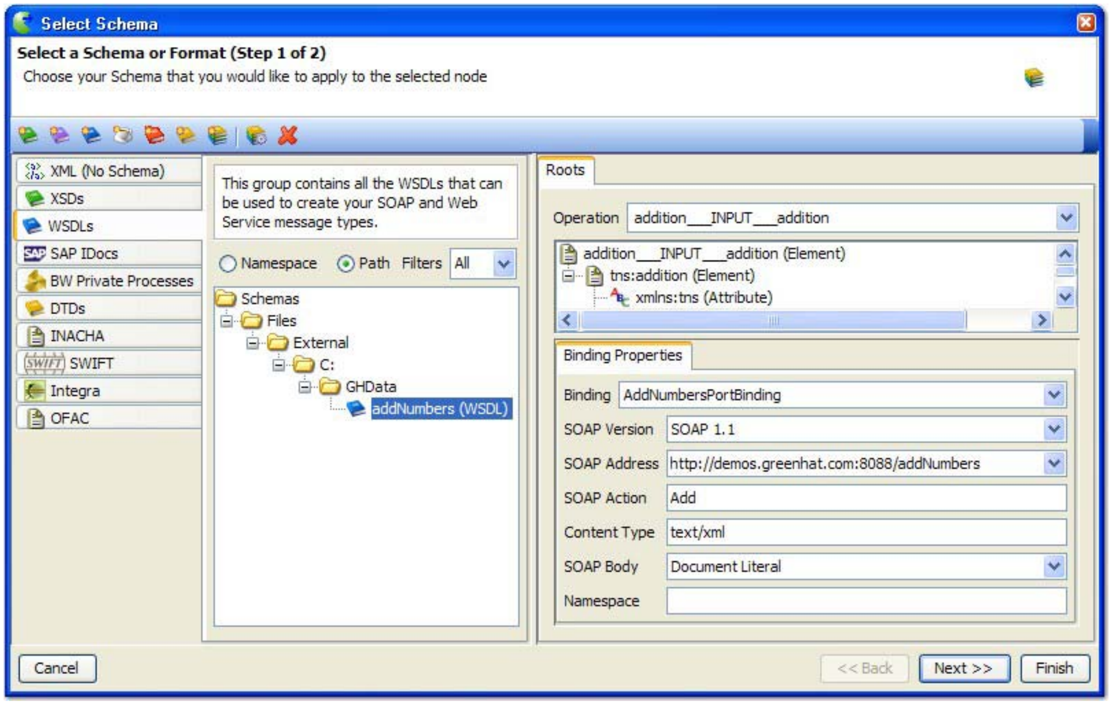
8. In the message configuration area, ensure that the **Message Type** is set to **Text**, then right-click on the **text (String)** node and select the **Schema** option.



Message Type Text

Message	Value
Text (Message)	Process Children
text (String)	

9. In the first dialog of the **Select Schema** wizard, select the **WSDL** tab, select the **AddNumbers** WSDL and select the **addition_INPUT_addition** operation, then click **Next** to proceed.



Select Schema

Select a Schema or Format (Step 1 of 2)

Choose your Schema that you would like to apply to the selected node

This group contains all the WSDLs that can be used to create your SOAP and Web Service message types.

○ Namespace ● Path Filters All

Operation: addition___INPUT___addition

Binding: AddNumbersPortBinding

SOAP Version: SOAP 1.1

SOAP Address: http://demos.greenhat.com:8088/addNumbers

SOAP Action: Add

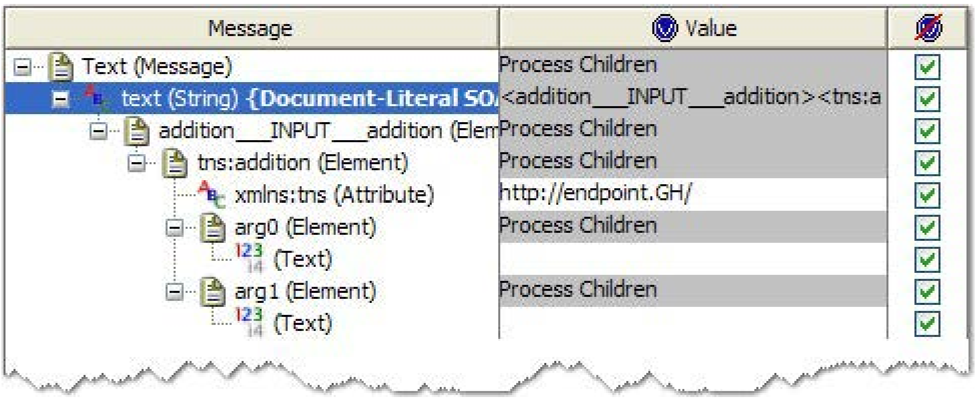
Content Type: text/xml

SOAP Body: Document Literal

Namespace:

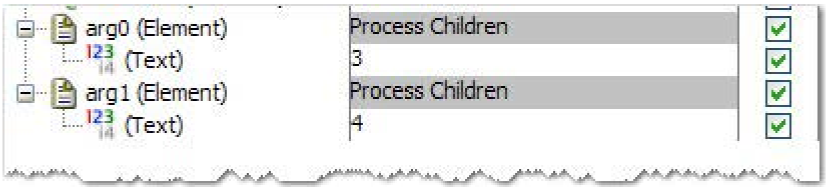
10. In the next window, ensure the **Include Text nodes** option is enabled (ticked) in the **Content** pane, then click **Finish**.

The structure and content of the message body will be updated according to the selected schema.



Message	Value	
Text (Message)	Process Children	☑
text (String) {Document-Literal SO	<addition___INPUT___addition><tns:a	☑
addition___INPUT___addition (Elem	Process Children	☑
tns:addition (Element)	Process Children	☑
xmlns:tns (Attribute)	http://endpoint.GH/	☑
arg0 (Element)	Process Children	☑
(Text)		☑
arg1 (Element)	Process Children	☑
(Text)		☑

11. Set the values of the **arg0** and **arg1** text nodes to 3 and 4 (double-click the empty field under the **Value** column and type in the values).



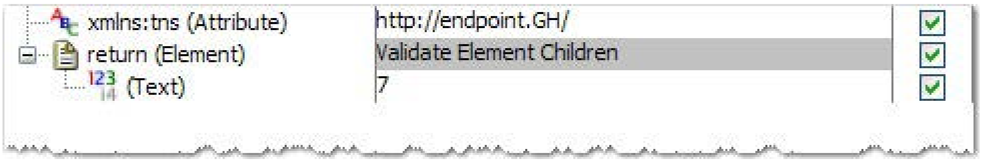
arg0 (Element)	Process Children	☑
(Text)	3	☑
arg1 (Element)	Process Children	☑
(Text)	4	☑

12. Click **OK** to save your changes and close the editor.

13. Double-click the **Receive Reply** action to open it and ensure **Reply to** is set to **Send Request 1**.

14. Apply the same schema to the message (as described in steps 8 through 10), except this time choose **addition__output__additionResponse** operation.

15. Enter "7" in the **return** element text node, which is the sum of **arg0** and **arg1**.



xmlns:tns (Attribute)	http://endpoint.GH/	☑
return (Element)	Validate Element Children	☑
(Text)	7	☑

16. Click **OK** to save your changes and close the editor.

17. Save the project (click 💾 or press **Ctrl + S**) and the test is now ready to run.

3.3.2 Creating Tests using MEP

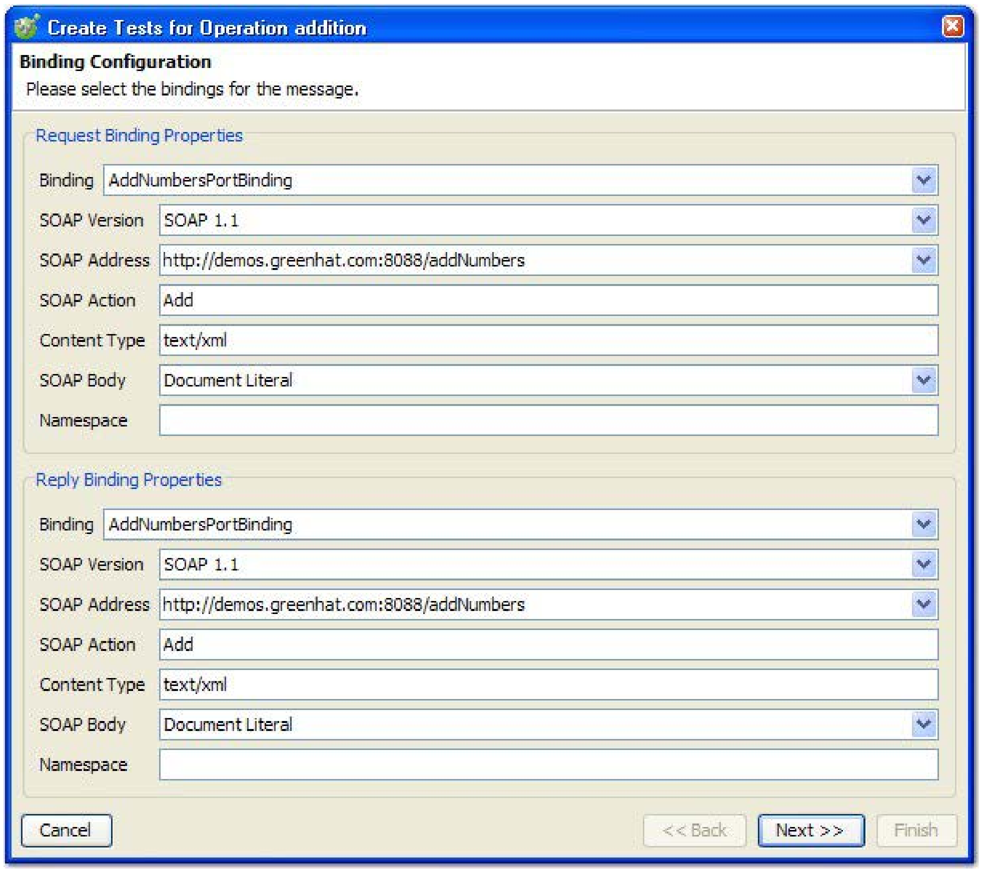
An operation's Message Exchange Pattern (MEP) defines the pattern, schema, and binding for messages that are exchanged by the operation. This information is used by the MEP wizard to quickly create tests.

You can create a single test using an operation's MEP, or multiple tests. The singular option quickly creates one test that you can populate and run, while the multiple option creates a number of structural tests according to content combinations that are based on the schemas defined in the WSDL. The MEP wizard can populate the tests with a variety of test data (for example, random data values, constants, regular expressions, and so on) that is defined by the schema.

The following example illustrates how to use the MEP wizard.

1. Open the Test Factory perspective (**F10**).
2. Right-click on the **addition** operation and select **New > Tests > Tests using MEP**.

The **Create Tests** wizard is displayed. The first window of the wizard enables you to configure the binding properties for the request and reply messages.



Create Tests for Operation addition

Binding Configuration

Please select the bindings for the message.

