

# Polyspace UML Link RH 5.6 for IBM Rational Rhapsody

## Trace Polyspace results to IBM Rational Rhapsody models

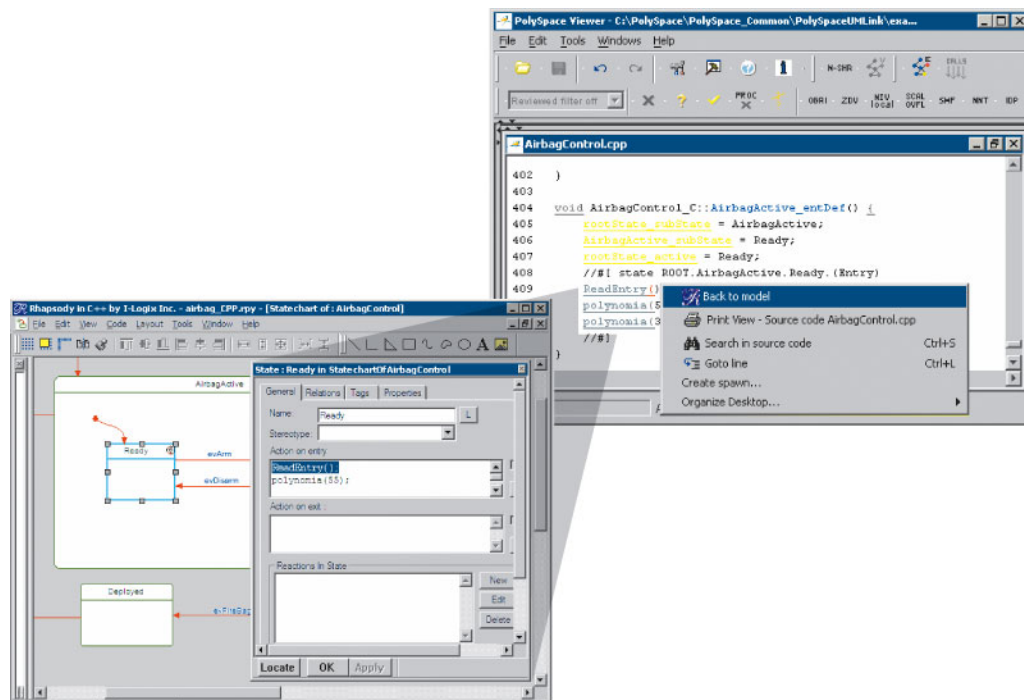
### Introduction

Polyspace UML Link™ RH extends Polyspace Client™ for C/C++ and Polyspace Server™ for C/C++ with tools that let you trace Polyspace® results from generated C++ code directly to your IBM® Rational® Rhapsody® models. As a result, you can identify which parts of the model are reliable, and correct design problems that will cause run-time errors in the code. You can verify a mix of generated and hand-written code before it is compiled.

Polyspace UML Link RH launches from within Rhapsody when you select a class or a package. You can also start the code verification directly from your model using context-sensitive predefined settings.

### Key Features

- Uses advanced code-based verification techniques to automatically verify all program executions
- Incorporates the entire range of parameter values specified in the model
- Traces run-time errors back to the model
- Launches from within the IBM Rational Rhapsody environment



Polyspace Client for C/C++ GUI showing a Rhapsody statechart of an airbag control system (left) and the Polyspace results of the C++ code generated from the Rhapsody model (right). Polyspace UML Link RH lets you right-click a colored diagnostic in the code to access the corresponding component in the Rhapsody model and correct model errors without modifying the C++ code.

### Working with Polyspace UML Link RH

Polyspace UML Link RH integrates with Polyspace Client for C/C++ and Polyspace Server for C/C++ (both available separately) to support the model development workflow. You can:

- Fix implementation errors caused by latent design deficiencies
- Independently verify generated code
- Determine how robust the code is on a particular target
- Correct errors during implementation, before deployment and testing

You can quickly navigate from the code to the relevant section of your Rhapsody model, facilitating design edits and debugging.

## Typical Run-Time Errors Detected

### In Models

- Overflows and underflows
- Division by zero and other arithmetic errors
- Out-of-bounds array access
- Dead code

### In Code

- Illegally dereferenced pointers
- Read-only access to noninitialized data
- Dangerous type conversions
- Access to null `this` pointer (C++)
- Dynamic errors related to object programming and inheritance (C++)
- Errors related to exception handling (C++)
- Noninitialized class members (C++)

## Resources

### Product Details, Demos, and System Requirements

[www.mathworks.com/products/polyspaceumlrh](http://www.mathworks.com/products/polyspaceumlrh)

### Trial Software

[www.mathworks.com/trialrequest](http://www.mathworks.com/trialrequest)

### Sales

[www.mathworks.com/contactsales](http://www.mathworks.com/contactsales)

### Technical Support

[www.mathworks.com/support](http://www.mathworks.com/support)

### Online User Community

[www.mathworks.com/matlabcentral](http://www.mathworks.com/matlabcentral)

### Training Services

[www.mathworks.com/training](http://www.mathworks.com/training)

### Third-Party Products and Services

[www.mathworks.com/connections](http://www.mathworks.com/connections)

### Worldwide Contacts

[www.mathworks.com/contact](http://www.mathworks.com/contact)