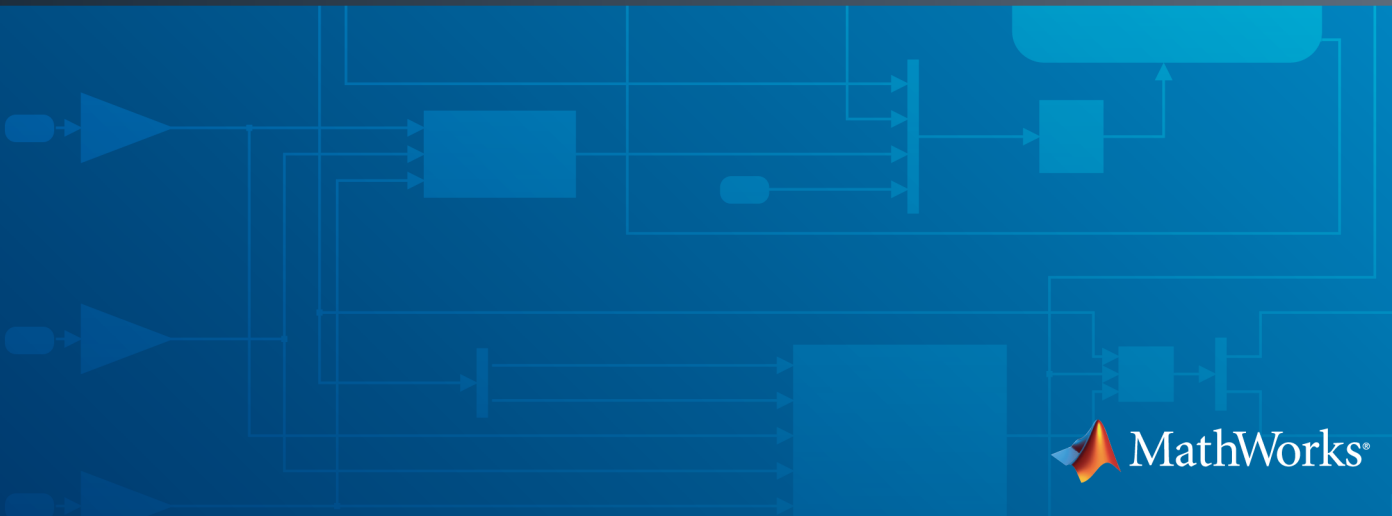
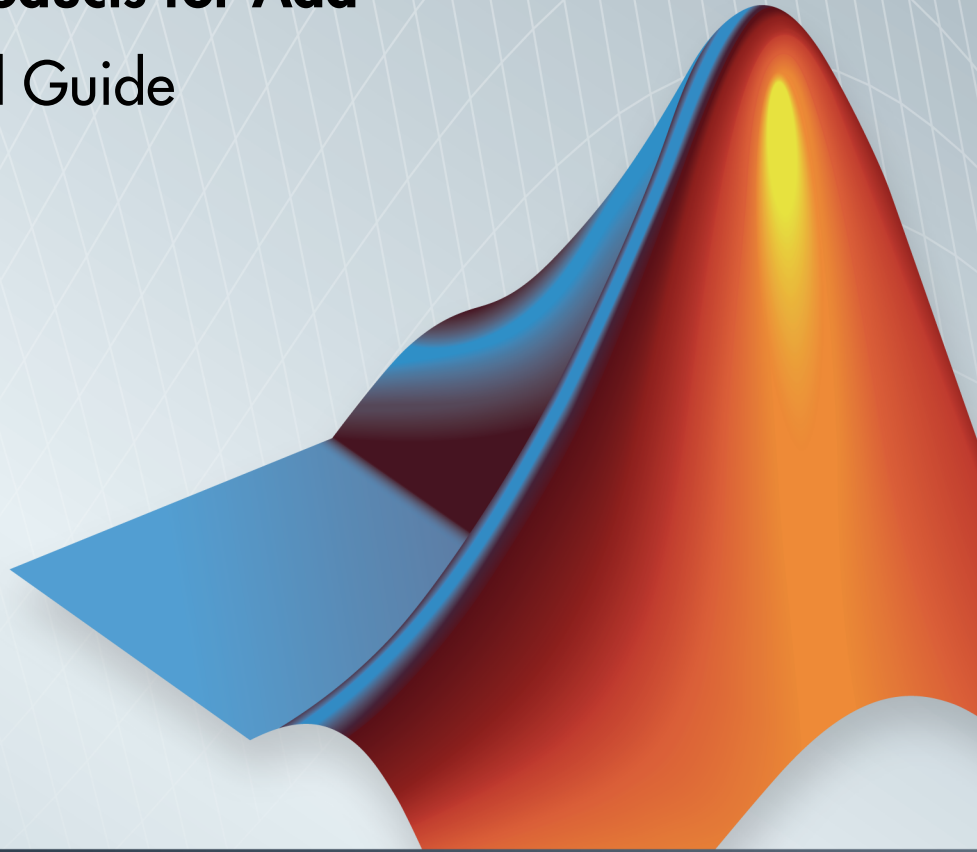


Polyspace[®] Products for Ada

Getting Started Guide

R2014b



How to Contact MathWorks



Latest news: www.mathworks.com
Sales and services: www.mathworks.com/sales_and_services
User community: www.mathworks.com/matlabcentral
Technical support: www.mathworks.com/support/contact_us



Phone: 508-647-7000



The MathWorks, Inc.
3 Apple Hill Drive
Natick, MA 01760-2098

Polyspace[®] Products for Ada Getting Started Guide

© COPYRIGHT 1997–2014 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Patents

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

Revision History

March 2008	First printing	Revised for Version 5.1 (Release 2008a)
October 2008	Second printing	Revised for Version 5.2 (Release 2008b)
March 2009	Third printing	Revised for Version 5.3 (Release 2009a)
September 2009	Online only	Revised for Version 5.4 (Release 2009b)
March 2010	Online only	Revised for Version 5.5 (Release 2010a)
September 2010	Online only	Revised for Version 6.0 (Release 2010b)
April 2011	Fourth printing	Revised for Version 6.1 (Release 2011a)
September 2011	Online only	Revised for Version 6.2 (Release 2011b)
March 2012	Online only	Revised for Version 6.3 (Release 2012a)
September 2012	Online only	Revised for Version 6.4 (Release 2012b)
March 2013	Online only	Revised for Version 6.5 (Release 2013a)
September 2013	Online only	Revised for Version 6.6 (Release 2013b)
March 2014	Online Only	Revised for Version 6.7 (Release 2014a)
October 2014	Online Only	Revised for Version 6.8 (Release 2014b)