Request Binding Properties

Binding: AddNumbersPortBinding

SOAP Version: SOAP 1.1

SOAP Address: http://demos.greenhat.com:8088/addNumbers

SOAP Action: Add

Content Type: text/xml

SOAP Body: Document Literal

Namespace:

Reply Binding Properties

Binding: AddNumbersPortBinding

SOAP Version: SOAP 1.1

SOAP Address: http://demos.greenhat.com:8088/addNumbers

SOAP Action: Add

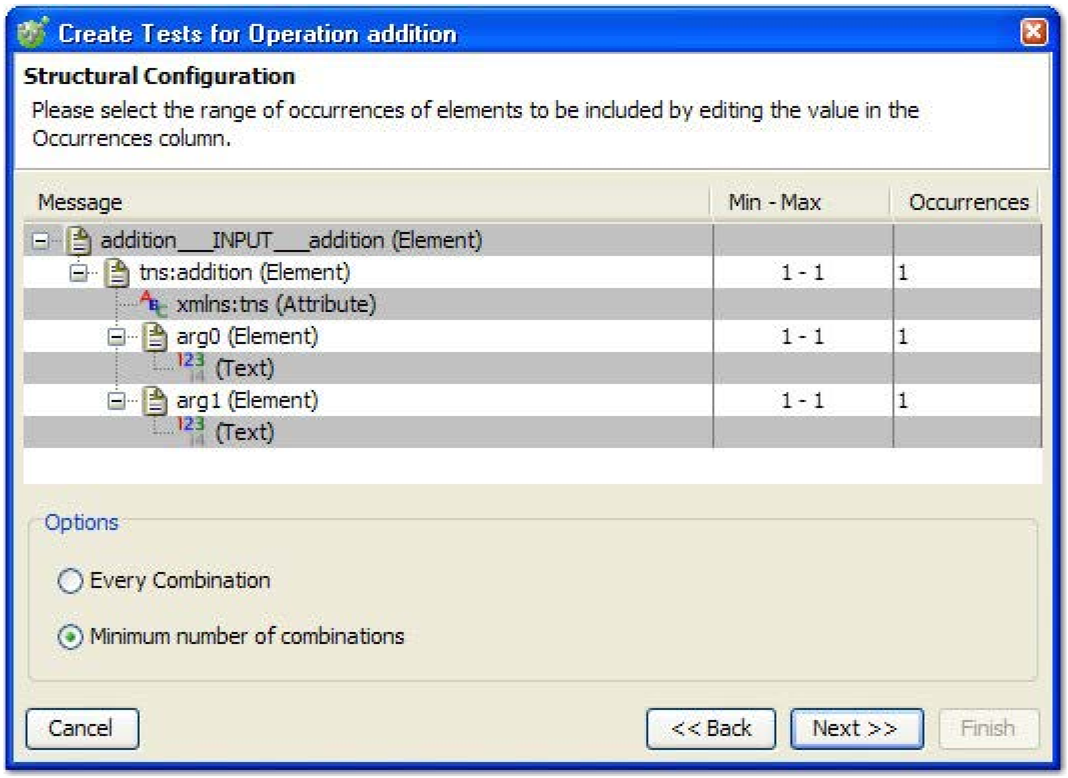
Content Type: text/xml

SOAP Body: Document Literal

Namespace:

3. Click **Next** to proceed as we do not need to change the message bindings.

   The **Structural Configuration** screen enables you to set the number of occurrences for each message element (0 through *n*) and select the desired combination option for generating tests.



Create Tests for Operation addition

Structural Configuration

Please select the range of occurrences of elements to be included by editing the value in the Occurrences column.

Message	Min - Max	Occurrences
addition___INPUT___addition (Element)		
tns:addition (Element)	1 - 1	1
xmlns:tns (Attribute)		
arg0 (Element)	1 - 1	1
(Text)		
arg1 (Element)	1 - 1	1
(Text)		

Options

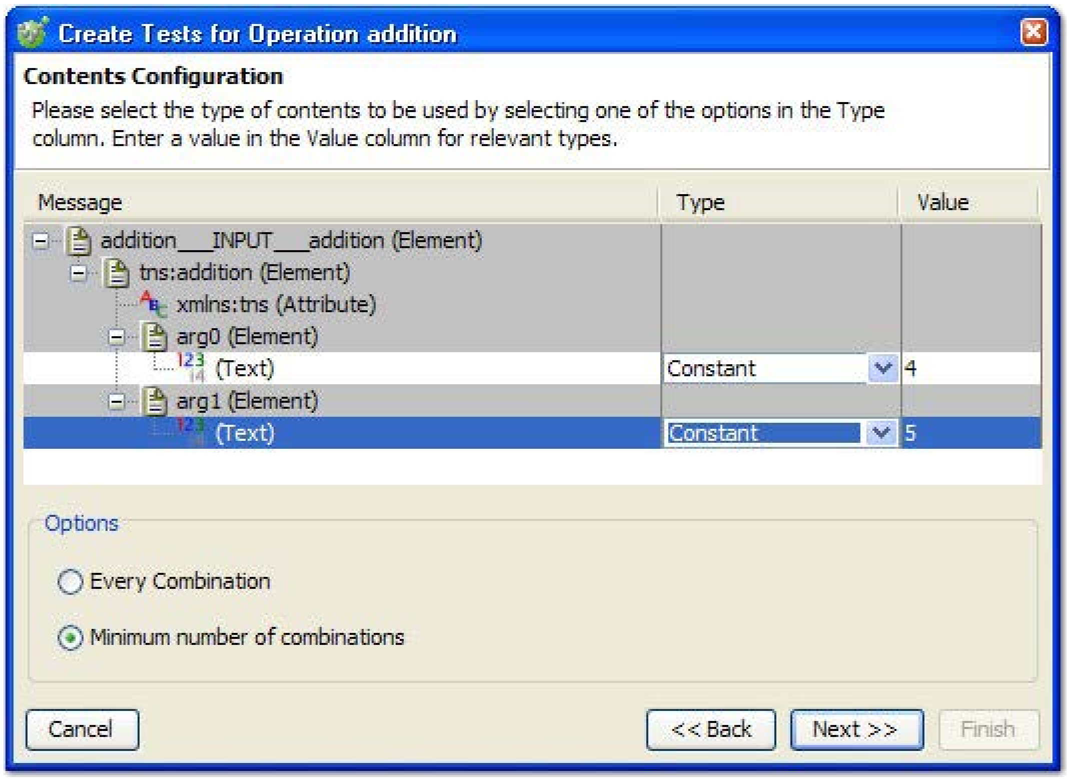
○ Every Combination

● Minimum number of combinations

- If **Minimum number of combinations** is selected, at least one unique element will be generated in a message for each occurrence, but only enough tests will be generated to satisfy the highest single occurrence value (that is, if the most occurrences for any element is three, then three tests would be generated).
- If **Every Combination** is selected, a test will be created so that every unique combination of occurrences is satisfied (that is, if one element specifies two occurrences and another specifies three, then six tests would be generated).

4. Leave the default number of occurrences and combination options, then click **Next**.

The **Contents Configuration** screen allows you to select the type of content to be populated in the text nodes of the message.



Contents Configuration

Please select the type of contents to be used by selecting one of the options in the Type column. Enter a value in the Value column for relevant types.

Message	Type	Value
addition___INPUT___addition (Element)		
tns:addition (Element)		
xmlns:tns (Attribute)		
arg0 (Element)		
(Text)	Constant	4
arg1 (Element)		
(Text)	Constant	5

Options

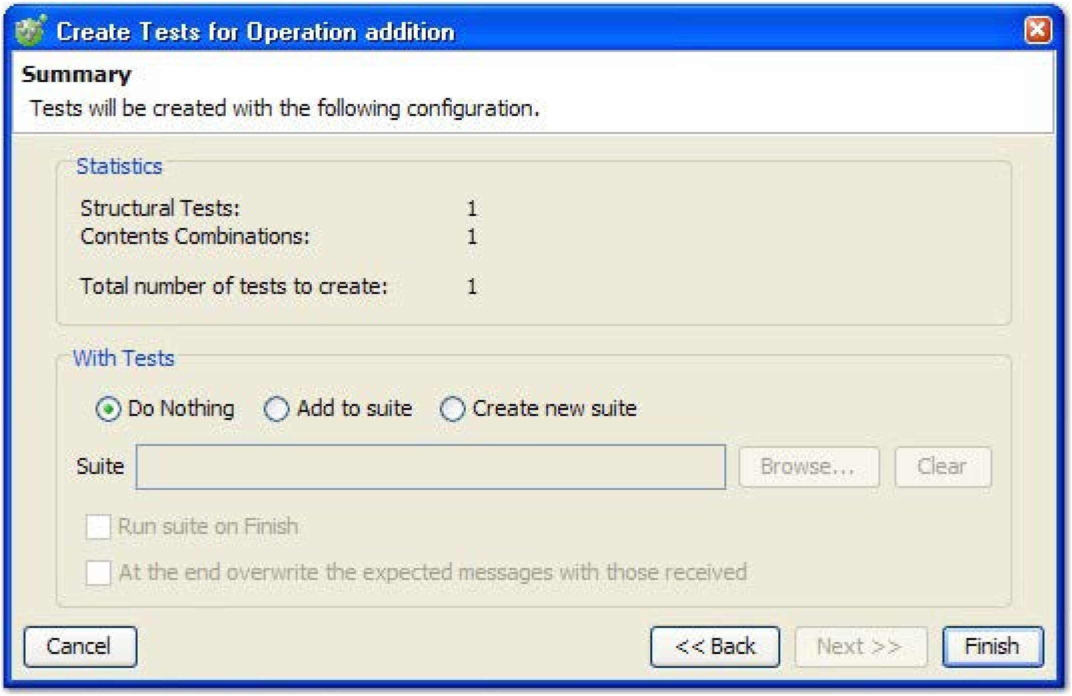
○ Every Combination

● Minimum number of combinations

5. For the **arg0** and **arg1** text fields, change the type to **Constant** and enter values of “4” and “5”, respectively.

6. Click **Next** to proceed.

A summary is displayed, showing the number of tests that the wizard will create.



Create Tests for Operation addition

Summary

Tests will be created with the following configuration.

Statistics

Structural Tests:	1
Contents Combinations:	1
Total number of tests to create:	1

With Tests

● Do Nothing ○ Add to suite ○ Create new suite

Suite

□ Run suite on Finish

□ At the end overwrite the expected messages with those received

7. For this example, select **Do Nothing** under **With Tests** to simply create the tests.

NOTE: The MEP test can be added to an existing test suite or added to a new test suite, as described in Create a Test Suite with the MEP Test Wizard.

8. Click **Finish** to close the wizard and create the tests as specified. You should see a new test called "Test" with pre-populated test data in the messages.

9. Rename the test "additionCheckMEP" (right-click and select **Rename**, or select the test and press **F2**). You can double-click the new test to open it for editing.

NOTE: The response message has not been populated with an expected value. The test repair wizard will be demonstrated in the next section to remedy this.

3.4 Running the Tests

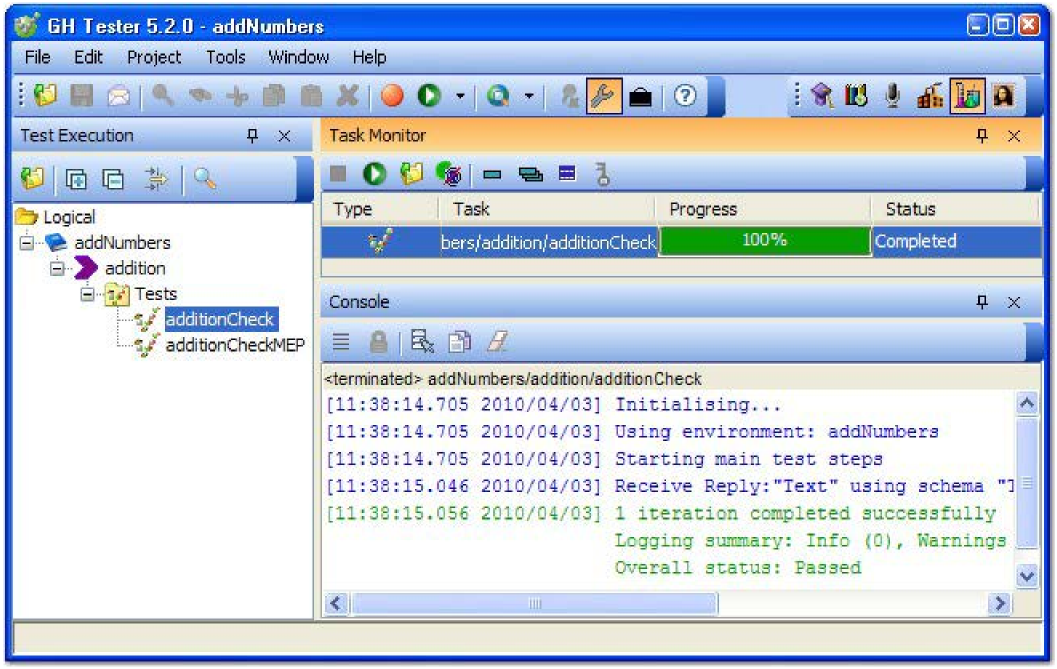
Tests – regardless of how they are created – are executed in the Test Lab perspective. This section describes how to run the two tests that were created previously.

3.4.1 Running the additionCheck Test

Follow the steps below to run the test made from scratch, **additionCheck**:

1. Navigate to the original WSDL URL and ensure that the sample web service is still running.
2. Open the Test Lab perspective (**F11**) and select the **additionCheck** test.
3. Right-click the test and select **Run**, or press **F5**.

