

Polyspace Client for Ada 6.0

Prove the absence of run-time errors in source code

Introduction

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 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 methods-based abstract interpretation
- Display of run-time errors directly in code
- Eclipse™ IDE integration

Green:
reliable

Red:
faulty

Orange:
unproven

Gray:
dead

```
Beta : Long_Float;  
procedure Square_Root is  
  Alpha : Float := Random.random;  
  Gamma : long_float;  
begin  
  Square_Root_conv (Alpha, Beta);  
  Beta := Beta - 0.75;  
  Gamma := sqrt(Beta);  
end Square_Root;  
  
procedure Unreachable_Code is  
  x : integer := Random.random;  
  y : integer := Random.random;  
  Z : Integer;  
begin  
  if (x > y) then  
    x := x - y;  
    if (x < 0) then  
      Z := x / Y;  
    end if;  
  end if;  
end Unreachable_Code;
```

Polyspace Viewer, showing color-coding for each file, procedure, and line of code.

Working with Polyspace Client for Ada

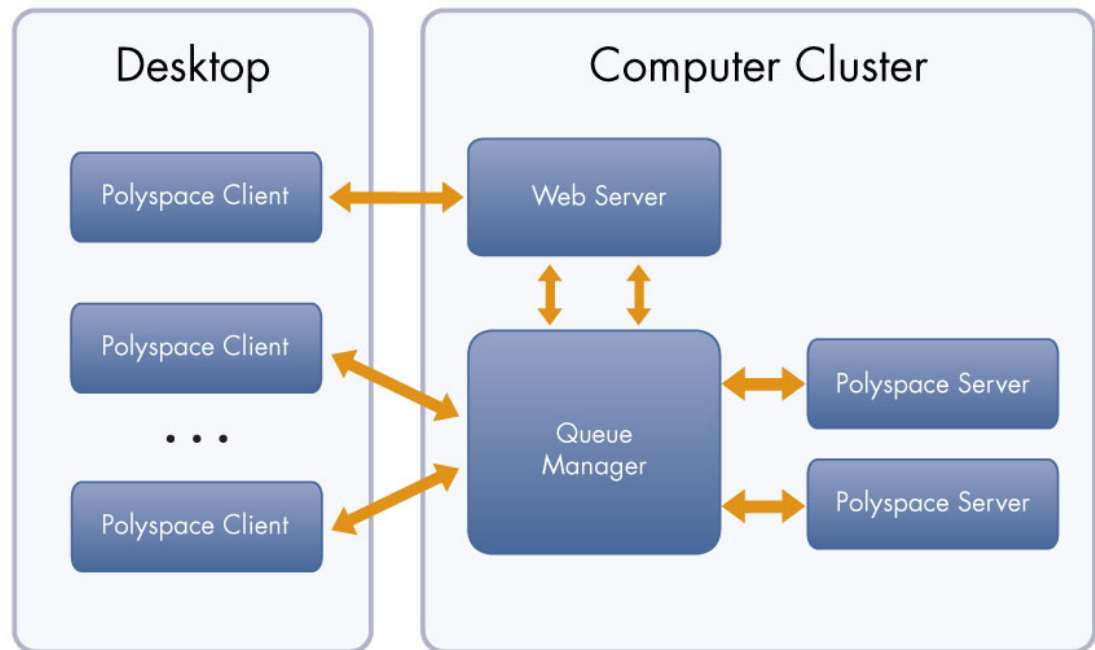
Polyspace Client for Ada provides management and visualization capabilities for verifying software. You can verify individual packages and package sets as soon as the source code is written or updated. When used with Polyspace Server™ for Ada, Polyspace Client for Ada lets you submit verification jobs to computer clusters.

Using the Polyspace Client for Ada command line or graphical user interfaces with the Eclipse IDE, you can:

- Import the source code (Ada 83 or Ada 95)
- Customize a project by target, cross-compiler, or other options
- Submit multiple verification jobs to Polyspace® servers running on a computer cluster or computer farm
- Download data from the server to report or visualize verification results
- View a summary of your software quality metrics on a Web browser dashboard
- Focus on differences in run-time behavior compared with previous verification results

You can use Polyspace Client for Ada to support all critical activities in a software development workflow, including:

- Detecting specific run-time errors in source code
- Proving the absence of certain run-time errors
- Monitoring software quality metrics
- Creating artifacts for certification



Code verification workflow with Polyspace Client for Ada and Polyspace Server for Ada. The queue manager receives the Polyspace verification request and selects the first available server to run the job.

