

Polyspace Client for C/C++ 8.1

Prove the absence of run-time errors in source code

Polyspace Client™ for C/C++ 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 C/C++ 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

- File- and class-level software component verification
- Formal methods-based abstract interpretation
- Display of run-time errors directly in code
- MISRA-C[®]:2004, MISRA-C++:2008, and JSF++ coding standard enforcement, with direct source file links
- Cyclomatic complexity and other code metrics
- Eclipse™ and Microsoft® Visual Studio® IDE integration

```
static void Pointer_Arithmetic (void)
{
    int array[100];
    int i, *p = array;

    for(i = 0; i < 100; i++, p++)
        *p = 0;

    if(get_bus_status() > 0) {
        if (get_oil_pressure() > 0)
            *p = 5;
        else
            i++;
    }

    i = get_bus_status();
    if (i >= 0) { *(p-i) = 10; }

    if ((0 < i) && (i <= 100)) {
        p = p - i;
        *p = 5;
    }
}
```

Green: reliable

Red: faulty

Gray: dead

Orange: unproven

Proven

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

Working with Polyspace Client for C/C++

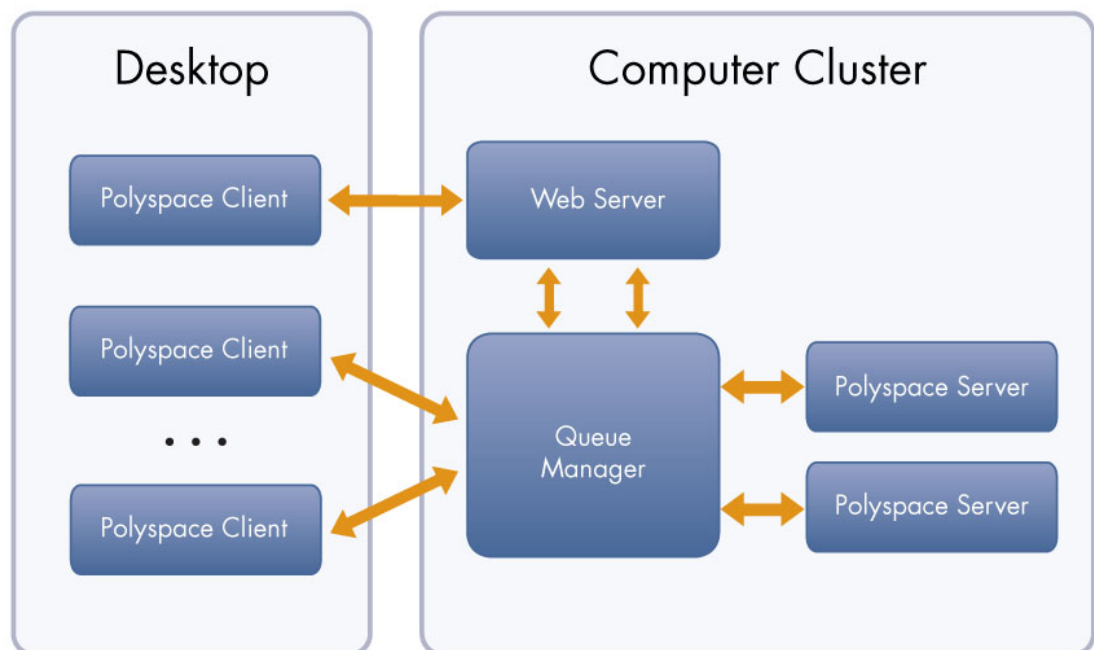
Polyspace Client for C/C++ provides management and visualization capabilities for verifying software components on a desktop computer. It processes file-by-file or class-by-class verification as soon as the source code is written, updated, or generated. When used with [Polyspace Server™ for C/C++](#), Polyspace Client for C/C++ lets you submit verification jobs to computer clusters.

Using the Polyspace Client for C/C++ command line or graphical user interfaces with Visual Studio or Eclipse, you can:

- Define which components, files, or classes of the source code you want to verify
- Customize a project by target, cross-compiler, or other options
- Check code for compliance with [MISRA-C:2004](#), [MISRA-C++:2008](#), or [JSF++](#) (Joint Strike Fighter Air Vehicle C++) standards
- Submit multiple verification jobs to Polyspace® servers running on a computer cluster or computer farm
- Ensure that the appropriate [software quality objectives](#) are met along your code life cycle using a Web browser (requires Polyspace Server for C/C++)
- Download data from the server to report or visualize verification results
- Focus on differences in run-time behavior compared with previous results

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

- Detecting run-time errors
- Enforcing coding standards
- Proving the absence of certain run-time errors
- Tracking the progress of your software quality objectives
- Creating artifacts for certification



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

Detect Run-Time Errors

Polyspace Client for C/C++ 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, underflows, divide-by-zero, and other arithmetic errors
- Out-of-bounds array access and illegally dereferenced pointers
- Always true/false statement due to dataflow propagation
- Read access to noninitialized data
- Dead code
- Access to null `this` pointer (C++)
- Dynamic errors related to object programming, inheritance, and exception handling (C++)
- Noninitialized class members (C++)
- Dangerous type conversions

Enforce Coding Standards

Polyspace Client for C/C++ supports the detection of MISRA-C:2004, MISRA-C++:2008, and JSF++ coding violations. The product also generates code metrics and produces reports that you can use to monitor and help improve code reliability and quality.