While the test is running, the Task Monitor and Console will provide execution details. Once finished, the **Progress** bar should turn green and show "100%", and the **Status** of the test should be "Complete". In the Console, you should see a message indicating that the test has passed.



Type	Task	Progress	Status
✓	bers/addition/additionCheck	100%	Completed

```
<terminated> addNumbers/addition/additionCheck
[11:38:14.705 2010/04/03] Initialising...
[11:38:14.705 2010/04/03] Using environment: addNumbers
[11:38:14.705 2010/04/03] Starting main test steps
[11:38:15.046 2010/04/03] Receive Reply:"Text" using schema "]
[11:38:15.056 2010/04/03] 1 iteration completed successfully
                          Logging summary: Info (0), Warnings
                          Overall status: Passed
```

The Task Monitor shows the status of all current and previous tests, and the Console displays the detailed execution results for the currently selected run. If desired, you can remove selected test runs from the Task Monitor with the Delete icon ▭, or clear all runs with the Delete All icon ▭.

3.4.2 Running the additionCheckMEP Test

Follow the steps below to run the test made with the wizard, **additionCheckMEP**:

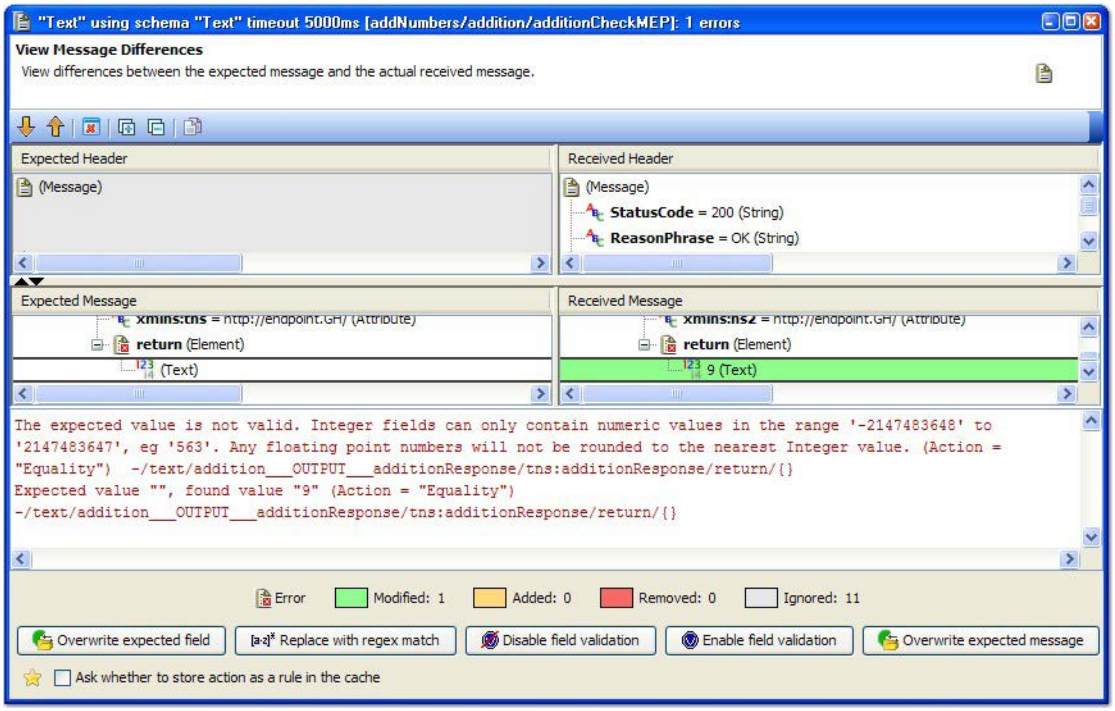
1. Run the test in the same way as described in Running the additionCheck Test.

   The test should fail (that is, not pass validation) because the value returned in the response does not match the expected result – remember this value was left blank. The following errors should be displayed in the Console for the Receive Reply action:

   ```
   The expected value is not valid. Integer fields can only contain...
   Expected value "", found value "9" (Action = "Equality")
   ```

   The test, however, can be repaired so that it passes the next time it is run.

2. Click the *"Expected value..."* line to display the **View Message Difference** window.



"Text" using schema "Text" timeout 5000ms [addNumbers/addition/additionCheckMEP]: 1 errors

View Message Differences

View differences between the expected message and the actual received message.

Expected Header	Received Header
(Message)	(Message) **StatusCode** = 200 (String) **ReasonPhrase** = OK (String)

Expected Message	Received Message
xmlns:tns = http://endpoint.GH/ (Attribute)	**xmlns:ns2** = http://endpoint.GH/ (Attribute)
return (Element)	**return** (Element)
(Text)	9 (Text)

```
The expected value is not valid. Integer fields can only contain numeric values in the range '-2147483648' to
'2147483647', eg '563'. Any floating point numbers will not be rounded to the nearest Integer value. (Action =
"Equality")  -/text/addition___OUTPUT___additionResponse/tns:additionResponse/return/{}
Expected value "", found value "9" (Action = "Equality")
-/text/addition___OUTPUT___additionResponse/tns:additionResponse/return/{}
```

Error 🟩 Modified: 1 🟨 Added: 0 🟥 Removed: 0 □ Ignored: 11

3. Select the highlighted line in the expected or received message to see error details.

4. From the available options, select **Overwrite expected field** and close the window.

   NOTE: For information about the available repair options, refer to *IBM Rational Integration Tester Reference Guide.*

5. Run the test again by clicking the **Launch again** button ▶ in the Task Monitor and the test should pass.

3.5 Creating Test Suites

Test suites are used to run sets of tests and test suites. For example, if you have a set of regression tests to run, you can group and run them together in a single test suite.

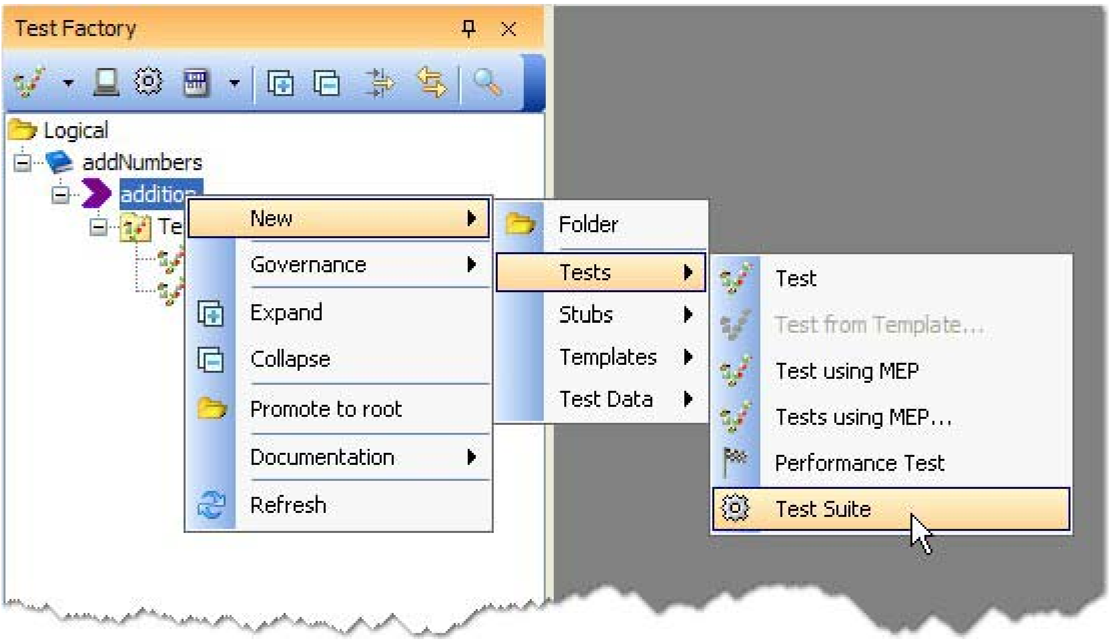
The following sections illustrate two different ways to create test suites:

- Create a Test Suite Manually
- Create a Test Suite with the MEP Test Wizard

3.5.1 Create a Test Suite Manually

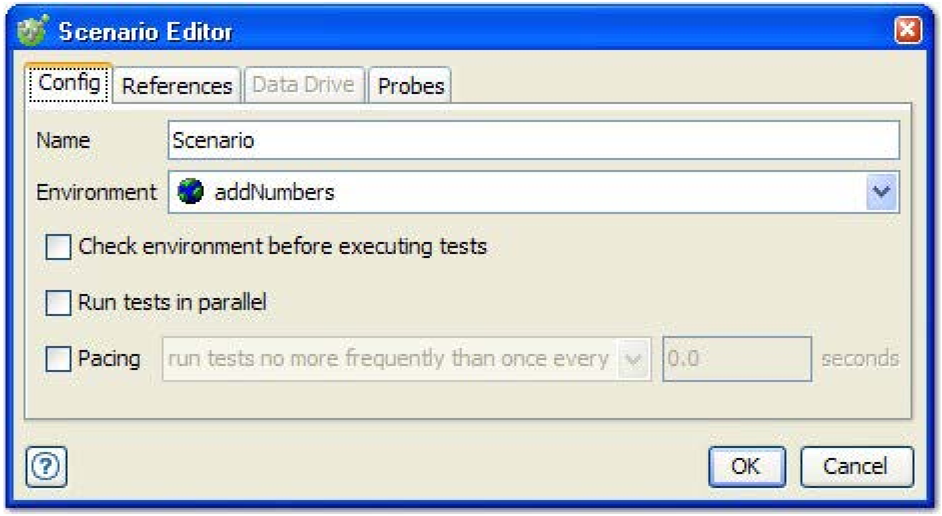
The steps below illustrate how to create a very simple test suite that will run both of the tests that were created earlier.

1. Open the Test Factory perspective (**F10**).
2. Right-click on the **addition** operation and select **New > Tests > Test Suite**.



3. When prompted, name the suite "additionCheckTestSuite", then click **OK**.
4. Double-click the new test suite to open it for editing.
5. In the suite editing window to the right, double-click **Scenario**.

The **Scenario Editor** is displayed.



Name: Scenario

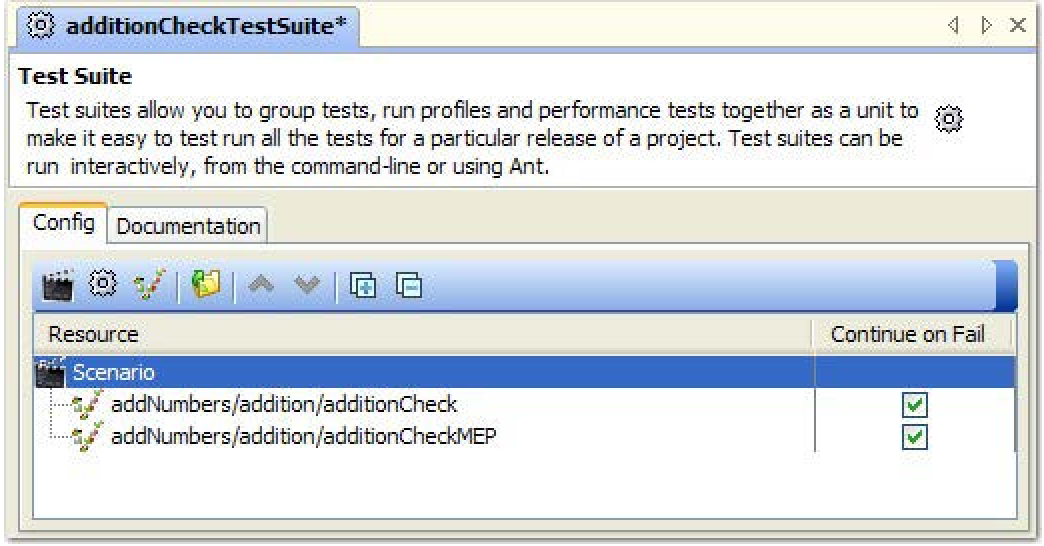
Environment: addNumbers

□ Check environment before executing tests

□ Run tests in parallel

□ Pacing   run tests no more frequently than once every   0.0   seconds

6. Under the **Config** tab, select the **addNumbers** environment and click **OK**.
7. Add the **additionCheck** and **additionCheckMEP** tests to the suite by dragging them from the **Tests** folder, or click the **Add Tests** icon and select the tests from the project resource dialog.