Detect Specific Run-Time Errors

Polyspace Client for Ada uses color-coding to indicate the status of each element in the code, as follows:

Green: proven free of run-time errors

Red: proven faulty each time the operation is executed

Gray: proven unreachable (may indicate a functional issue)

Orange: unproven

Errors detected include:

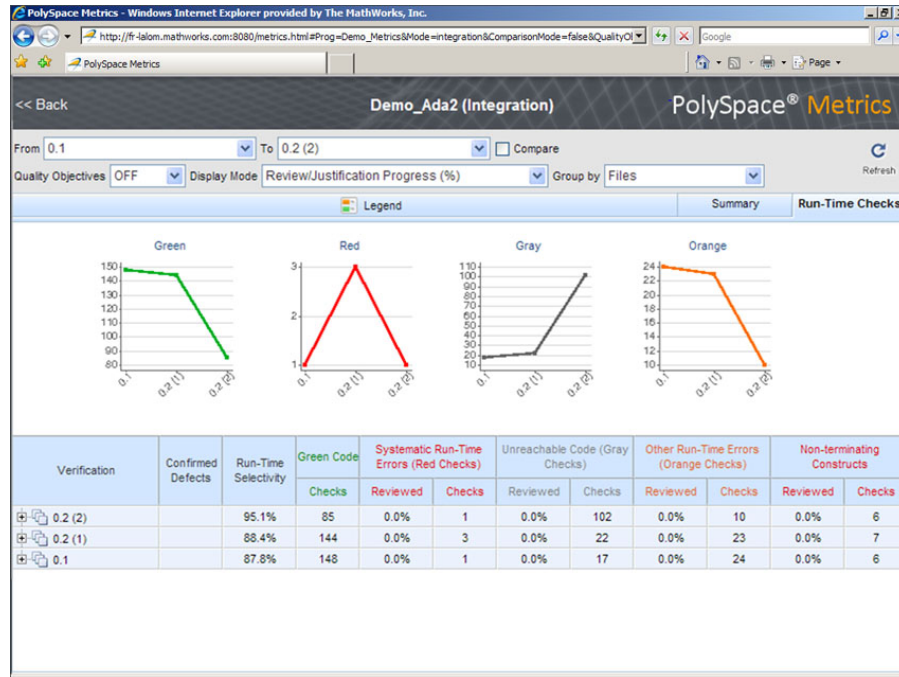
- Overflows and underflows
- Divide-by-zero and other arithmetic errors
- Out-of-bounds array access
- Read access to noninitialized data
- Dead code
- Dangerous type conversions

Prove the Absence of Certain Run-Time Errors and Track Software Quality Metrics

Traditional bug-finding tools either fail to report software errors (yielding false negatives) or produce too many warnings (yielding false positives). Dynamic testing, which typically uses a finite number of test cases, may miss errors entirely. Polyspace Client for Ada verifies program execution, for each instruction, taking into account all

possible values of every variable at every point in the code. The results provide a formal diagnostic for each operation in the code.

You can also view a summary of your software quality metrics on a Web browser dashboard (requires Polyspace Server for Ada). The software quality metrics summary helps you review code verification results as your code evolves from the first iteration to the ultimate delivery version. By confirming the absence of certain run-time errors and measuring the rate of improvement in code quality, Polyspace Client for Ada enables developers, testers, and project managers to target, deliver, and assess code that is free of certain run-time errors.



Web dashboard (requires Polyspace Server for Ada). You can monitor the quality of your software and decide which part of the software needs more attention.

Create Artifacts for Certification

Using Polyspace Client for Ada, you can create reports and artifacts that show the final quality of the code, highlight sections that have been reviewed, and document the application’s run-time error status. You can create these reports in PDF, HTML, RTF, and other formats.

Integrate Code Verification as Part of a Continuous Verification Process

You can automate verification job scheduling and integrate it with e-mail notifications (Polyspace Server for Ada required).

You can assign a Polyspace client to schedule the posting of a job to the server, and get notified by e-mail when the results are available. Results will contain only the differences compared with the previous version of your code. The server computes these differences automatically.

You can define the frequency of these analyses and the e-mails you want your users to receive when the results are available. Also, you can define which characteristics of the build process you want the automated verifications to encompass.

Resources

Product Details, Demos, and System Requirements

www.mathworks.com/products/polyspaceclientada

Trial Software

www.mathworks.com/trialrequest

Sales

www.mathworks.com/contactsales

Technical Support

www.mathworks.com/support

Online User Community

www.mathworks.com/matlabcentral

Training Services

www.mathworks.com/training

Third-Party Products and Services

www.mathworks.com/connections

Worldwide Contacts

www.mathworks.com/contact

© 2010 The MathWorks, Inc. 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.