Introduction to Polyspace Products for Verifying Ada Code

1

Product Overview	1-2
Polyspace Client for Ada	1-2
Polyspace Server for Ada	1-2
Polyspace Verification	1-4
Overview of Polyspace Verification	1-4
The Value of Polyspace Verification	1-4
Product Components	1-7
Polyspace Verification Environment	1-7
Other Polyspace Components	1-9
Working with Polyspace Software	1-11
Basic Workflow	1-11
The Workflow in This Guide	1-12
Learning More	1-13
Product Help	1-13
Context Sensitive Help	1-13
MathWorks Online	1-13
Related Products	1-14
Polyspace Code Prover	1-14
Polyspace Bug Finder	1-14

Setting Up Polyspace Project

2

Set Up Polyspace Project	2-2
Tutorial Overview	2-2
What Is a Project?	2-2
Prepare Project Folder	2-2
Open Polyspace Verification Environment	2-3
Create Project	2-4

Run Verification

3

Run Verification	3-2
Tutorial Overview	3-2
Before You Start the Tutorial	3-2
Prepare for Verification	3-2
Run Remote Verification	3-3
Run Local Verification	3-4
Next steps	3-6

Review Verification Results

4

Review Verification Results	4-2
Tutorial Overview	4-2
Open Results	4-2
Review Results	4-4
Generate Report	4-5

Code Verification in IBM Rational Rhapsody Environment

5

Verify Code in IBM Rational Rhapsody Environment	5-2
Code Verification Approach	5-2
Adding Polyspace Profile to Model	5-3
Accessing Polyspace Features	5-3
Configuring Verification Options	5-6
Running a Verification	5-7
Monitoring a Verification	5-7
Viewing Polyspace Results	5-7
Locating Faulty Code in Rhapsody Model	5-8
Template Configuration Files	5-8

Introduction to Polyspace Products for Verifying Ada Code

- “Product Overview” on page 1-2
- “Polyspace Verification” on page 1-4
- “Product Components” on page 1-7
- “Working with Polyspace Software” on page 1-11
- “Learning More” on page 1-13
- “Related Products” on page 1-14

Product Overview

In this section...
“Polyspace Client for Ada” on page 1-2
“Polyspace Server for Ada” on page 1-2

Polyspace Client for Ada

Prove the absence of run-time errors in source code

Polyspace Client for Ada provides code verification that proves the absence of overflow, divide-by-zero, out-of-bounds array access, and certain other run-time errors in source code using static code analysis that does not require program execution, code instrumentation, or test cases. Polyspace Client for Ada uses formal methods-based abstract interpretation techniques to verify code. You can use it on handwritten code, generated code, or a combination of the two, before compilation and test.

Key Features

- Verification of individual packages and package sets
- Formal method based abstract interpretation
- Display of run-time errors directly in code
- Eclipse™ IDE integration

Polyspace Server for Ada

Perform code verification on computer clusters and publish metrics

Polyspace Server for Ada provides code verification that proves the absence of overflow, divide-by-zero, out-of-bounds array access, and certain other run-time errors in source code. For faster performance, Polyspace Server for Ada lets you schedule verification tasks to run on a computer cluster. Jobs are submitted to the server using Polyspace Client for Ada. You can integrate jobs into automated build processes and set up e-mail notifications. You can view defects and regressions via a Web browser. You then use the client to download and visualize verification results.

Key Features

- Web-based dashboard providing code metrics and quality status

- Automated job scheduling and e-mail notification
- Multi-server job queue manager
- Verification report generation
- Mixed operating system environment support

Polyspace Verification

In this section...
“Overview of Polyspace Verification” on page 1-4
“The Value of Polyspace Verification” on page 1-4

Overview of Polyspace Verification

Polyspace products verify Ada code by detecting run-time errors before code is compiled and executed.

To verify the source code, you set up verification parameters in a project, run the verification, and review the results. A graphical user interface helps you to efficiently review verification results. The software assigns a color to operations in the source code as follows

- **Green** – Indicates that an operation is proven to not have certain kinds of error.
- **Red** – Indicates that an operation is proven to have at least one error.
- **Gray** – Indicates unreachable code.
- **Orange** – Indicates that the operation can have an error along some execution paths.

The color-coding helps you to quickly identify errors and find the exact location of an error in the source code. After you fix errors, you can easily run the verification again.

The Value of Polyspace Verification

Polyspace verification can help you to:

- “Enhance Software Reliability” on page 1-4
- “Decrease Development Time” on page 1-5
- “Improve Development Process” on page 1-5

Enhance Software Reliability

Polyspace software enhances the reliability of your Ada applications by proving code correctness and identifying run-time errors. Using advanced verification techniques, Polyspace software performs an exhaustive verification of your source code.

Polyspace software can:

- Prove that your code has certain kinds of errors.
- Prove that your code does not have certain kinds of errors.
- Identify unreachable code.
- Identify code that can have an error along some execution paths.

With this information, you can be confident that you know how much of your code is run-time error free, and you can improve the reliability of your code by fixing the errors.

Decrease Development Time

Polyspace software reduces development time by automating the verification process and helping you to efficiently review verification results. You can use it at any point in the development process. However, using it early in coding phases allows you to find errors when they are less costly to fix.

You use Polyspace software to verify Ada source code before compile time. To verify the source code, you set up verification parameters in a project, run the verification, and review the results. This process takes significantly less time than using manual methods or tools that require you to modify code or run test cases.

Color-coding of results helps you to quickly identify errors. You will spend less time debugging because you can see the exact location of an error in the source code. After you fix errors, you can easily run the verification again.

Using Polyspace verification software helps you to use your time effectively. Because you know the parts of your code that do not have errors, you can focus on the code with proven or potential errors.

Reviewing the code that might have errors (orange code) can be time-consuming, but Polyspace software helps you with the review process. You can use filters to focus on certain types of errors or you can allow the software to identify the code that you should review.

Improve Development Process

Polyspace software makes it easy to share verification parameters and results, allowing the development team to work together to improve product reliability. Once verification parameters have been set up, developers can reuse them for other packages in the same application.

Polyspace verification software supports code verification throughout the development process:

- An individual developer can find and fix run-time errors during the initial coding phase.
- Quality assurance can check overall reliability of an application.
- Managers can monitor application reliability by generating reports from the verification results.

