



# Accelerate Safety Standards Compliance for C & C++ with Automated Unit & Integration Testing



The Software Quality Company

Driving embedded  
software quality



## Key Benefits

Automated test harness and test case creation

Extensive platform support and toolchain integrations

Easy to use Eclipse® GUI

Tests edited directly in C/C++

Bi-directional requirements traceability

Unique function call control to simulate and intercept calls

Flexible user code injection

Support for Test Drive Development

Integrated code coverage analysis

Automated regression testing

Test maintenance for code changes

Free tool certification kit for all major safety standards

## Works with your environment

Cantata installs on Windows® and Linux® host operating systems, with a Built-on-Eclipse® IDE, as an Eclipse Ready™ plug-in set, and as Visual Studio Code® Extensions. It supports GCC and Microsoft® Visual Studio® compilers, and is also integrated with an extensive set of embedded development toolchains:

- ✓ IDEs / RTOSs
- ✓ Cross-Compilers
- ✓ Debuggers
- ✓ Build / Continuous Integration
- ✓ Software Configuration Management
- ✓ Requirements Management

Non-Eclipse® toolchain build settings are automatically imported into Cantata test projects. To confirm how your tools and platforms are supported, please contact QA Systems.

## Unrestricted embedded target use

For target environments, a built-in wizard provides unlimited deployment and use without any license restrictions. Cantata deployments consist of libraries and configuration options, binary compatible with your code. These are tested and controlled for certified use on safety related projects. Deployment switching makes it easy to use one set of tests across multiple targets for different product variants.

## Easy and flexible testing on target

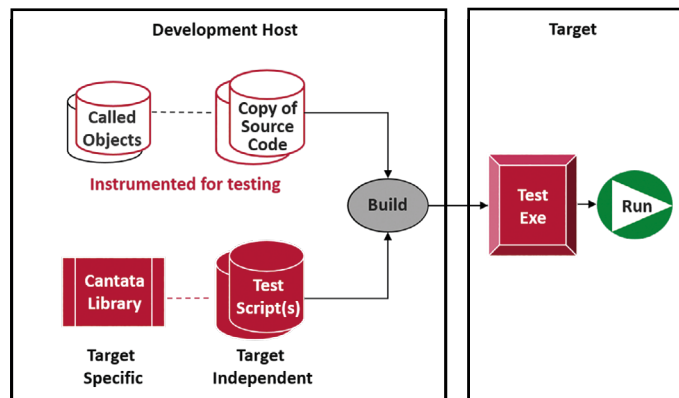
Cantata tests (platform independent test scripts in C/C++ and platform specific library deployments) are built as C/C++ executables, downloaded, and run on a target platform just as you would with your own code on a:

- ✓ Simulator
- ✓ Emulator
- ✓ Physical Target-Board

Functional and code coverage test results are directed back to the host for diagnostics and reporting. The process is completely automated using Cantata Makefiles, test scripts and platform customisations for easy and flexible on-target testing from the GUI or CLI.

## Types of Testing Supported

- black-box / white-box
- positive / negative
- requirements / robustness
- single / large input data sets
- procedural / object oriented
- from code / from headers
- call simulation / interception
- isolation / integration
- host / target execution
- new / regression



User code is driven by portable test scripts with target compatible libraries and built as a single test executable to run on multiple host or target platforms. Instrumentation is used for white-box access and code coverage, so production code is never modified for testing.

# Why Industry Leaders use Cantata

## Cut the cost of standards compliance

Cantata meets the dynamic testing requirements of software safety standards. It is a single solution for dynamic unit and integration testing on host and target platforms.


Certification of development tools can be a heavy compliance cost burden. Cantata has been independently certified by SGS-TÜV SAAR GmbH and provides a tool certification kit with everything needed out-of-the-box, available free of charge.

Understanding how to comply with software safety standards is complex and time consuming. Comprehensive guidance is provided free of charge for using the full feature set in Cantata to meet dynamic testing requirements of specific standards. This combination of guidance and Cantata capabilities accelerates the achievement of standards compliance.

### SGS-TÜV Certified for:

-  ISO 26262
-  EN 50128
-  EN 50657
-  IEC 62304
-  IEC 61508
-  IEC 60880

### Qualifiable for:

-  DO-178C / DO-330
- Other Standards as Required

## Reduce risk of software failure

Product recalls and infection of the wider brand and corporate reputations can far exceed the development cost of each application. Unit testing is the most thorough way to test application code and prevent bugs within shipping devices.

Project over-runs can be mitigated by shifting verification effort to the earliest stages in the software development lifecycle. This reduces the risks of delays during later testing stages because unit tested components are easier and more predictable to integrate.