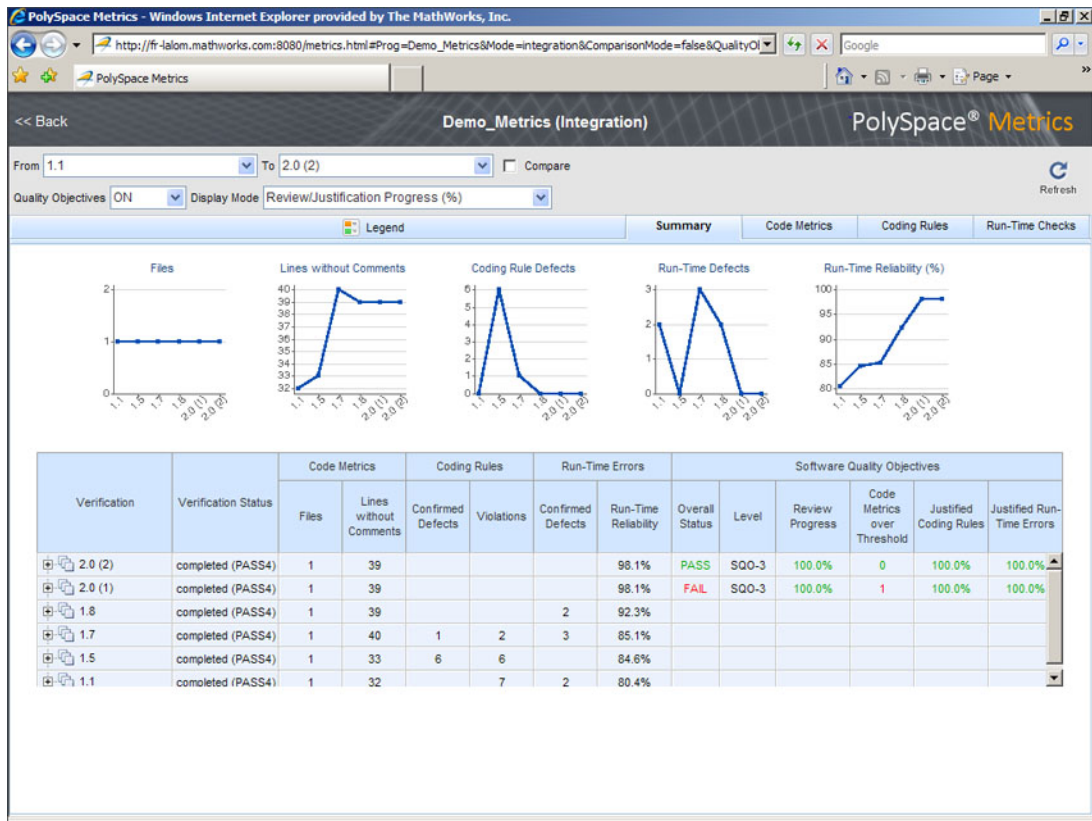
You can configure the client to focus on all the rules of the standard or individually select the rules you want to enforce. You can also mirror this configuration on the Polyspace server to make sure that the same coding rules are enforced within your team.

You can decide to fix the errors by double-clicking on the errors within Polyspace, or to justify the coding rule violations in the results for the purpose of documentation or code comments. The Polyspace user interface lets you focus on differences from the previous runs to avoid reviewing a check twice.

Prove the Absence of Certain Run-Time Errors and Track the Progress of Your Software Quality Objectives

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 C/C++ 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 define a centralized quality model on the Polyspace server (requires Polyspace Server for C/C++) to track coding rules violations, code metrics, and run-time errors. Using these metrics, you can track your progress toward predefined [software quality objectives](#) 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 C/C++ enables developers, testers, and project managers to target, deliver, and assess code that is free of run-time errors.



Web dashboard (requires Polyspace Server for C/C++). You can monitor the progress of the software quality and decide which part of the software needs more attention.

Create Artifacts for Certification

You can use Polyspace Client for C/C++ and Polyspace Server for C/C++ with [IEC Certification Kit \(for IEC 61508 and ISO 26262\)](#) and [DO Qualification Kit \(for DO-178B\)](#) in the certification process for projects based on these industry standards.

You can create reports and artifacts that show the final quality of the code, highlight sections that have been reviewed, generate code metrics, and document the application of coding rules and 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 C/C++ 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, the quality model you want to apply for a given portion of your code base, 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/polyspaceclientc

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