Product Components

In this section...
“Polyspace Verification Environment” on page 1-7
“Other Polyspace Components” on page 1-9

Polyspace Verification Environment

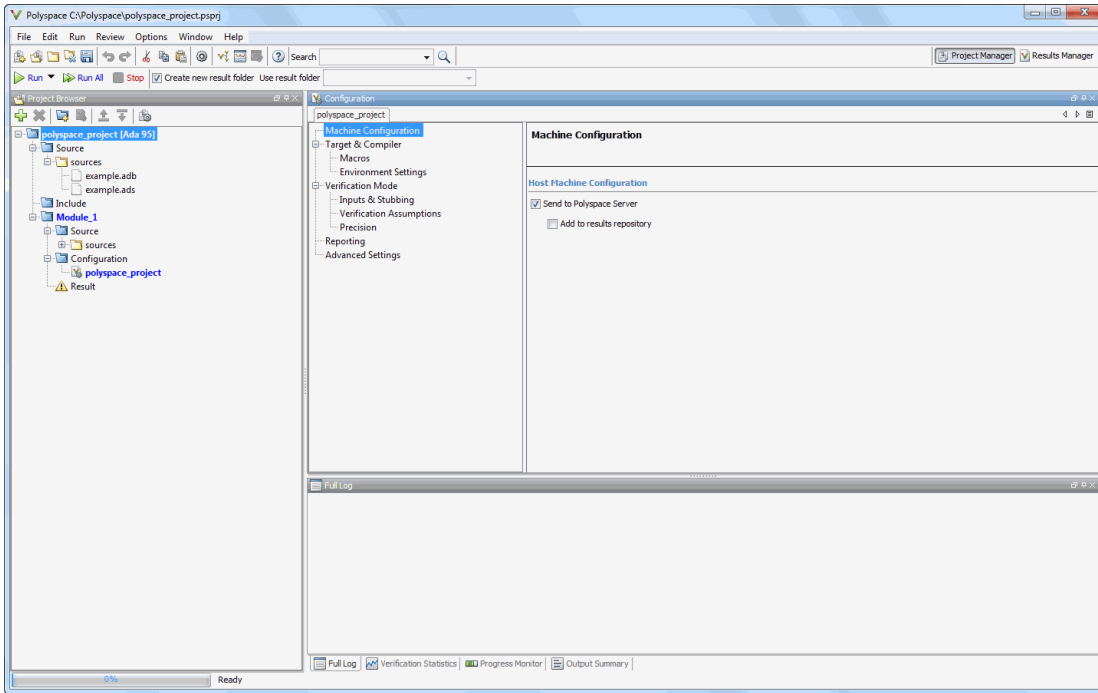
The Polyspace verification environment (PVE) is the graphical user interface of the Polyspace Client for Ada software. You use the Polyspace verification environment to create Polyspace projects, launch verifications, and review verification results.

For Ada verification, you use two perspectives of the Polyspace verification environment:

- “Project Manager Perspective” on page 1-7
- “Results Manager Perspective” on page 1-8

Project Manager Perspective

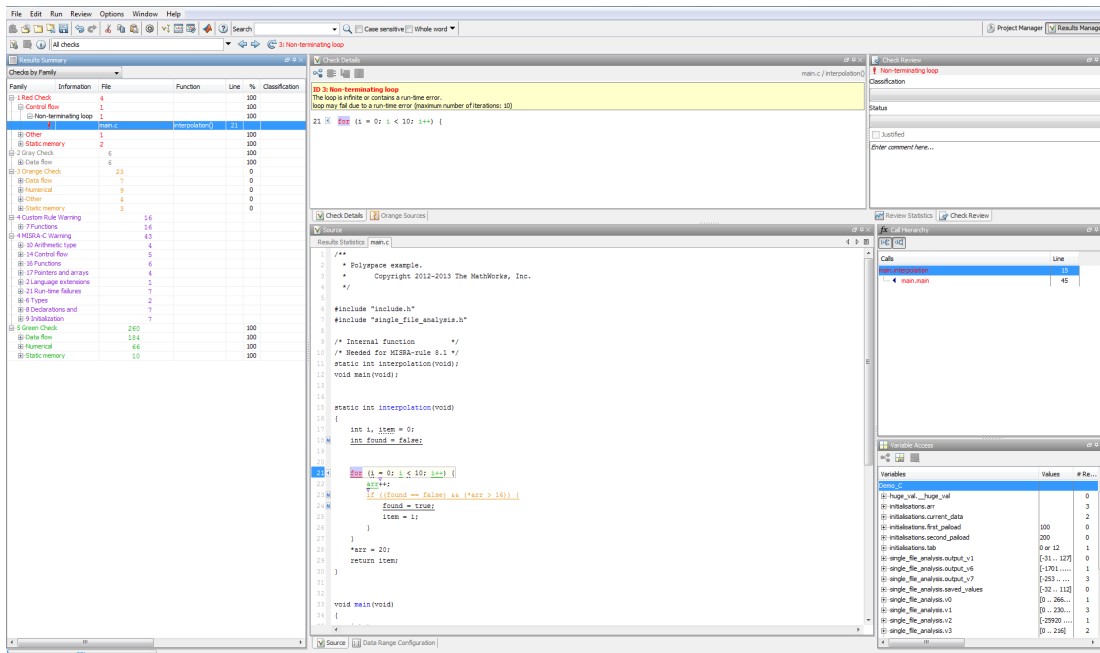
The Project Manager perspective allows you to create projects, set verification parameters, and start verifications.



You use the Project Manager perspective in the tutorial in “Set Up Polyspace Project” on page 2-2.

Results Manager Perspective

The Results Manager perspective allows you to review verification results, comment individual checks, and track review progress.



You use the Results Manager perspective in the tutorial “Review Verification Results” on page 4-2.

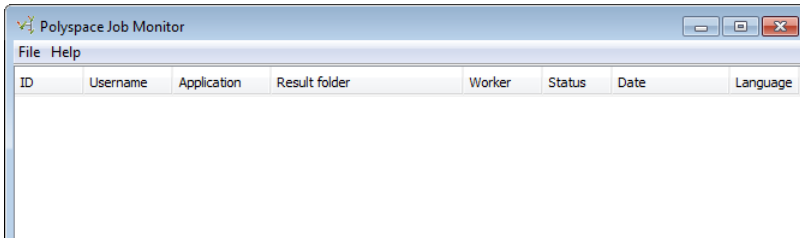
Other Polyspace Components

In addition to the Polyspace verification environment, Polyspace products provide other components to manage verifications, improve productivity, and track software quality. These components include:

- Polyspace Job Monitor
- Polyspace Metrics Web Interface

Polyspace Job Monitor

The Polyspace Job Monitor is the graphical user interface of the Polyspace Server for Ada software. You use the Polyspace Job Monitor to move jobs within the queue, remove jobs, monitor the progress of individual verifications, and download results.



You use the Polyspace Job Monitor in the tutorial “Run Remote Verification” on page 3-3.

Polyspace Metrics Web Interface

Polyspace Metrics is a web-based tool for software development managers, quality assurance engineers, and software developers. Polyspace Metrics allows you to evaluate software quality metrics, and monitor changes in code metrics and run-time checks through the lifecycle of a project.