Fitness for purpose litigation against companies and individuals is now an increasing risk. Where companies fail to employ accepted industry practices such as thorough unit testing with Cantata, they cannot use the “state of the art” legal defence against such litigation.

## Lower testing costs

Testing earlier lowers costs by minimising code rework later in the development lifecycle. Developers can identify defects with Cantata unit and integration testing as soon as each component is available.

The high cost of standards compliant unit and integration testing can be dramatically lowered through automation. Satisfying the dynamic testing requirements of safety standards is accelerated by Cantata automation of:

- › Test framework generation
- › Test case generation
- › Baseline tests on legacy code
- › Test execution on host or target
- › Regression testing in Continuous Integration
- › Results diagnostics and report generation

Integrating tools into a toolchain can add hidden testing costs. Cantata’s tight integration with cross-compilation environments, DevOps and CI/CD workflows and its intuitive C/C++ code tests in Eclipse® GUI or code editors make it easy to slot into any toolchain. These integrations lower the tool learning curve and accelerate the testing activity, lowering overall testing costs.

## Shorten time to market

Industry leaders recognise the need to ship faster without endangering quality. Cantata tests provide two key time advantages for development managers:

- › Team collaboration and efficiency is improved with structured consistent tests and certification ready reports.
- › Integration times are shorter and more predictable when integrating individually proven software components.



## Flexible test framework

A flexible test framework (test scripts and supporting library) enables any combination of testing styles for both unit and scalable integration testing.

Tests can be generated, edited and run in:

- › An Eclipse® GUI
- › Any C/C++ code editor for code centric editing
- › Visual Studio Code with Cantata VSCode extension pack

## Test Driven Development (TDD)

Cantata tests are created from function prototypes within header files. Cantata for TDD enhances traditional black-box TDD techniques, by giving access to fully featured white-box testing on encapsulated code internals such as private/static data and functions.

## Black and white box testing

Highly automated test cases provide power and precision for black-box testing, and more efficient thorough white-box testing. Black-box testing is enabled with user-selected or predefined parameterised looping tests, a combinatorial effect calculator, and CSV import/export for large data sets.

Precise white-box testing via Cantata instrumentation automatically accesses encapsulated code directly from the test script, without conditional compilation, giving control over both static/private functions and data. Users can inject extra code (via instrumentation) for testing purposes, without modifying the production code.

## Robustness testing

Robustness testing is made easy with Cantata Rule Sets of predefined values for basic data types in looping test cases. All accessible global data is automatically initialised and checked for inadvertent changes.

## Object-oriented testing

Cantata object-oriented style tests are implemented as classes for testing methods, templates, or clusters of classes. They feature automated:

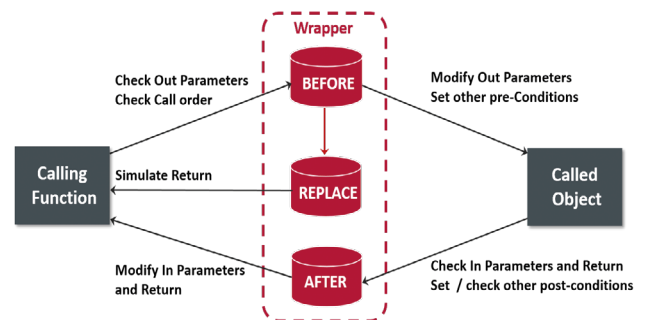
- › Test case reuse via a parallel inheritance hierarchy
- › Test class inheritance structure for inherited classes
- › Concrete implementation of abstract base classes (ABCs) or pure virtual methods (PVMs)
- › Resolution of dependencies on undefined references that are not directly called by the code

## Unique function call control

Cantata automatically generates test controls to both simulate (stub) and intercept (wrap) all function calls from the software under test, providing:

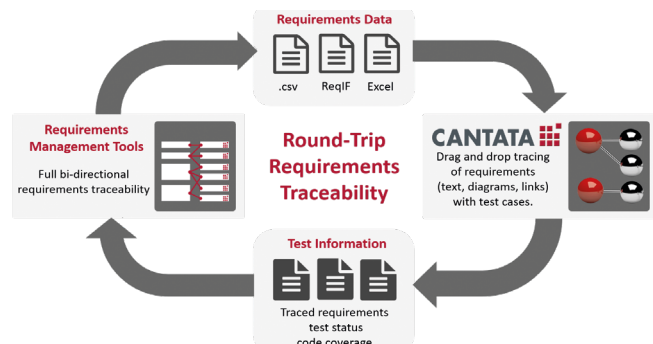
- › Optional automatic checks on parameters and data
- › Multiple instances for differential call behaviour
- › Flexible call order verification in each test case
- › Interface error detection and error injection
- › Control coupling testing

Wrappers intercept calls to verify the actual, not the assumed, or simulated behaviour of the called object. Where simulation is not possible or desirable (internal calls at integration, OS calls, hardware interfaces etc.), wrappers provide powerful call control unique to Cantata.