Test Suite

Test suites allow you to group tests, run profiles and performance tests together as a unit to make it easy to test run all the tests for a particular release of a project. Test suites can be run interactively, from the command-line or using Ant.

Resource	Continue on Fail
Scenario	
addNumbers/addition/additionCheck	☑
addNumbers/addition/additionCheckMEP	☑

8. Save the project (click or press **Ctrl + S**) and run the suite (right-click and select **Run**, or select and press **F5**).

   The Test Lab is opened and the suite is executed. The suite will run both tests, and you should see all three items (the suite and the two tests) pass.

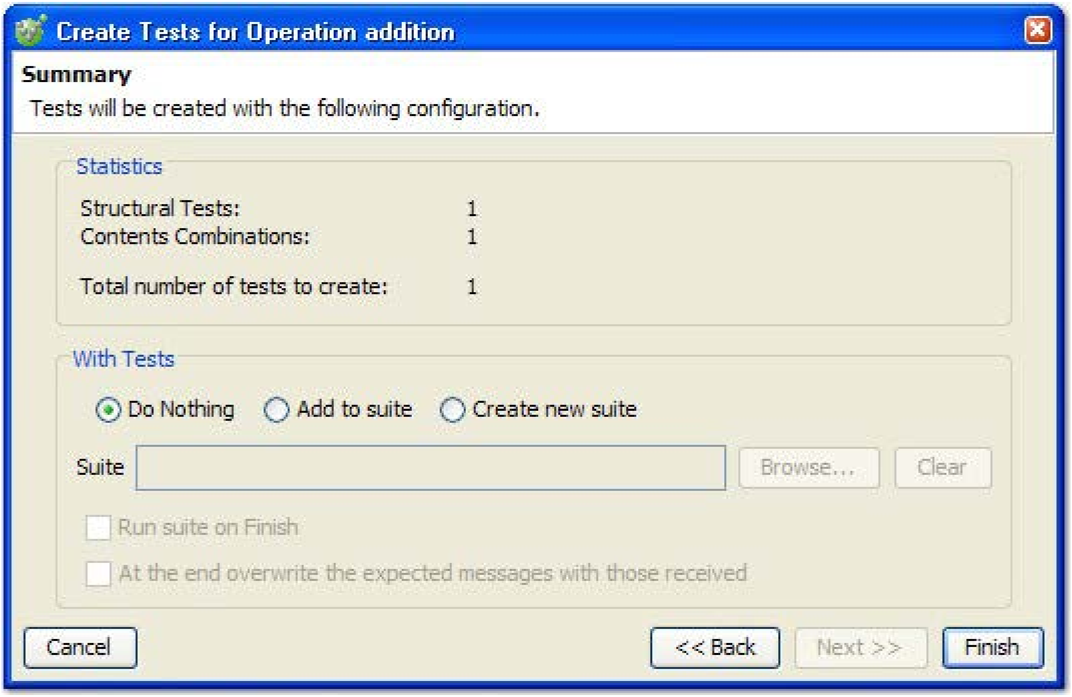
3.5.2 Create a Test Suite with the MEP Test Wizard

Earlier (in Creating Tests using MEP) we created a test based on an operation's message exchange pattern using the MEP wizard. In the last step of the wizard, you have the option to simply create the tests, add the tests to an existing test suite, or create a new test suite that contains the created tests.

The following steps illustrate how to create a new test suite using one or more tests created from MEP:

1. Return to Creating Tests using MEP and carry out steps 1 through 6, then return to this section and proceed.

   The MEP wizard summary should be displayed, showing the number of tests that the wizard will create.



Create Tests for Operation addition

Summary

Tests will be created with the following configuration.

Statistics

Structural Tests: 1
Contents Combinations: 1

Total number of tests to create: 1

With Tests

● Do Nothing ○ Add to suite ○ Create new suite

Suite

□ Run suite on Finish

□ At the end overwrite the expected messages with those received

2. Select **Create new suite** under **With Tests** to create a new test suite that contains the test being created (**Do Nothing** simply creates the tests from the MEP wizard and **Add to suite** will add the new tests to an existing test suite, selected by clicking **Browse** and selecting the desired suite).

3. Provide a name for the new suite in the **Suite** field.

4. Enable any of the desired execution options under the suite name, as follows.

    - The **Run suite on Finish** option will execute a new or existing test suite when the MEP wizard closes.

    - The **At the end...** option will overwrite expected messages in any executed tests with the messages that are received.

5. Click **Finish** to close the wizard and create the tests and test suite as specified. You should see a new test called “Test” and a new suite (with the name that you specified) containing the new test.

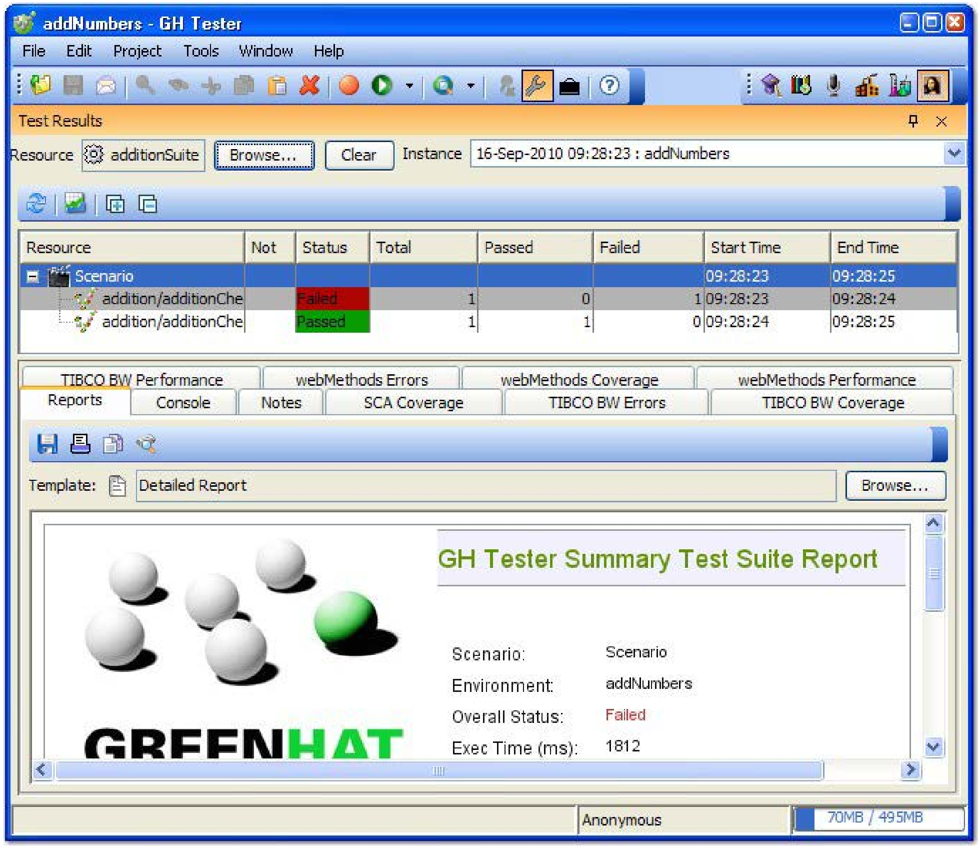
3.6 Viewing Test Reports

The Results Gallery enables you to view summary and detailed coverage, error, and performance reports for tests and suites that have been executed in the project.

NOTE: Test suite results can only be stored and viewed if the connection to the project database has been established properly. For more information, refer to *IBM Rational Integration Tester Installation Guide.*

Follow the steps below to view reports for the suite that was executed earlier.

1. Open to the Results Gallery perspective (**F12**).
2. The **additionCheckTestSuite** suite should already be selected in the **Resource** field. If not, click **Browse** and select it from the project resource dialog.
3. If the suite has been executed more than once, you can select the desired execution instance from the **Instance** field.
4. Ensure that the **Reports** tab is selected to view a summary report of the selected suite and execution instance.



Resource: additionSuite

Instance: 16-Sep-2010 09:28:23 : addNumbers

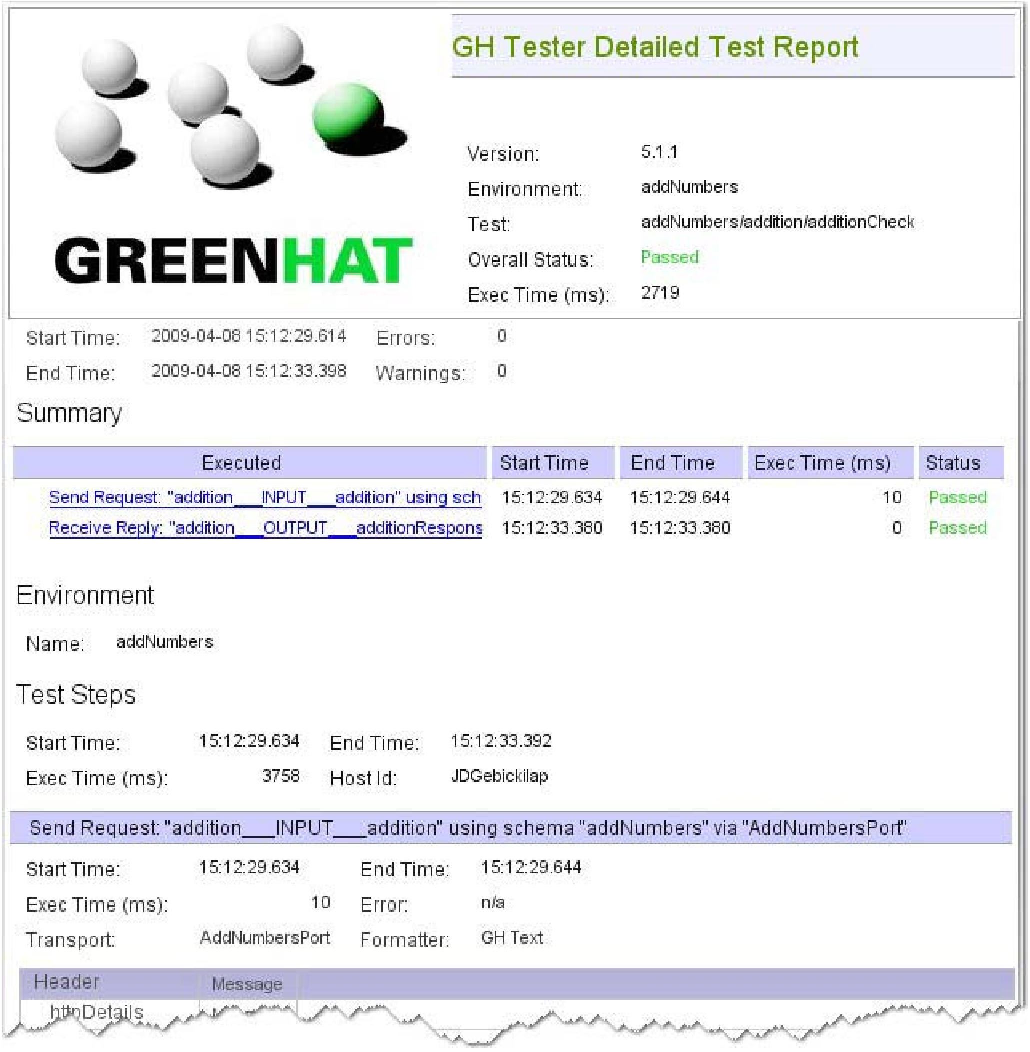
Resource	Not	Status	Total	Passed	Failed	Start Time	End Time
Scenario						09:28:23	09:28:25
addition/additionChe		Failed	1	0	1	09:28:23	09:28:24
addition/additionChe		Passed	1	1	0	09:28:24	09:28:25

Template: Detailed Report

GH Tester Summary Test Suite Report

Scenario:	Scenario
Environment:	addNumbers
Overall Status:	Failed
Exec Time (ms):	1812

5. Under the **Resource** column in the suite summary table, select the **additionCheck** test to view a detailed report of its results.