For information on using Polyspace Metrics, see “Software Quality with Polyspace Metrics”.

Working with Polyspace Software

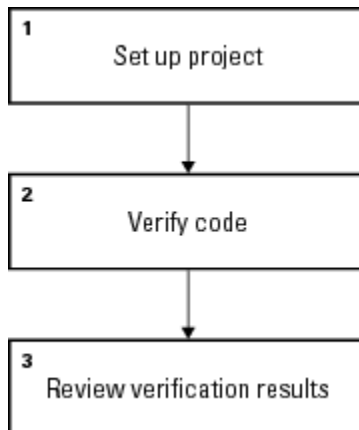
In this section...

“Basic Workflow” on page 1-11

“The Workflow in This Guide” on page 1-12

Basic Workflow

The basic workflow for using Polyspace software to verify Ada source code is:



In this workflow, you:

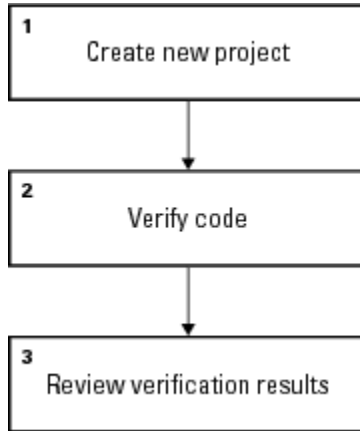
- 1 Use the Project Manager perspective to set up a project file.
- 2 Verify code on a server or client.

You can use the Project Manager perspective to start the verification or you can select files from a Microsoft® Windows® folder and send them to Polyspace software for verification. For verifications that run on a server, you can use the Polyspace Job Monitor to manage the verifications and download the results to a client.

- 3 Use the Results Manager perspective to review verification results.

The Workflow in This Guide

The tutorials in this guide take you through the basic workflow, including the different options for running verifications:



In this workflow, you will:

- 1** Create a new project that you can use for the other steps in the workflow. See the tutorial “Set Up Polyspace Project” on page 2-2.
- 2** Verify a single Ada source code package.

See the tutorial “Run Verification” on page 3-2. In this tutorial, you will use the Project Manager to run a verification of the package on a server and a client.
- 3** Review the verification results. See the tutorial “Review Verification Results” on page 4-2.

Learning More


In this section...

“Product Help” on page 1-13

“Context Sensitive Help” on page 1-13


“MathWorks Online” on page 1-13

Product Help

To access the help that came with your installation, select **Help > Help** or click  in the Polyspace window.

Context Sensitive Help

In addition to the full documentation, you can also access contextual help for verification options and checks.

- In the **Configuration** window, hover your cursor over a verification option. In the tooltip, select **More Help** to get help on the option.
- In the **Check Details** window, select  to get help on the check.

MathWorks Online

For additional information and support, see:

www.mathworks.com/products/polyspace-ada

Related Products

In this section...
“Polyspace Code Prover” on page 1-14
“Polyspace Bug Finder” on page 1-14

Polyspace Code Prover

For information about Polyspace products that verify C/C++ code, see the following:

<http://www.mathworks.com/products/polyspace-code-prover>

Polyspace Bug Finder

For information about Polyspace products that analyze C/C++ code to find possible defects, see the following:

<http://www.mathworks.com/products/polyspace-bug-finder>

Setting Up Polyspace Project

Set Up Polyspace Project

In this section...

“Tutorial Overview” on page 2-2

“What Is a Project?” on page 2-2

“Prepare Project Folder” on page 2-2

“Open Polyspace Verification Environment” on page 2-3

“Create Project” on page 2-4

Tutorial Overview

In this tutorial, you create a new Polyspace project to verify Ada code.

What Is a Project?

A Polyspace project consists of:

- **Source** files.
- **Include** folders.
- One or more modules. You run verification on the source files in each module. Each module has the following folders:
 - **Source** — Contains files used for verification.
 - **Configuration** — Contains analysis options used for verification.
 - **Result** — Contains results of verification.

Prepare Project Folder

In the following procedures, *Polyspace_Install* is the Polyspace installation folder, for example, C:\Program Files\Polyspace\.

- 1 Create a folder `polyspace_project` in a particular location, for example C:\.
- 2 Open `polyspace_project` and create subfolders:
 - `sources`

- `includes`
- 3 Copy `example.adb` and `example.ads` from `Polyspace_Install\polyspace\examples\ada\Demo_Ada_Single-File\sources` to `polyspace_project\sources`.
- 4 Copy all the files from `Polyspace_Install\polyspace\examples\ada\Demo_Ada_Single-File\sources` to `polyspace_project\includes`.

Open Polyspace Verification Environment

- Windows: From the `MATLAB_Install\polyspace\bin` folder, double-click the `polyspace` executable.

You can create a desktop or **Start** menu shortcut to this executable with the icon



if it does not already exist.

- Linux[®] or Mac: Run the following command:

```
/MATLAB_Install/polyspace/bin/polyspace
```

By default, the Polyspace verification environment displays the Project Manager perspective. The Project Manager perspective has three main sections.


Use this section ...	For ...
Project Browser	Specifying: <ul style="list-style-type: none"> • Source files • Include folders • Results folder
Configuration	Specifying analysis options
Output Summary	Monitoring the progress of a verification, and viewing status, log messages, and general verification statistics.

You can resize or hide sections.

Create Project

- “Create New Project” on page 2-4
- “Specify Source Files and Include Folders” on page 2-4
- “Next steps” on page 2-4

Create New Project

- 1 Select **File > New Project**.
- 2 In the Project – Properties dialog box:
 - For **Project name**, enter `example_project`.
 - Clear the **Use default location** check box. To specify where your `polyspace_project` folder is, click .
 - For **Project language**, select **Ada 95**.
 - Clear the boxes under **Project Configuration**. For more information on the option **Use template**, see “What is a Project Template?”.
- 3 Click **Next**.

Specify Source Files and Include Folders

- 1 Select the `sources` folder that you created. Click **Add Source Files**.
- 2 Select the `includes` folder. Click **Add Include Folders**
- 3 Click **Finish**. You can see your project in the **Project Browser**.

Next steps

- 1 “Run Verification”
- 2 “Review Verification Results”

Run Verification

Run Verification

In this section...

- “Tutorial Overview” on page 3-2
- “Before You Start the Tutorial” on page 3-2
- “Prepare for Verification” on page 3-2
- “Run Remote Verification” on page 3-3
- “Run Local Verification” on page 3-4
- “Next steps” on page 3-6

Tutorial Overview

In this tutorial, you run verification on your source code. Perform the steps outlined for remote verification if you want to perform verification on another machine. Otherwise, perform the steps outlined for local verification.