## Requirements tracing

Requirements or test plan sets are imported to a Cantata server. Relationships are assigned with a drag-and-drop interface, and then exported with Cantata test results status and code coverage information for bi-directional traceability.

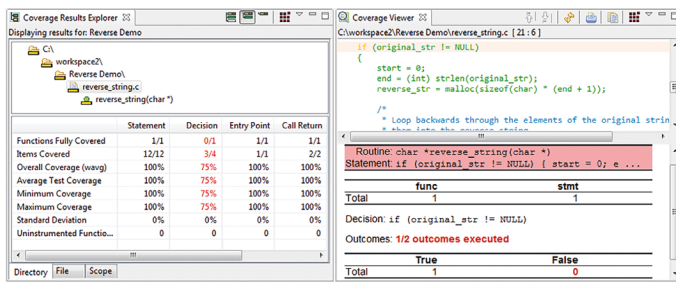


Import/export can be via CSV, Microsoft® Excel®, or Requirements Interchange Format (ReqIF™), to fit your workflow and specific requirements management tool version (e.g. IBM® Rational® DOORS®, PTC Windchill® & codebeamer®, Siemens Polarion® ALM™ and Visure Requirements ALM®).



## Code coverage

Cantata code coverage provides objective measurement of how thoroughly tests have executed the source code (whether or not driven by Cantata tests). Standard specific Cantata coverage Rule Sets make it easy to use by automating the instrumentation, data reporting and integrated checking of required code coverage levels.



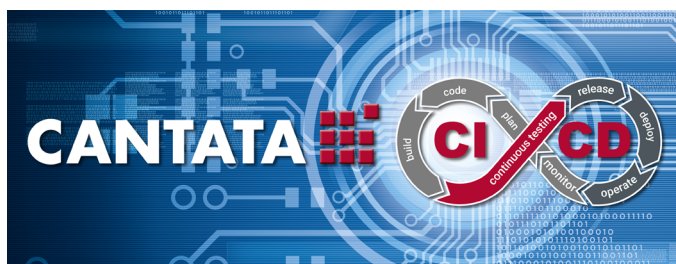
Code Coverage is measured using the following metrics:

- › Entry points
- › Call Returns
- › Statements
- › Basic Blocks
- › Decisions (branches)
- › Conditions
- › MC/DC
- › Loops
- › Relational Operators

Cantata Build Variant Coverage gathers data on source code, when defines are used to build executable code in different variants, providing analysis and certified reporting of coverage aggregated over all build variants.

Pin-point diagnostics can filter or aggregate coverage for complete project code-trees, drilling down to individual code constructs within each line of code, by test case, test run and metric type and code context (inheritance, threads, states, data coupling etc.) and source build variant. Automatic test case optimisation aids test case vector selection from large data sets and reduces regression testing overhead.

## Continuous testing in CI/CD



[LEARN MORE](#)

Cantata regression tests can be run in a CI/CD pipeline each time code is checked into a repository. A dedicated global Cantata CI/CD subscription license can allow up to 500 builds to run in parallel, independently of full Cantata user licences.

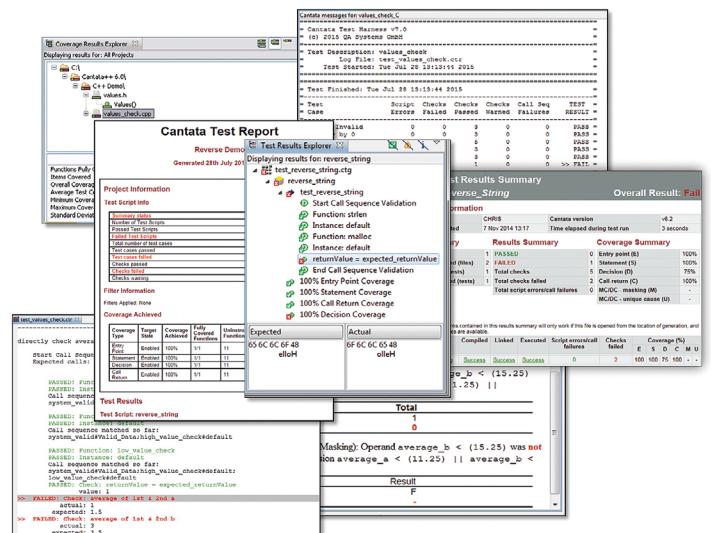
## AutoTest

Automatic generation of test case vectors can exercise 100% code coverage while checking data, parameters, and call order. This can plug code coverage gaps or create complete unit tests easily linked to requirements, so reducing reliance on manually generated unit tests and expensive system tests.