GH Tester Detailed Test Report

GREENHAT

Version:	5.1.1
Environment:	addNumbers
Test:	addNumbers/addition/additionCheck
Overall Status:	Passed
Exec Time (ms):	2719

Start Time:	2009-04-08 15:12:29.614	Errors:	0
End Time:	2009-04-08 15:12:33.398	Warnings:	0

Summary

Executed	Start Time	End Time	Exec Time (ms)	Status
Send Request: "addition___INPUT___addition" using sch	15:12:29.634	15:12:29.644	10	Passed
Receive Reply: "addition___OUTPUT___additionRespons	15:12:33.380	15:12:33.380	0	Passed

Environment

Name: addNumbers

Test Steps

Start Time:	15:12:29.634	End Time:	15:12:33.392
Exec Time (ms):	3758	Host Id:	JDGebickilap

Send Request: "addition___INPUT___addition" using schema "addNumbers" via "AddNumbersPort"

Start Time:	15:12:29.634	End Time:	15:12:29.644
Exec Time (ms):	10	Error:	n/a
Transport:	AddNumbersPort	Formatter:	GH Text

Header	Message
httpDetails	

The detailed test report is displayed, including the overall status of the test, any errors and warnings that were generated, execution times, and the detailed results of the Send Request and Receive Reply actions in the test.

Data Validation

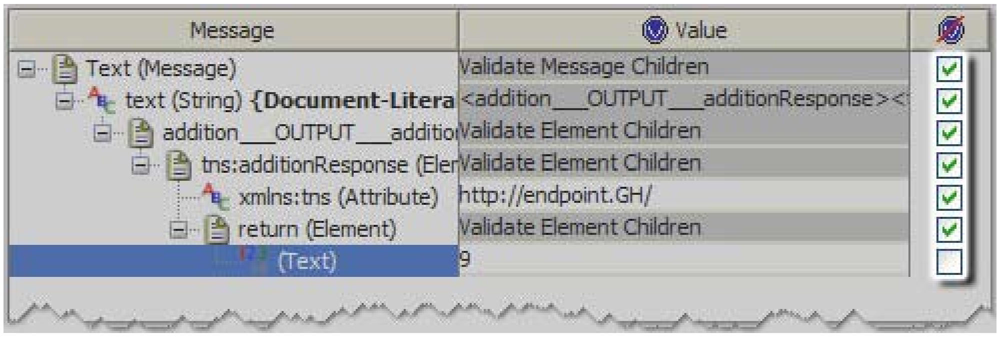
Contents

This chapter provides an overview of data validation in Rational Integration Tester, how it works, and how to configure it.

4.1 Overview

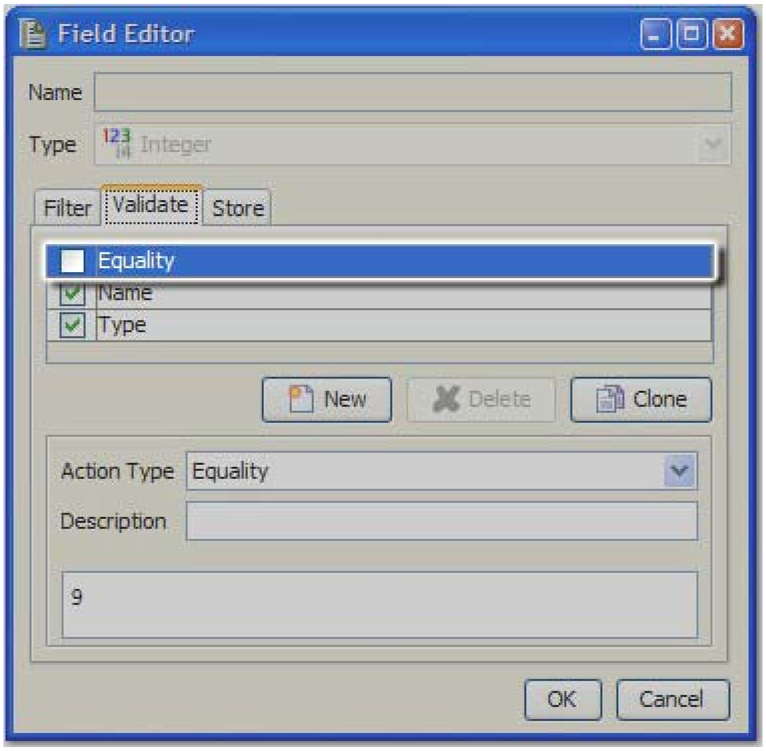
Data validation has been used throughout the exercises that have been illustrated in this guide. When validation is enabled for a selected field, the returned value is matched against the expected value.

Validation can be enabled or disabled for a message field by selecting or clearing the check box in the ⊘ column for that field, under the **Config** or **Assert** tab of the message editor.



Message	Value	⊘
Text (Message)	Validate Message Children	☑
text (String) {Document-Litera	<addition___OUTPUT___additionResponse><:	☑
addition___OUTPUT___additio	Validate Element Children	☑
tns:additionResponse (Eler	Validate Element Children	☑
xmlns:tns (Attribute)	http://endpoint.GH/	☑
return (Element)	Validate Element Children	☑
(Text)	9	□

Validation can also be controlled by enabling or disabling the **Equality** option, under the **Validate** tab, of the field editor (available by double-clicking a message element).



Field Editor

Name:

Type: Integer

Filter | Validate | Store

□ Equality
☑ Name
☑ Type

Action Type: Equality

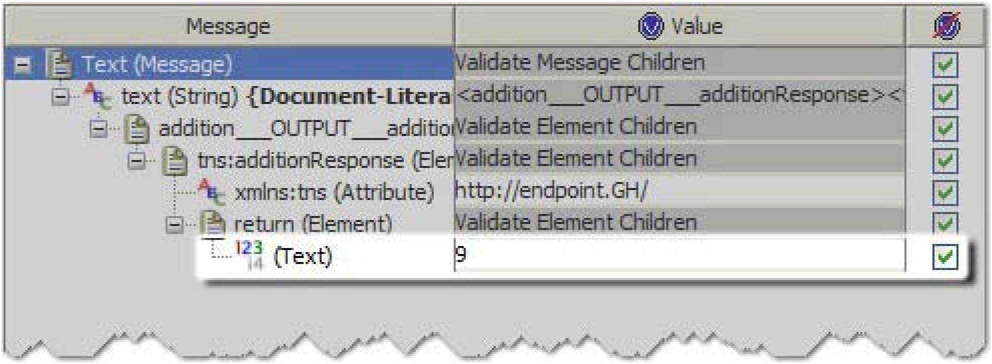
Description:

9

4.2 Exercise

The following exercises illustrates how validation can be enabled or disabled for individual message fields, and the effect it has on whether or not the test passes.

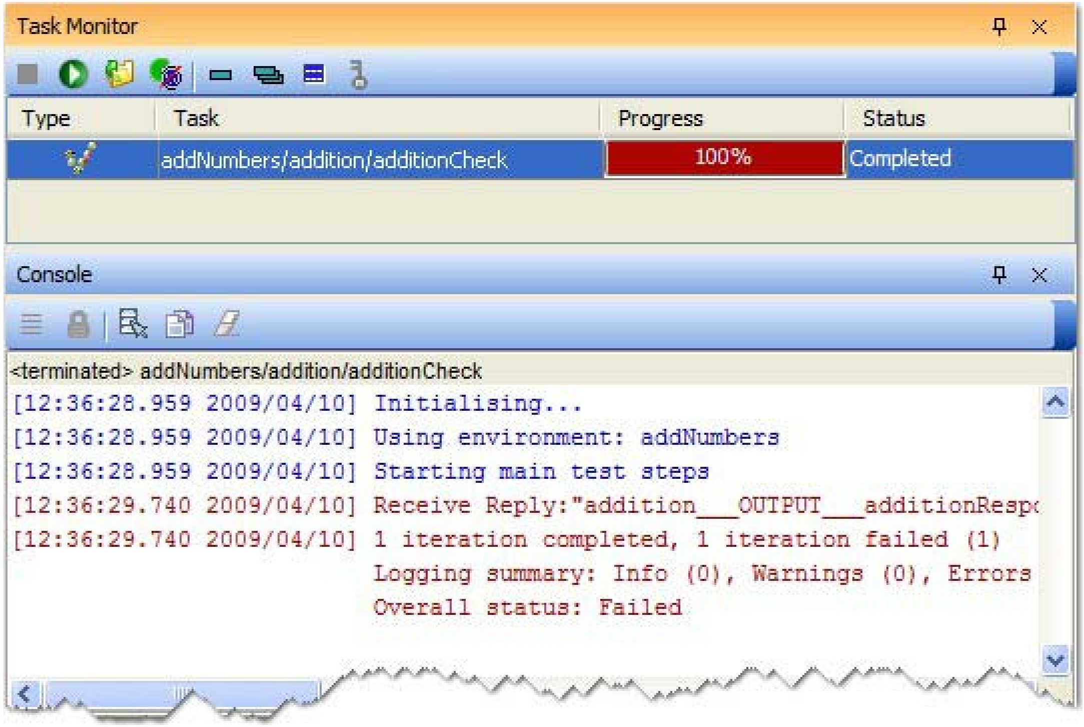
1. Open the Test Factory perspective (**F10**)
2. Double-click the **additionCheck** test to open it for editing in the Test Factory perspective (**F10**).
3. Open the Receive Reply action and note the checked box in the validation column for the **return** text node.



Message	Value	
Text (Message)	Validate Message Children	☑
text (String) **{Document-Litera**	<addition___OUTPUT___additionResponse><	☑
addition___OUTPUT___additio	Validate Element Children	☑
tns:additionResponse (Eler	Validate Element Children	☑
xmlns:tns (Attribute)	http://endpoint.GH/	☑
return (Element)	Validate Element Children	☑
(Text)	9	☑

4. Change the value of the node to any other number (that is, something that does not equal the sum of **arg0** and **arg1** in the Send Request action), then click **OK**.
5. Save the project (click 💾 or press **Ctrl + S**).

6. Run the **additionCheck** test in the **addNumbers** environment and the test should fail due to a validation error, as shown below.



Type	Task	Progress	Status
✓	addNumbers/addition/additionCheck	100%	Completed

```
<terminated> addNumbers/addition/additionCheck
[12:36:28.959 2009/04/10] Initialising...
[12:36:28.959 2009/04/10] Using environment: addNumbers
[12:36:28.959 2009/04/10] Starting main test steps
[12:36:29.740 2009/04/10] Receive Reply:"addition___OUTPUT___additionRespo
[12:36:29.740 2009/04/10] 1 iteration completed, 1 iteration failed (1)
                          Logging summary: Info (0), Warnings (0), Errors
                          Overall status: Failed
```

The specific error for the Receive Reply action would be something like the following:

```
Expected value "99", found value "7" (Action = "Equality")
```

7. Open the Receive Reply action again and clear the check box next to the same field to disable validation, or double-click the **(Text)** node and disable the **Equality** action in the field editor.

NOTE: If desired, as illustrated previously, you can disable validation in the **View Message Differences** dialog (see Running the additionCheckMEP Test).

8. Click **OK** in the Receive Reply editor to close it and save your changes.

9. Run the test again in the same environment and it should pass.

NOTE: In this example, the **Equality** action was used on a message field to validate the response, but numerous actions are available (for example, Xpath, Regex, Schema, Length, and custom validation).For more information about other validation options, refer to *IBM Rational Integration Tester Reference Guide.*

Test Data Sets

Contents

This chapter provides an overview of how to create data sets that can be used by Rational Integration Tester test resources to load test data dynamically.

5.1 Overview

So far, all of the data used in our example tests has been hard-coded. A more practical approach to testing, however, would be to decouple the tests from the data. One of the more common requirements of functional testing is to run a test through several iterations with different data being used each time. To do this, you will need to use test data sets. Data sets, along with the Iterate Test Data action, can be used to map test data into tags that are used to populate data fields in a test.