Before You Start the Tutorial

Before you start, you must:

- Complete “Set Up Polyspace Project”. You use the `polyspace_project` folder and the `example_project.psprj` file in this tutorial.
- “Modify Polyspace Server Configuration” for remote verification and “Configure Polyspace Metrics Web Interface” for Polyspace Metrics.

Prepare for Verification

If `example_project.psprj` is not already open in the **Project Browser**, then:


- 1 Select **File > Open Project**.
- 2 In the Open Project dialog box, from the **Look in** drop-down list, navigate to `polyspace_project`.
- 3 Select the project file `example_project`.
- 4 Click **Open**.

Run Remote Verification

- “Start Verification” on page 3-3
- “Monitor Progress” on page 3-3
- “Stop Verification” on page 3-3

Start Verification

Before you start remote verification, you must perform a one-time setup. See “Modify Polyspace Server Configuration”.

- 1 On the **Project Browser** pane, select **Module_1**.
- 2 On the **Configuration** pane, select **Machine Configuration**.
- 3 Select **Send to Polyspace Server**. By default, this action enables the **Add to results repository** option.
- 4 On the Project Manager toolbar, click  .

The following happens:

- a On the local host computer, Polyspace Code Prover™ compiles your code.
- b When the **Compile** phase is complete, you see the message **Verification queued on server** in the **Output Summary** tab.

If the verification fails, go to “Verification Process Failed Errors”.

Monitor Progress

To monitor the progress of a remote verification:

- 1 Select **Tools > Open Job Monitor**.
- 2 In the Polyspace Job Monitor, right-click your verification.
- 3 Select **View Log File**.

Stop Verification

To stop a remote verification:

- 1 Select **Tools > Open Job Monitor**.


- 2 In the Polyspace Job Monitor, right-click your verification.
- 3 Select **Remove From Queue**.

Run Local Verification

- “Start Verification” on page 3-4
- “Monitor Progress” on page 3-4
- “Stop Verification” on page 3-5

Start Verification

To start a verification on your local computer:

- 1 In the Project Manager perspective, in the **Project Browser**, select **Module_1**.
- 2 On the **Configuration** pane, select **Machine Configuration**. Clear **Send to Polyspace Server** if it is selected.
- 3 On the Project Manager toolbar, click . The Run button is a rectangular button with a green play icon on the left and the word 'Run' in blue text on the right.

If the verification fails, go to “Troubleshooting in Polyspace Products for Ada”.

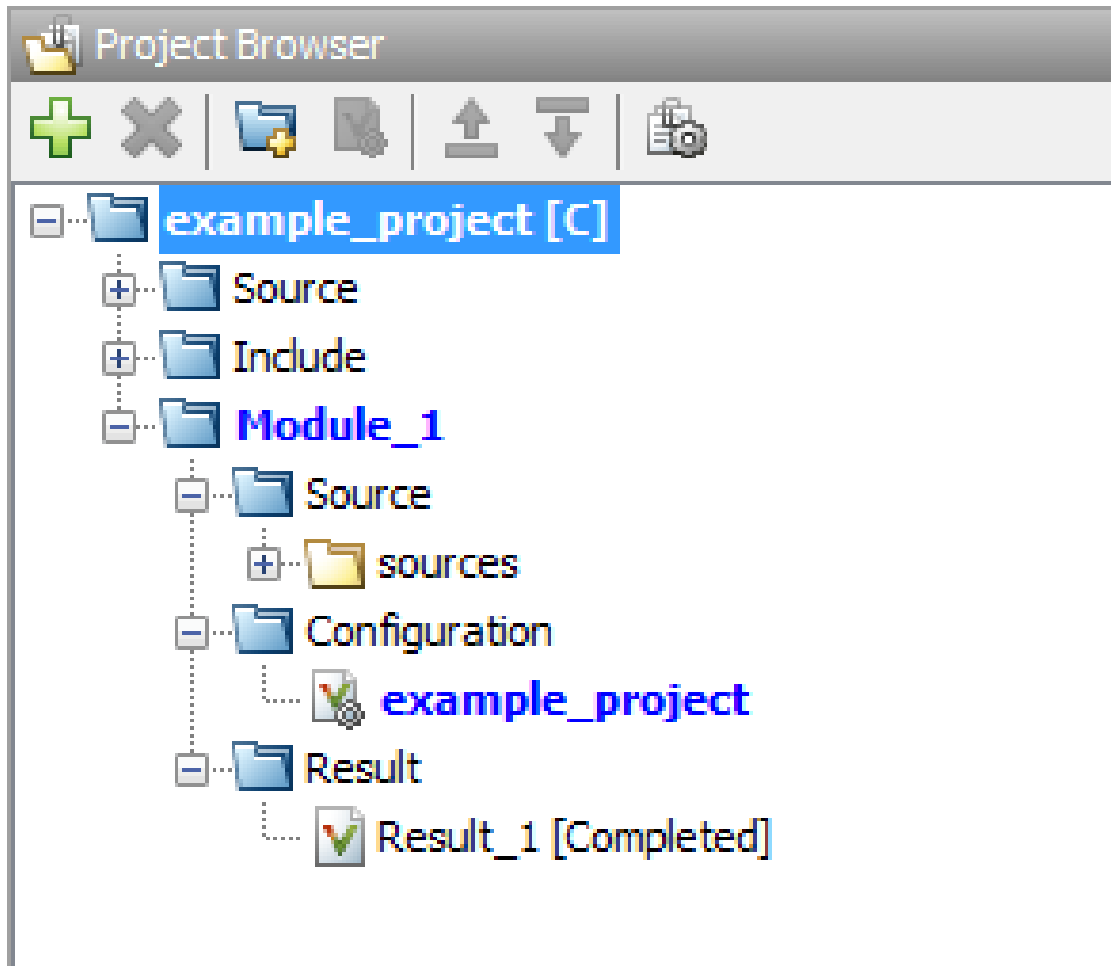
Monitor Progress

To monitor the progress of a local verification, on the **Output Summary** pane, use the following tabs:

- **Output Summary**
- **Full Log**

When the verification is complete, you see:

- The message `Verification process completed` in the **Output Summary**.
- The results file, for example **Result_1**, in the **Project Browser**.




- Statistics, such as **Code covered by verification** and **Check Distribution** in the **Dashboard** tab.

Stop Verification

To stop a local verification:

1

On the Project Manager toolbar, click .

A warning dialog box opens.

- 2 Click **Yes**.

The verification stops. If you restart the verification, it starts from the beginning.