## Code Change Based Testing

Cantata Code Change Analysis helps automate unit test maintenance. Changes which impact existing tests are identified and suitable updates suggested. Test scripts are then automatically refactored.

## Diagnostics and reports



Cantata provides powerful filterable diagnostics of test and code coverage results within the Eclipse® GUI, and flexible offline user configurable reports in XML and certification-ready ASCII text and HTML.

## Team Reporting test status dashboard

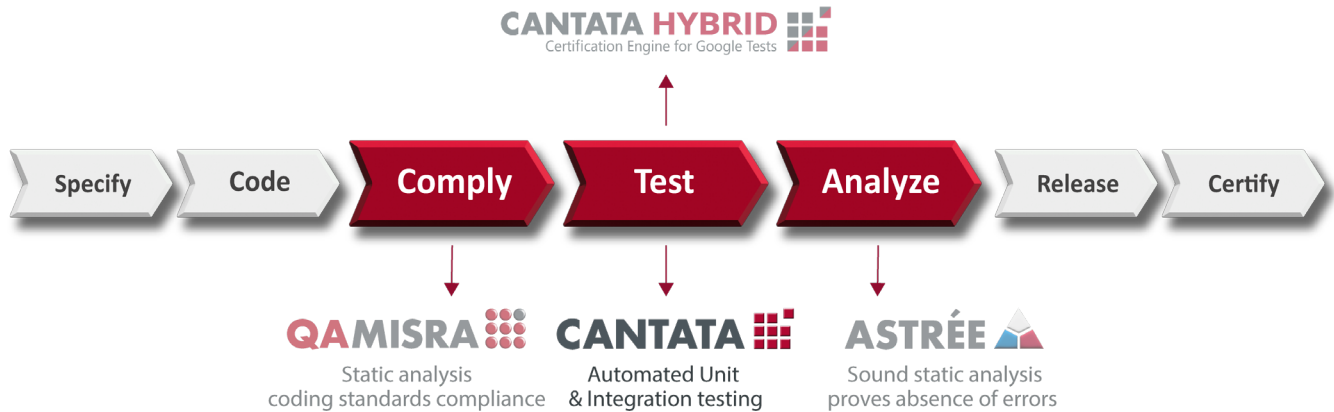
Cantata Team Reporting (add-on), with a client-server architecture, web interface and REST API, provides current testing status and historical trends over multiple codebases.



[LEARN MORE](#)

# Verification Centric Tools

QA Systems static analysis and dynamic software testing tools support verification in the linear flow of software development below. We recommend applying sequential approach to these verification stages with tools targeted for each purpose.



## Comply

Use **QA-MISRA** for fast coding standard compliance at the developer's desktop first.

## Test

Use **Cantata** for automated dynamic execution of the standard compliant software.

Use **Cantata Hybrid** to generate certified Cantata test results from existing Google tests.

## Analyze

Use **Astrée** for proving absence of run-time errors on whole application.

NB: Astrée uses the same configuration as QA-MISRA, so the effort to apply it later to a QA-MISRA project is low.

## Special shared license bundle option

QA-MISRA and Cantata share the same license technology. This allows customers to obtain a bundled solution for both tools to share the same concurrent user license pool, as well as the tools being integrated together in the Eclipse® based IDE.

When QA-MISRA is purchased as a bundle with Cantata or when an existing Cantata license is converted to a bundle, there are very attractive combined prices available. Please contact us for more information.

### Start free trial

Take Cantata for a test drive with your own code and development environment.

 **START TRIAL**

### What to expect in trial

- › A bespoke demo kick-start and Q&A session
- › Very simple installation
- › Simple configuration for your compiler
- › Unrestricted use, time-limited trial license

### Learn more



[qa-systems.com/cantata](http://qa-systems.com/cantata)



**QA Systems Group** | [sales@qa-systems.com](mailto:sales@qa-systems.com) | [www.qa-systems.com](http://www.qa-systems.com)

With offices in **Stuttgart, Germany** | Bath, UK | Milan, Italy | Boston, United States | Da Nang, Vietnam | Bengaluru, India

QA-MISRA® and Cantata® are registered trademarks of QA Systems GmbH. ©Copyright QA Systems GmbH.

Astrée® is a registered trademark of AbsInt Angewandte Informatik GmbH, developed under license from the CNRS/ENS.

"MISRA" and "MISRA C" are registered trademarks owned by MISRA Consortium Limited. QA-MISRA is an independent tool of QA Systems and is not associated with the MISRA Consortium Limited. Google & GoogleTest are registered Trademarks of Google LLC.