Test data can be supplied to Rational Integration Tester from four types of sources:

- Files (comma separated and fixed-width)
- Microsoft Excel spreadsheets
- Database queries
- Directory (a set of files containing XML or other data)

5.2 Creating Data Sources

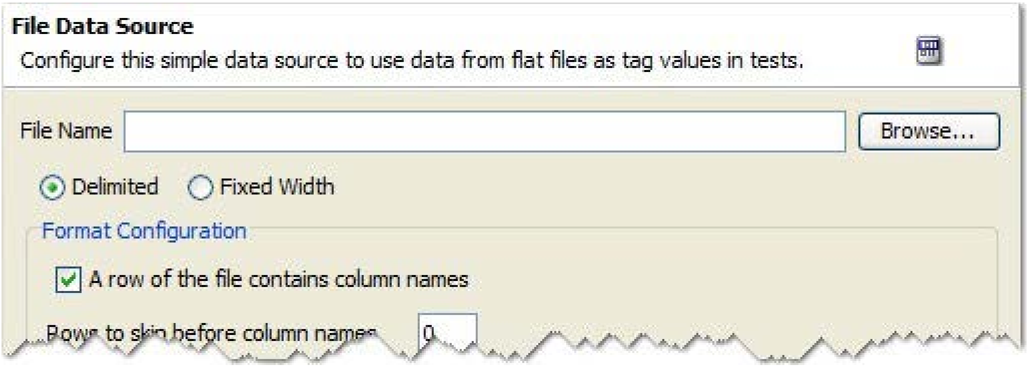
For our example, we will create a comma separated file that contains the data we will utilize in the **additionCheck** test.

1. Create and open a new text file and enter the following data in it:

   ```
   arg0,arg1
   2,6
   3,3
   1,8
   ```

   The fields in the first line – *arg0* and *arg1* – represent the column names.

2. Save the file as **data.txt** to your local drive.
3. In the Test Factory perspective (**F10**), right-click the **addition** operation and select **New > Test Data > File Data Source** from the context menu.
4. Enter "addNumbers" for the data source name and click **OK**. The properties of the new data source will be opened for editing to the right.



File Data Source

Configure this simple data source to use data from flat files as tag values in tests.

File Name

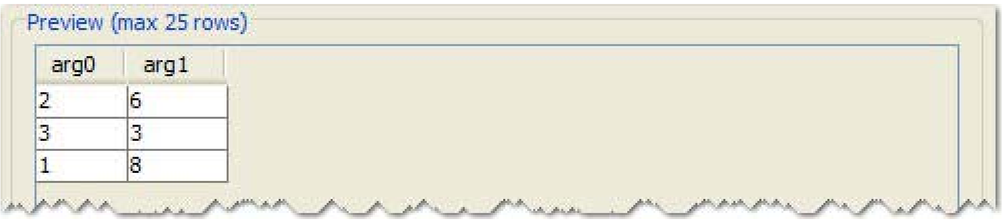
● Delimited ○ Fixed Width

Format Configuration

☑ A row of the file contains column names

Rows to skip before column names 0

5. Click **Browse** next to the **File Name** field and select the data file you just created.
6. Leave all of the other settings as they are and click **Refresh** at the bottom of the data source editor – preview of the data in the selected file is displayed.



Preview (max 25 rows)

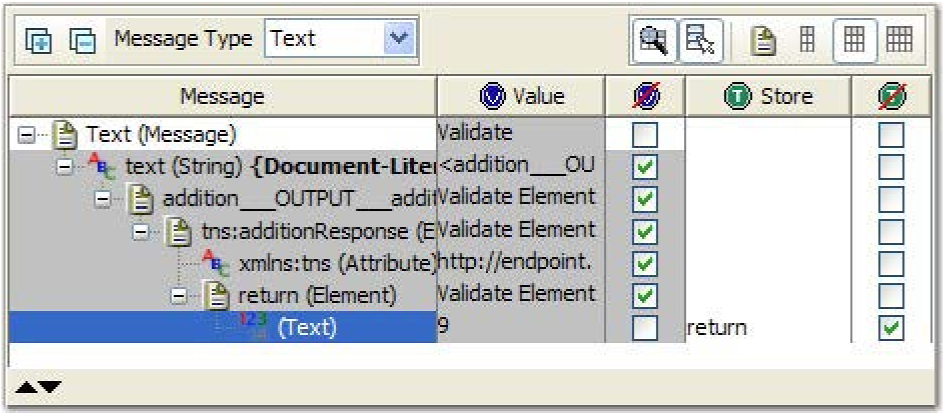
arg0	arg1
2	6
3	3
1	8

7. Click 💾 or press **Ctrl + S** to save the new data source.

5.3 Adding Log Actions to Tests

Before configuring the test to use the new data source, we will add a Log action that will display the results of adding the arguments in the console.

1. In the Test Factory perspective (**F10**), double-click the **additionCheck** test to open it.
2. Right-click the **Receive Reply** action and select **New > Flow > Log** to add a log action.
3. Double-click the **Receive Reply** action and click the **Simple Editing View** ▦ icon in the message editor.
4. Double-click the **Store** field next to the **Text** field (under the **return** element) and enter "return."
5. Press **Enter** when finished.



Message Type: Text

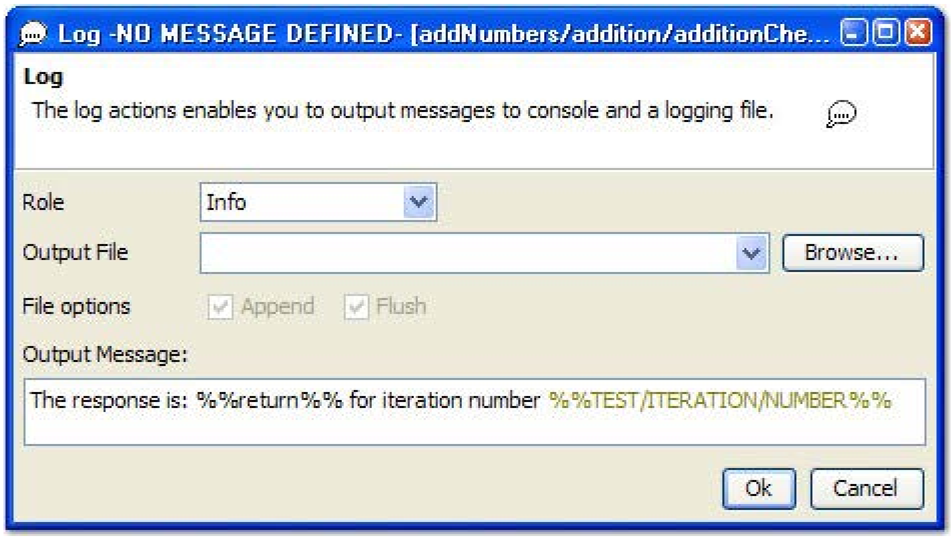
Message	Value		Store	
Text (Message)	Validate	□		□
text (String) {Document-Lite	<addition___OU	☑		□
addition___OUTPUT___addi	Validate Element	☑		□
tns:additionResponse (E	Validate Element	☑		□
xmlns:tns (Attribute)	http://endpoint.	☑		□
return (Element)	Validate Element	☑		□
(Text)	9	□	return	☑

This will create a store action that will store the results of the field in a tag named %%return%%.

6. Click **OK** to apply the change and close the **Receive Reply** action.
7. To log the iterations as they are processed, open the Log step.
8. Enter "The response is: ", then right-click at the end of the line and select **Insert Tag > Test Scope> return**.

9. Now add “for iteration step” to the end of the line, then right-click at the end of the line and select **Insert Tag > Built-In > TEST > ITERATION > NUMBER**.

   The Log step should now contain the following text: The response is %%return%% for iteration step %%TEST/ITERATION/NUMBER%%



Log -NO MESSAGE DEFINED- [addNumbers/addition/additionChe...

Log

The log actions enables you to output messages to console and a logging file.

Role: Info

Output File:

File options: ☑ Append ☑ Flush

Output Message:

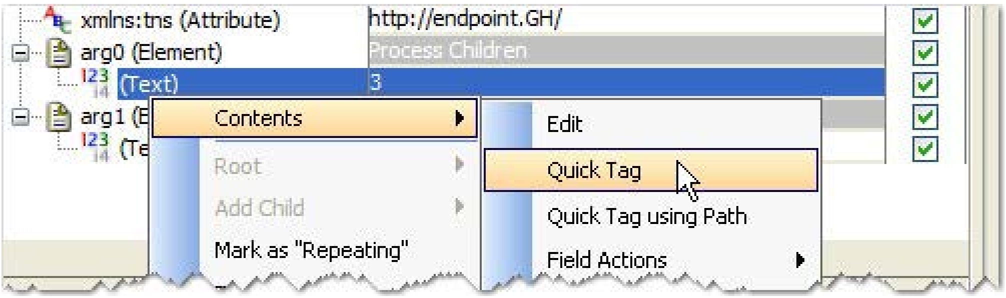
The response is: %%return%% for iteration number %%TEST/ITERATION/NUMBER%%

10. Click **OK** to save the test action and return to the Test Factory.

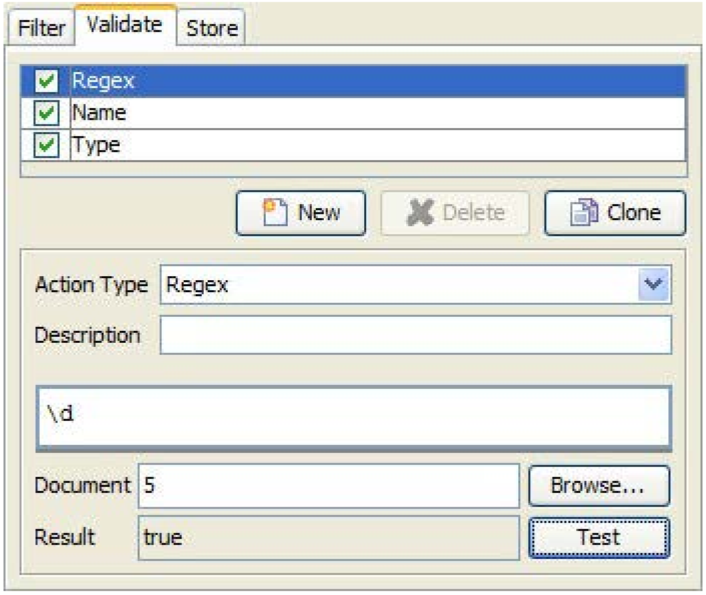
5.4 Configuring Tests

Now that the test data set is pointing to the new data source (our simple text file), we can configure a test to utilize the data that the source contains.

1. In the Test Factory perspective (**F10**), double-click the **additionCheckMEP** test to open it.
2. Double-click the **Send Request** action to open it.
3. Right-click on each of the text nodes of the **arg0** and **arg1** elements and select **Contents > Quick Tag** – click **Yes** when prompted to overwrite data.


xmlns:tns (Attribute) http://endpoint.GH/ ☑
arg0 (Element) Process Children ☑
(Text) 3 ☑
arg1 (E ☑
(Te ☑

4. Click **OK** when finished.
5. Open the **Receive Reply** action.
6. On the return text node, either disable validation or replace the **Equality** action with the **Regex** action and enter "\d" to match the expected result, then click **OK**.



☑ Regex
☑ Name
☑ Type

Action Type: Regex
Description:

```
\d
```

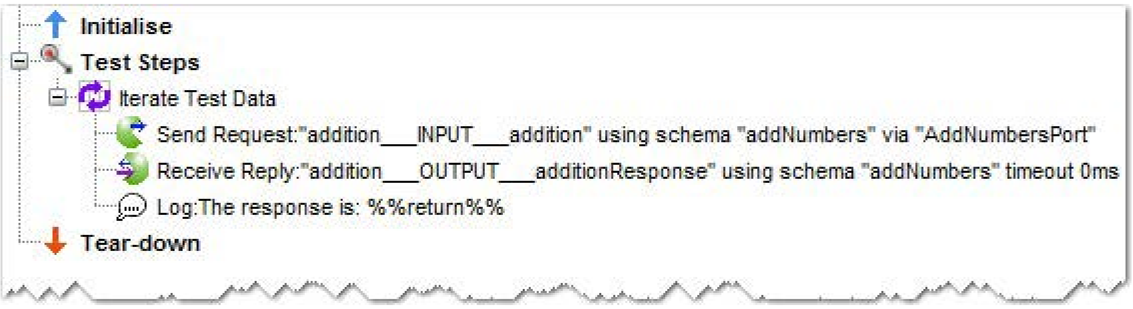
Document: 5
Result: true

7. Right-click the **Test Steps** phase and select **New > Flow > Iterate Test Data**.

   The new action should be created below the existing actions.

8. Select the Send Request, Receive Reply, and Log actions (using the **Ctrl** or **Shift** key) and drag them on to the new Iterate Test Data action.