Next steps

“Review Verification Results”

Review Verification Results

Review Verification Results

In this section...
“Tutorial Overview” on page 4-2
“Open Results” on page 4-2
“Review Results” on page 4-4
“Generate Report” on page 4-5

Tutorial Overview

In this tutorial, you explore the results of verifying `example.c`. Before starting this tutorial, complete “Run Verification”.

Open Results

- “Remote Verification” on page 4-2
- “Local Verification” on page 4-2

Remote Verification

To open results from a remote verification:

- 1 Select **Metrics > Open Metrics**.

Alternatively, you can enter the remote address directly in a web browser. For more information, see “View Polyspace Metrics Project Index”.

- 2 Click the **Project** cell of your verification.

You can see a summary of your project.

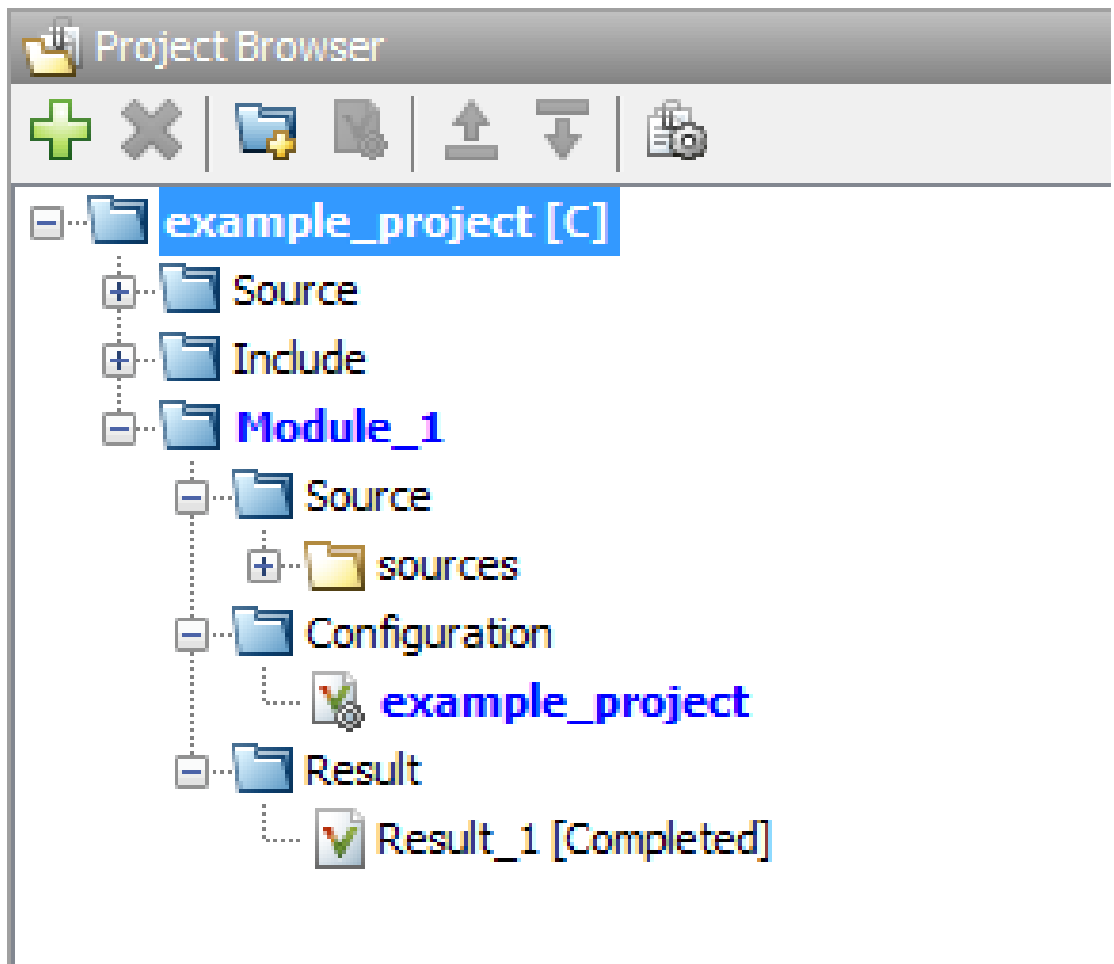
- 3 On the **Summary** tab, click the **1.0** cell in the **Verification** column.

Your results are downloaded into the user interface.

Local Verification

Do one of the following:

- If your project is open in the **Project Browser**, double-click the results file **Result_1**.



The software opens the results in the Results Manager perspective.

- If your project is not open in the **Project Browser**:
 - 1 Select **File > Open Result**.
 - 2 In the Open Results dialog box, navigate to the results folder:
`polyspace_project\Module_1\Result_1\example_project`

- 3 Select `example_project.rte`.
- 4 Click **Open**.

Review Results

Polyspace performs checks on each operation in your code. The software reports whether a check is green, red, orange or gray.

Check color	Indicates
Red	The code operation fails the check on every execution path.
Green	The code operation passes the check on every execution path.
Orange	The code operation fails the check on some execution paths.
Gray	The code operation is unreachable from entry-point functions.

- 1 On the **Results Summary** pane, select **Group by > File**.

The checks are grouped by file. Within each file, the checks are grouped by function.


- 2 Expand the following function names and select a check in the function. The corresponding line of code on the **Source** pane appears highlighted.

Function	Check	Source Code Appearance	Reason
UNREACHABLE_CODE	Gray Unreachable code	The <code>:</code> on line 182 is gray.	<code>x</code> is greater than 0. So the <code>if</code> statement branch cannot be reached.
INFINITE_LOOP	Red Non-terminating loop	The keyword <code>loop</code> on line 123 is red.	<code>x</code> is never less than 0. So the code cannot exit the loop.
RECURSION	Orange Division by Zero	The <code>/</code> sign on line 62 is orange.	The denominator can be zero.

Function	Check	Source Code Appearance	Reason
NON_INFINITE_LOO	Green Overflow	The + sign on line 112 is green.	cur does not overflow. The loop on line 110 terminates before cur can overflow.

- 3** To find further information about a check, do one of the following:
- View the message on the **Check Details** pane.
 - Place your cursor on the check in the **Source** pane. View the tooltip.

- 4** Filter **Division by Zero** checks. To do this:

- a** Click  on the **Check** column header.
- b** From the drop-down list, clear **All** and select **Division by Zero**.

The **Results Summary** pane displays only the **Division by Zero** checks.

- 5** On the **Results Summary** pane, select the red **Division by Zero** check in the function PROCEDURE_ZDV. Enter the following review information.

Column	Action
Classification	High
Status	Fix
Comment	x is always zero.

Generate Report

To generate a verification report:

- 1** If your verification results are not already open, open them.
- 2** Select **Reporting > Run Report**.
- 3** In the **Select Reports** section, select **Developer**.
- 4** For **Output folder**, select C:\polyspace_project\Module_1\Result_1\Polyspace-Doc.
- 5** For **Output format**, select PDF .

6 Click **Run Report**.

The software creates the specified report and opens it.

Code Verification in IBM Rational Rhapsody Environment

Verify Code in IBM Rational Rhapsody Environment

In this section...

- “Code Verification Approach” on page 5-2
- “Adding Polyspace Profile to Model” on page 5-3
- “Accessing Polyspace Features” on page 5-3
- “Configuring Verification Options” on page 5-6
- “Running a Verification” on page 5-7
- “Monitoring a Verification” on page 5-7
- “Viewing Polyspace Results” on page 5-7
- “Locating Faulty Code in Rhapsody Model” on page 5-8
- “Template Configuration Files” on page 5-8

Code Verification Approach

In a collaborative Model-Driven Development (MDD) environment, software run-time errors can be produced by either design issues in the model or faulty handwritten code. You may be able to detect the flaws using code reviews and intensive testing. However, these techniques are time-consuming and expensive.

With Polyspace Products for Ada, you can verify Ada code that you generate from your IBM® Rational® Rhapsody® model. As a result, you can detect run-time errors and automatically identify model flaws quickly and early during the design process.

For information about installing and using IBM Rational Rhapsody, go to www-01.ibm.com/software/awdtools/rhapsody/.

The approach for using Polyspace Products for Ada within the IBM Rational Rhapsody MDD environment is:

- Integrate the Polyspace add-in with your Rhapsody project. See “Adding Polyspace Profile to Model” on page 5-3.
- If required, specify Polyspace configuration options in the Polyspace verification environment. See “Configuring Verification Options” on page 5-6.
- Specify the `include` path to your operating system (environment) header files and run verification. See “Running a Verification” on page 5-7 and “Monitoring a Verification” on page 5-7.

- View results, analyze errors, and locate faulty code within model. See “Viewing Polyspace Results” on page 5-7 and “Locating Faulty Code in Rhapsody Model” on page 5-8.

Adding Polyspace Profile to Model

Before you try to access Polyspace features, you must add the Polyspace profile to your model :

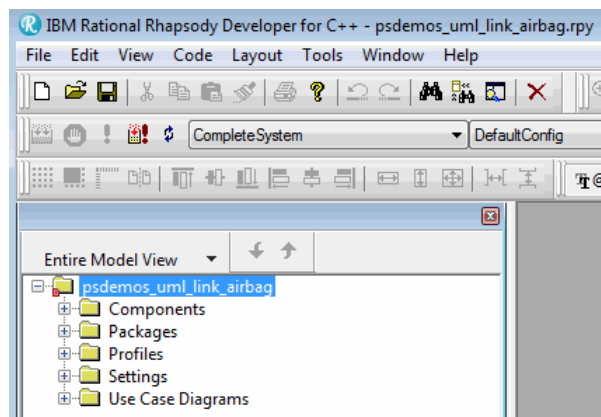
- 1 In the Rhapsody editor, select **File > Add Profile to Model**. The Add Profile to Model dialog box opens.
- 2 Navigate to the folder *Polyspace_Install\polyspace\plugin\rhapsody\profiles\Polyspace*.
- 3 Select the file *Polyspace.sbs*. Then click **Open**.

Now, if you right-click a package or file, you see the **Polyspace** item in the context menu. Selecting **Polyspace** opens the Polyspace Verification dialog box.

Accessing Polyspace Features

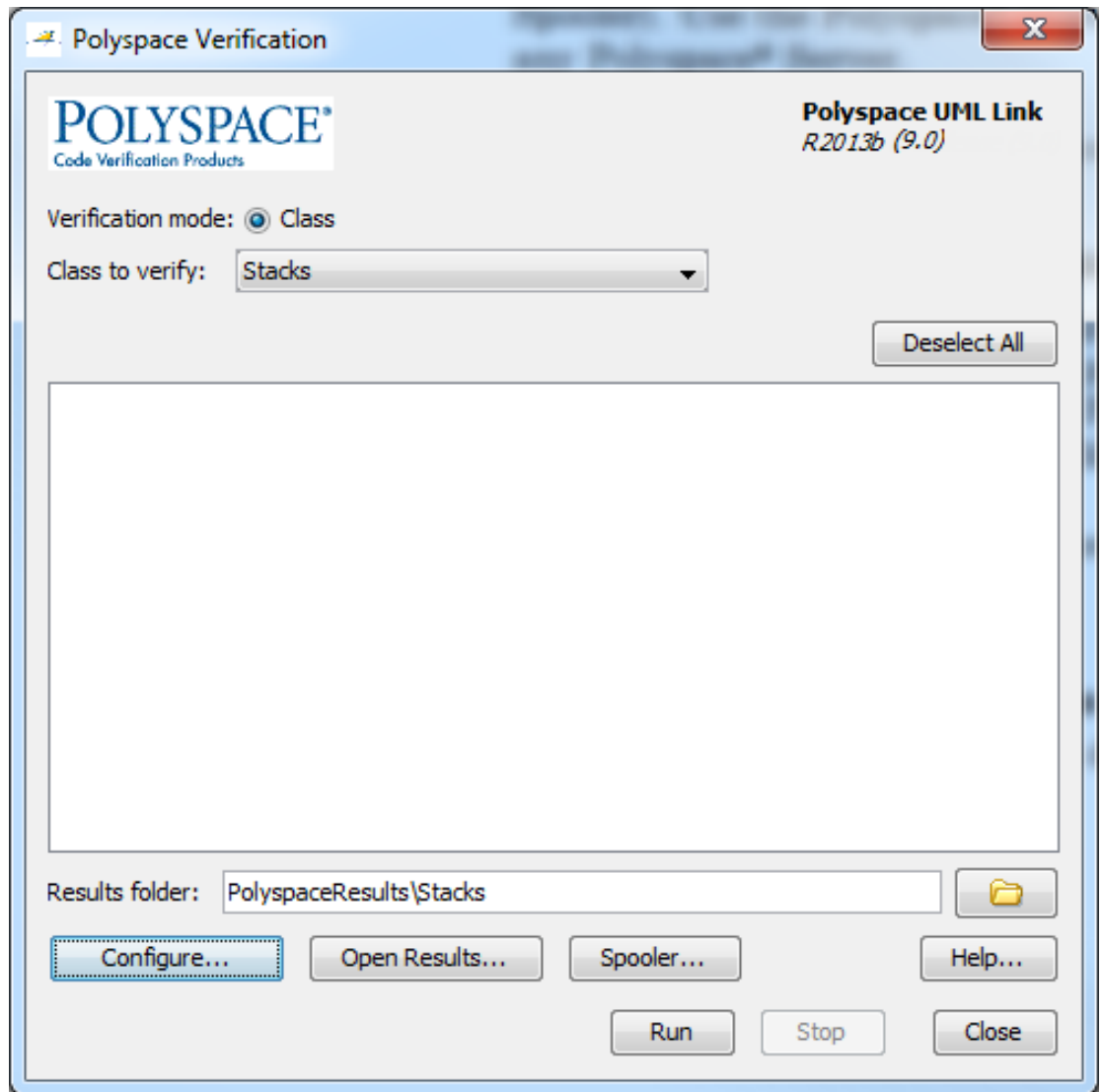
To access Polyspace features in the Rhapsody editor:

- 1 Open the model that you want to verify. For example, *psdemos_um1_link_airbag.rpy* in *Polyspace_Install/polyspace/plugin/rhapsody/psdemos.Polyspace_Install* is the location of the Polyspace installation folder.