   The actions should now be inside the Iterate Test Data action, as shown below:

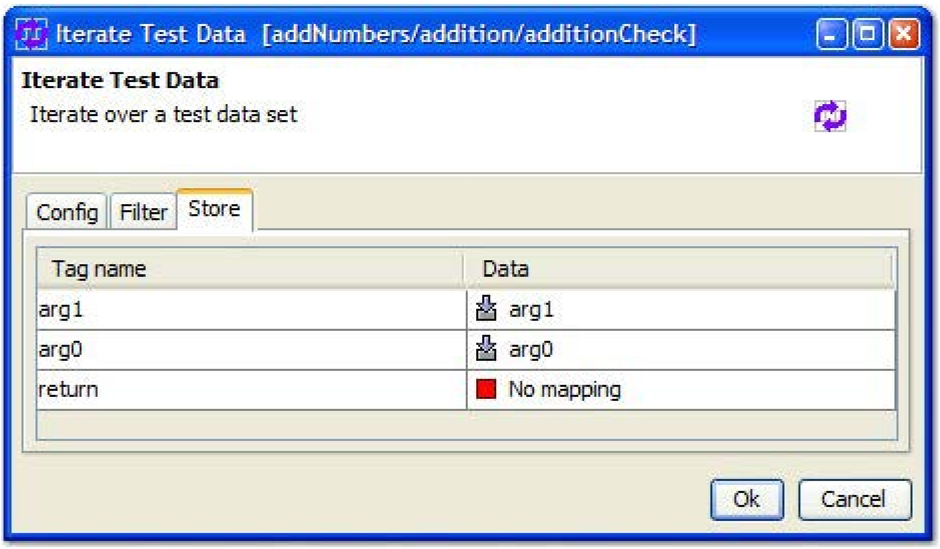


```
Initialise
Test Steps
  Iterate Test Data
    Send Request:"addition___INPUT___addition" using schema "addNumbers" via "AddNumbersPort"
    Receive Reply:"addition___OUTPUT___additionResponse" using schema "addNumbers" timeout 0ms
    Log:The response is: %%return%%
Tear-down
```

9. Double-click the Iterate Test Data action to open it for editing.

10. Under the **Config** tab, click **Browse** next to the **Test data set** field and select the **addNumbers** data set from the project resource tree.

11. Select the **Store** tab to verify the mappings from the data set to the test tags.



Iterate Test Data [addNumbers/addition/additionCheck]

Iterate Test Data

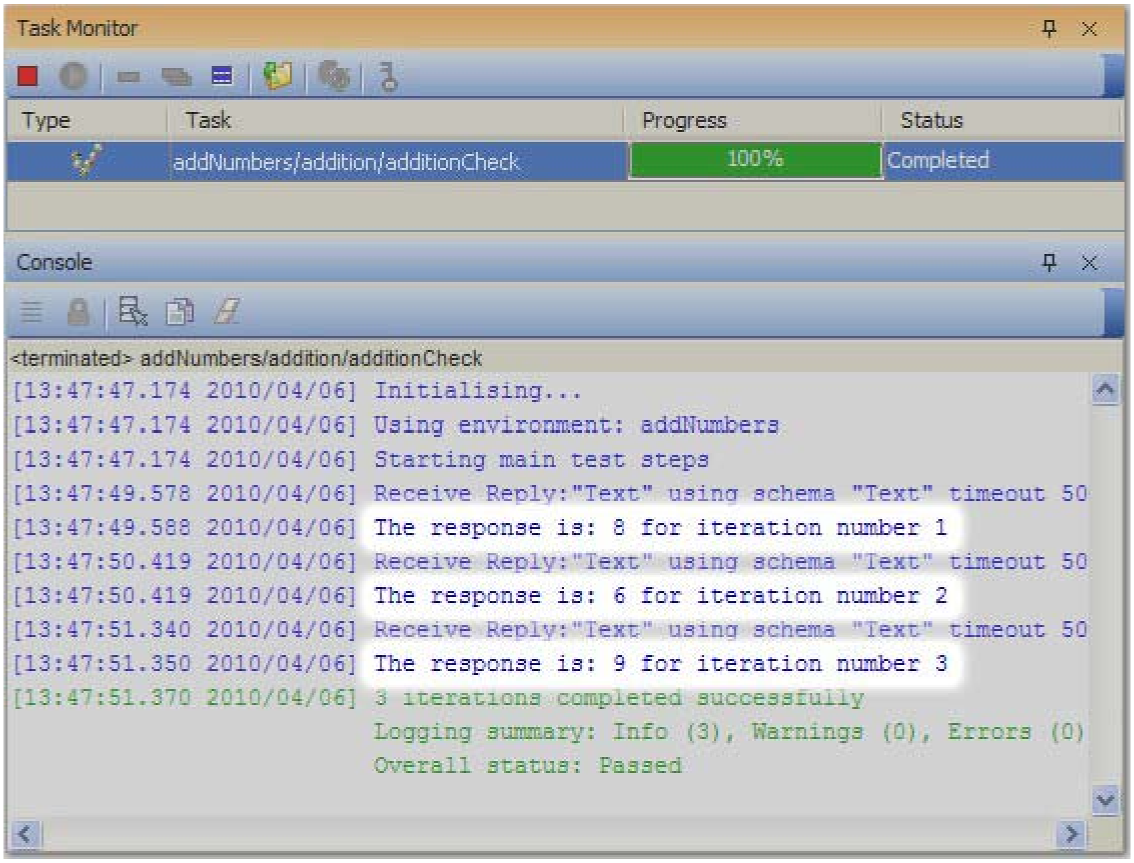
Iterate over a test data set

Tag name	Data
arg1	arg1
arg0	arg0
return	🟥 No mapping

NOTE: The return node is unmapped since there is no such column in the test data set.

12. Run the **additionCheck** test.

The Iterate Test Data action iterates through the data in the data set and substitutes the tags according to the defined mappings. The test should pass and the results should be written to the console by the Log action.



Type	Task	Progress	Status
	addNumbers/addition/additionCheck	100%	Completed

```
<terminated> addNumbers/addition/additionCheck
[13:47:47.174 2010/04/06] Initialising...
[13:47:47.174 2010/04/06] Using environment: addNumbers
[13:47:47.174 2010/04/06] Starting main test steps
[13:47:49.578 2010/04/06] Receive Reply:"Text" using schema "Text" timeout 50
[13:47:49.588 2010/04/06] The response is: 8 for iteration number 1
[13:47:50.419 2010/04/06] Receive Reply:"Text" using schema "Text" timeout 50
[13:47:50.419 2010/04/06] The response is: 6 for iteration number 2
[13:47:51.340 2010/04/06] Receive Reply:"Text" using schema "Text" timeout 50
[13:47:51.350 2010/04/06] The response is: 9 for iteration number 3
[13:47:51.370 2010/04/06] 3 iterations completed successfully
                          Logging summary: Info (3), Warnings (0), Errors (0)
                          Overall status: Passed
```

Functions and Decisions

Contents

This chapter provides an overview of how to use functions and decisions in Rational Integration Tester.

6.1 Overview

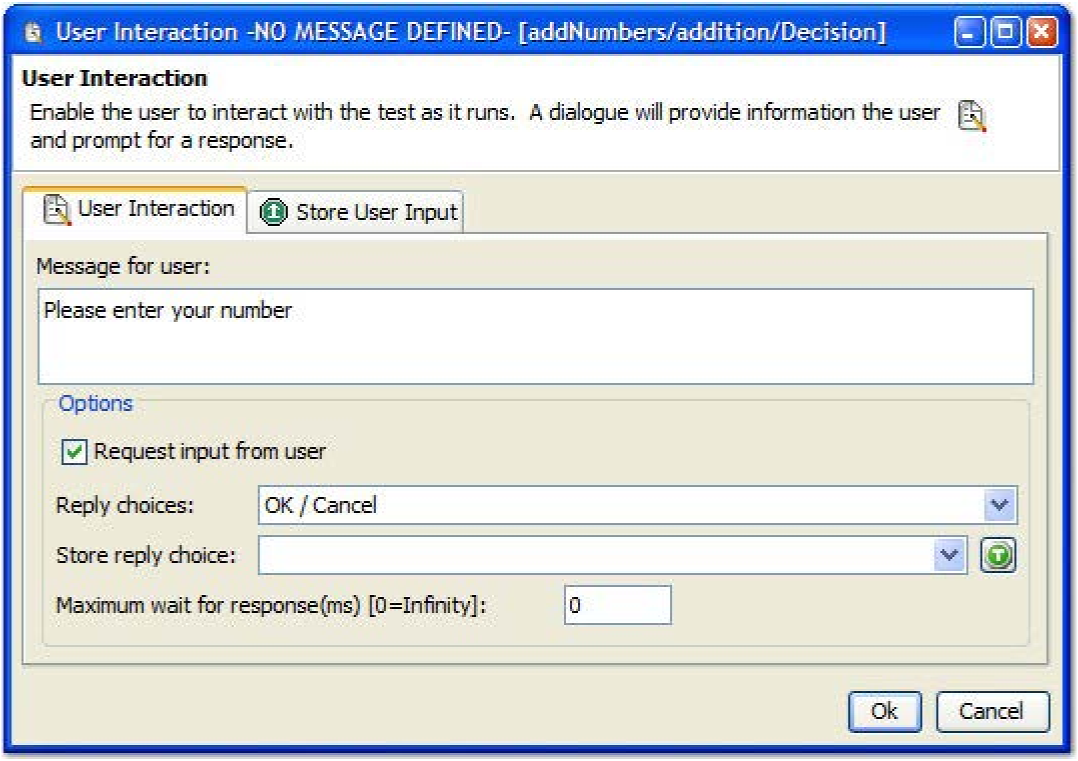
A fundamental requirement for some tests is to apply logic that tests something and takes alternative actions depending on the outcome. One of several built in or custom functions in Rational Integration Tester can be used to apply the logic, and a *Decision* action allows you to create one or more expressions that evaluate the outcome of the function. Two possible execution paths can be taken from the decision.

In this chapter we will create a trivial example that will test the value contained in a tag and generate an output a message to the user if the value is greater than 100. The first part of the test will prompt the user for a value and store it in a tag. This test will use the *User Interaction* action to solicit a response from the user. User interactions can be useful for debugging, as you can use them to display tag values at any point during the execution of a test.

6.2 Exercise

The following exercise creates a new test and uses a function and the *User Interaction* action to evaluate the results of user input.

1. Open the Test Factory perspective (**F10**).
2. Click on the **addition** operation and create a new test called "Decision".
3. Double-click the new test to open it for editing, then right-click the Test Steps phase and add a User Interaction action (**New > General > User Interaction**).
4. Double-click the new action and add the following message in the **Message for user** field: "Please enter your number"
5. Tick the box next to the **Request input from user** option.



User Interaction -NO MESSAGE DEFINED- [addNumbers/addition/Decision]

User Interaction

Enable the user to interact with the test as it runs. A dialogue will provide information the user and prompt for a response.

Message for user:
Please enter your number

Options

☑ Request input from user

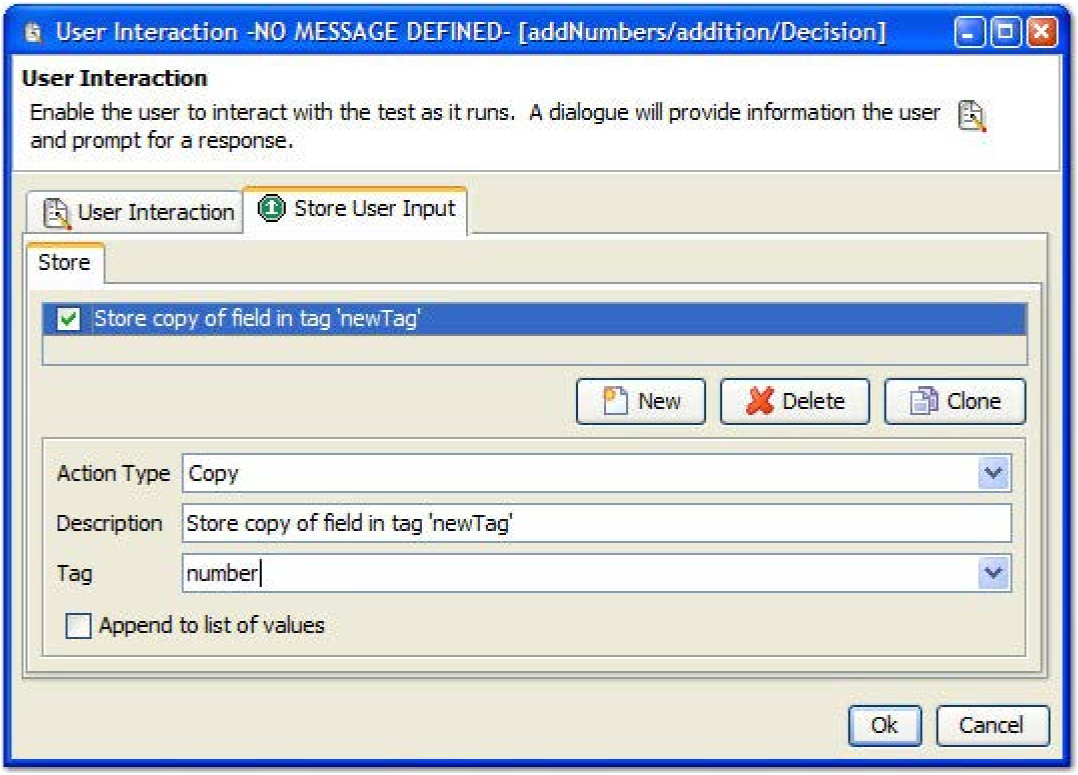
Reply choices: OK / Cancel

Store reply choice:

Maximum wait for response(ms) [0=Infinity]: 0

6. Select the **Store User Input** tab and click **New** to add a store action.

7. Rename the default tag (newTag) to "number".



User Interaction -NO MESSAGE DEFINED- [addNumbers/addition/Decision]

User Interaction

Enable the user to interact with the test as it runs. A dialogue will provide information the user and prompt for a response.