- 2 In the **Entire Model View**, expand the **Packages** node.
- 3 Right-click a package, for example, **AirBagFiles**.
- 4 From the context menu, select **Polyspace**.

The Polyspace Verification dialog box opens.



Through the Polyspace Verification dialog box, you can:

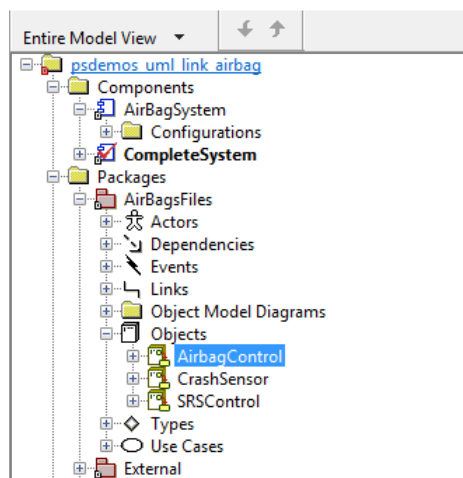
- Specify verification options. See “Configuring Verification Options” on page 5-6.


- Start verifications. See “Running a Verification” on page 5-7.
- Stop client-based verification. See “Running a Verification” on page 5-7.
- View verification results. See “Viewing Polyspace Results” on page 5-7.
- Open help.
- Open the Polyspace Job Monitor. See “Monitoring a Verification” on page 5-7.

Configuring Verification Options

To specify options for your verification:

- 1 In the **Entire Model View**, right-click a package or class, for example, **AirbagControl**.



- 2 From the context menu, select **Polyspace**.
- 3 In the Polyspace Verification dialog box, click **Configure**. The **Configuration** pane of the Polyspace verification environment opens.
- 4 Select options for your verification.
- 5 To save your options, on the toolbar, click .

For information on how to choose your options, see “Analysis Options”.

Running a Verification

Before starting a verification, make sure that the generated code for the model is up to date.

To start a verification:

- 1 In the Rhapsody editor, select **Tools > Polyspace**. The Polyspace Verification dialog box opens.
- 2 In the **Results folder** field, specify a location for your verification results.
- 3 By default, the **Verification mode** is **Class**.
- 4 Click **Run**. In the **Log** view of the Rhapsody editor, you see verification messages.

To stop a client verification, in the Polyspace Verification dialog box, click **Stop**.

To stop a verification on the Polyspace Server, use the Polyspace Job Monitor. See “Monitoring a Verification” on page 5-7.

Monitoring a Verification

If your verification is client-based, you can observe progress in the **Log** view of the Rhapsody editor.

If your verification is running on a Polyspace Server:

- 1 In the Rhapsody editor, select **Tools > Polyspace**.
- 2 In the Polyspace Verification dialog box, click **Job Monitor**.

Use the Polyspace Job Monitor to manage jobs running on a Polyspace Server.

For more information, see “Verification Management”.

Viewing Polyspace Results

To view results from the last completed verification:

- 1 In the Rhapsody editor, select **Tools > Polyspace**.
- 2 In the Polyspace Verification dialog box, click **Open Results**.

The Polyspace verification environment opens, displaying results in the Results Manager perspective.

For more information on Polyspace verification results, see “Run-Time Error Review”.

Locating Faulty Code in Rhapsody Model

To identify the faulty code within your Rhapsody model using Polyspace verification results:

- 1 In the Results Manager perspective of the Polyspace verification environment, navigate to an error.
- 2 In the Source pane, right-click the error. From the context menu, select **Back To Model**.

Tip For the **Back To Model** command to work, you must have your Rhapsody model open.

The **Back To Model** command works best when the Polyspace check is enclosed by the tags `//#[` and `]#//`.

The software locates the faulty code within your Rhapsody model. Depending on the Rhapsody configuration, the faulty code appears either in a dialog box or in the code view.

The 64-bit version of the Polyspace product supports the **Back To Model** command only for version 8.0 of the IBM Rational Rhapsody product. For other versions, use the 32-bit Polyspace version.

To install the 32-bit Polyspace version, from a DOS command window, run the following command:

```
DVD\Installer32bits\Windows\Disk1\InstData\VM\Polyspace.exe
```

Template Configuration Files

- “Using Template Configuration Files” on page 5-9
- “Default Configuration Options” on page 5-9

Using Template Configuration Files

The first time you perform a verification, the software copies a template, Polyspace configuration file, from *Polyspace_Install/polyspace/plugin/rhapsody/etc/template_language.psprj* to the project folder. The software also renames the copy *model_language.psprj*, where:

- *model* is the name of your model.
- *language* is the name of the language that the model targets, that is ADA.

You can update the template *.psprj* file by one of the following means:

- Editing it through the Polyspace verification environment
- Double-clicking the file in a Windows Explorer window
- Replacing the template file with a copy of the *.psprj* file from a Rhapsody model folder

You can then share a configuration among project members and use the configuration with other projects.

Default Configuration Options

The *template_language.psprj* XML files specify the default option values for code verification.

The file *template_Ada.psprj* is:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright 1999-2012 The MathWorks, Inc. -->
<!-- DO NOT EDIT THIS FILE: Some changes can lead to a
      crash of Polyspace products -->
<polyspace_project name="template_psprj" language="Ada 95" author="polyspace"
version="1.0" date="08/04/2011" path="file:/C:/Program Files/Polyspace
/polyspace/plugin/rhapsody/etc/template_ADA.psprj" rev="1.3">
  <source>
  </source>
  <include>
  </include>
  <module name="Verification_1" isactive="true">
    <source>
    </source>
    <optionset name="template_psprj" isactive="true">
      <option flagname="-OS-target">no-predefined-OS</option>
    </optionset>
  </module>
</polyspace_project>
```