☑ Store copy of field in tag 'newTag'

Action Type: Copy

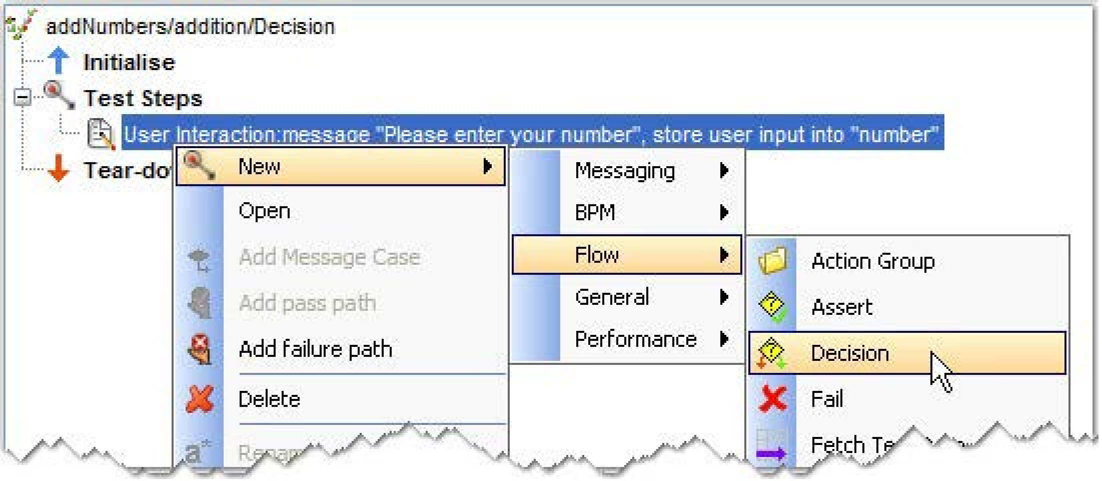
Description: Store copy of field in tag 'newTag'

Tag: number

□ Append to list of values

8. Click **OK** to save the changes and return to the test editor.

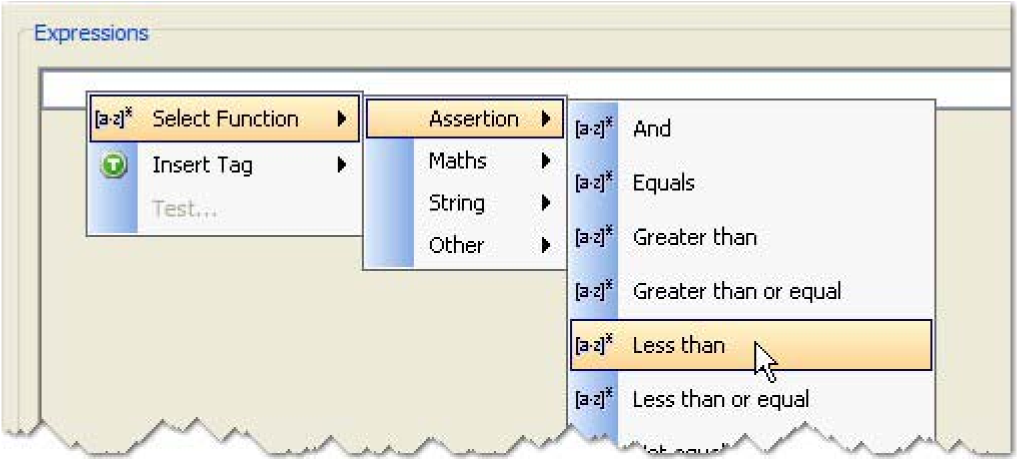
9. Right-click the *User Interaction* step and add a *Decision* step to it (**New > Flow > Decision**).



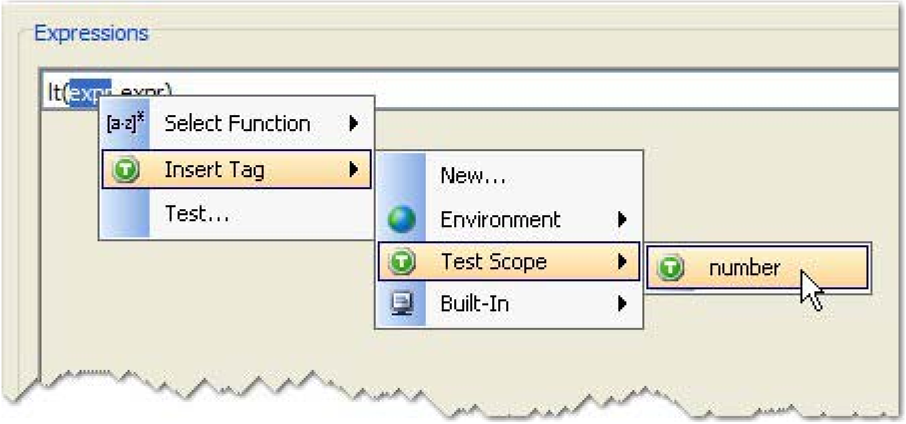
10. Double-click the *Decision* action to edit it.

11. Click **Add** to add an expression.

12. Right-click in the empty expression field and select the **Select Function > Assertion > Less than** option.



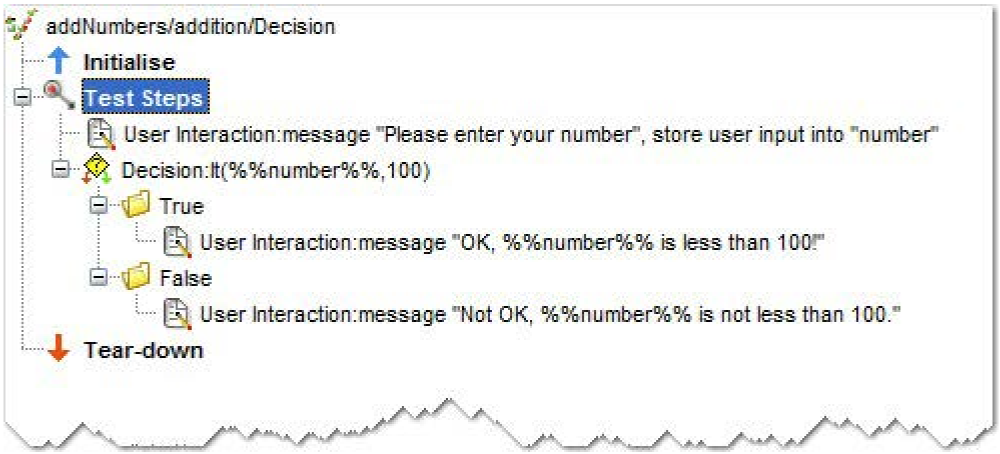
13. Select the first expression and right-click to insert the number tag (**Insert Tag < Test Scope > number**).



14. Replace the second expression with "100" and the expression should read as follows: `lt(%%number%%,100)`

15. Add *User Interaction* actions to the **True** and **False** paths to display the correct message.

The completed test sequence should look as follows.

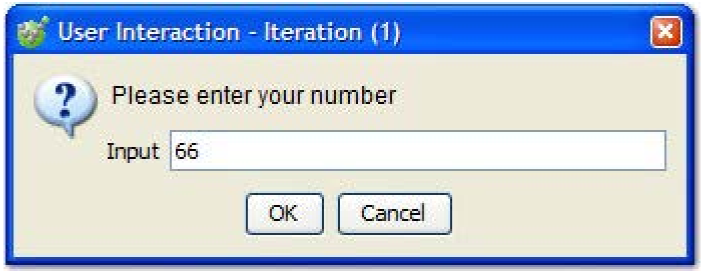


```
addNumbers/addition/Decision
  Initialise
  Test Steps
    User Interaction:message "Please enter your number", store user input into "number"
    Decision:lt(%%number%%,100)
      True
        User Interaction:message "OK, %%number%% is less than 100!"
      False
        User Interaction:message "Not OK, %%number%% is not less than 100."
  Tear-down
```

16. Save the project (click 💾 or press **Ctrl + S**) and run the test.

    You will be prompted for input.

17. Enter a number less than 100 (for example, 66) and click **OK**.

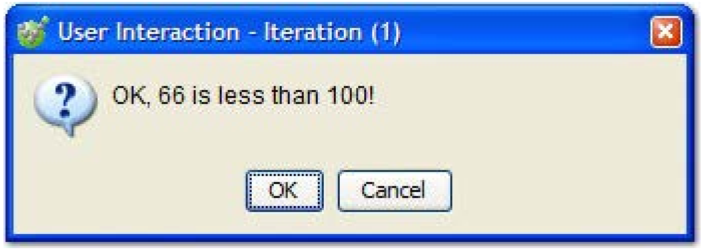


User Interaction - Iteration (1)

Please enter your number

Input: 66

The number will be evaluated and the result is displayed.



User Interaction - Iteration (1)

OK, 66 is less than 100!

Glossary

The following table below lists some of the key terms used in this document, and provides a description of each.

Term	Description
Field	A bit of data constituent to a message. Most fields are scalar and therefore unitary, equivalent to data attributes. Vector fields are an aggregation of fields both scalar and vector, and are usually referred to as Messages. See also Message.
Message	A unit of information made up of a header consisting of meta-information and a body consisting of the message data.
Host	The computer on which a software process runs.
Publish-Subscribe	A messaging paradigm for efficient one-to-many communication in which one process (the publisher) sends information to zero or more other processes (subscribers).
Transport	Informally, the messaging software in use. For instance, TIBCO EMS, TIBCO ActiveEnterprise, IBM WebSphere® MQ (JMS).
Publishing	Making a message (data) available on a message channel.
Subscribing	Receiving a stream of messages (data) on a given message channel.
Server	A host computer on a network shared by more than one user.
JMS	Java Message Service, a J2EE technology. Several implementations of JMS exist, for instance IBM MQ and TIBCO EMS.
JMS Topic	The JMS equivalent of a subject, used by JMS providers to implement Publish-Subscribe messaging paradigms.
JMS Queue	A JMS Queue is normally used for one-to-one messaging.

Term	Description
TIBCO Rendezvous	A software toolkit for creating distributed applications that can inter-operate with TIBCO servers and applications on a TIBCO network. The product includes a communications daemon and APIs that define protocols for publish/subscribe and request/reply data distribution and exchange. It uses the subject-based addressing™ messaging technique for data delivery, and defines rules for supported subject naming formats.
TIBCO AE Message	A TIBCO (Active Enterprise) proprietary format for messages contained within the TIBCO Repository
Subject	A user-defined, meaningful name for identifying messages on transports. For example, the subject EQ.IBM might identify all pricing data about IBM stocks, while EQ.IBM.N might identify price data from the New York Stock Exchange only.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM United Kingdom Limited
Intellectual Property Law
Hursley Park
Winchester
SO21 2JN
Hampshire
United Kingdom

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corporation 2001, 2012.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks and service marks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

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

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

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

IBM®