

Static Analysis



Accelerate C/C++ Coding Standards Compliance & Code Quality Insights



Driving embedded
software quality





Key Benefits

Fast C/C++ coding standards compliance analysis

Multiple coding standards

Reduces effort wasted on assessing false positives

Code quality insight metrics & visualizations

Permanent floating licensing

Improves code complexity, maintainability, portability

Contextual rule explanations

Configurable compliance rulesets & reports

Full traceability of issues

Robust finding classification

Interactive result exploration

Tracking and visualization of project progress and analysis revisions

Full batch mode execution

3rd party integrations

Free tool certification kit for all major safety standards

Automated coding standards compliance

All the standards for C and C++ in one tool

With QA-MISRA there are no hidden extras, coding language variants, or compliance module add-ons. It provides a single solution to automatically check your C or C++ source code for compliance against the most common international software safety and security standards below.

- **MISRA C:2012**
 - C Coding Guidelines incl. Amendments 1, 2 & 3
- **MISRA C:2023**
 - C Coding Guidelines
- **MISRA C:2025**
 - C Coding Guidelines
- **MISRA C++:2008**
 - C++ Coding Guidelines
- **MISRA C++:2023**
 - C++ Coding Guidelines
- **AUTOSAR C++14**
 - C++ for Adaptive AUTOSAR
- **HIS Metrics**
 - Hersteller Initiative Software
- **JSF AV C++**
 - C++ coding standard for JSF (F-35 aircraft program)
- **CERT C/C++ 2016**
 - C/C++ Coding Guidelines
- **CWE 4.14**
 - Common Weakness Enumeration
- **ISO TS 17961:2013**
 - C Secure Coding Rules

QA-MISRA analyses all the source code language versions below:

- › ISO/IEC9899:1990 (C90)
- › ISO/IEC9899:1999 (C99)
- › ISO/IEC9899:2011 (C11)
- › ISO/IEC9899:2018 (C18)
- › ISO/IEC 14882:2003 (C++03)
- › ISO/IEC 14882:2011 (C++11)
- › ISO/IEC 14882:2014 (C++14)
- › ISO/IEC 14882:2017 (C++17)
- › ISO/IEC 14882:2023 (C++20)
- › ISO/IEC 14882:2024 (C++23)

Comprehensive rule compliance



The development of QA-MISRA took place in close cooperation with experts in the MISRA committee. Full compliance matrices for each coding guidelines standard classify the degree of support provided by QA-MISRA for each individual rule.

A comprehensive knowledge base is available in QA-MISRA with extensive sample code sets, helping developers comply with these standards and automated reports provide clear and definitive compliance status of analysed code.

Why Targeted Static Analysis?

Cut the cost of coding standards compliance

All safety critical software development standards highly recommend or mandate the use of static analysis tools to ensure compliance with safe and secure coding best practice. International functional safety coding standards such as MISRA and AUTOSAR, and security standards such as CERT and CWE, have become the de-facto coding standards. QA-MISRA automates the static analysis of C and C++ source code targeted for complying with all these coding standards in a single tool.

Certification of static analysis tools can be a heavy compliance cost burden. QA-MISRA 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. A Qualification Support Kit is also available.

SGS-TÜV Certified for:

-  ISO 26262
-  EN 50716
-  IEC 62304
-  IEC 61508
-  IEC 60880

Qualifiable for:

-  DO-178C / DO-330

Other Standards on Request

Detect software errors early

Through static analysis with QA-MISRA, dangerous structures, problems with security, maintenance and portability can easily be found as soon as source code components are written. The earlier software errors are identified and eliminated in the development process, the more your costs are reduced. QA-MISRA checks for over 900 potential software errors in source code.

Even ISO-standard-compliant C/C++ can behave differently than expected, because not all problems are classified as incorrect. Static analysis with QA-MISRA uncovers problems with your code that are often overlooked by developers and compilers. Ensuring your code complies with functional safety coding standards such as MISRA and AUTOSAR, or security standards such as CERT and CWE, provides developers with a fully automated code review early in development and saves them time.

Reduce risk of software failure

Product recalls and jeopardy of brand and corporate reputations can far exceed the development cost of each application. Static analysis for coding standards compliance as soon as source code is written is the first stage of software verification to reduce risk of failure. Project over-runs can be mitigated by shifting verification effort to the earliest stage in the software development lifecycle. This reduces the risks of delays during later testing and complete system analysis stages because compliant components are easier and more predictable to integrate.

Fitness for purpose litigation against companies and individuals is now an increasing risk. Companies who fail to follow industry best practices, such as coding standards compliance, cannot use the “state of the art” legal defence against such litigation.

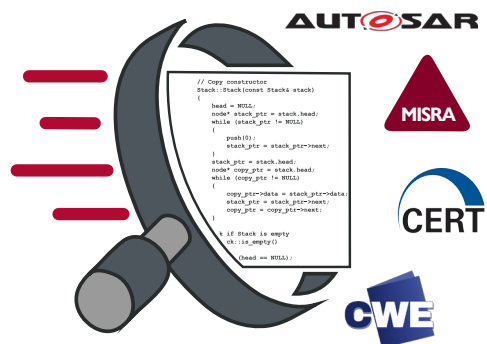
Shorten time to market

Industry leaders recognise the need to ship faster without endangering quality. QA-MISRA provides two key time advantages for development managers:

- Not delaying static analysis until a full system is available, means standards compliant components are easier to unit test.
- Automated code reviews for coding standards compliance, trains developers to produce code with lower error rates.



Fast static analysis of source code



QA-MISRA analyses extensive and complex software packages very quickly, regardless of the size of the codebase. By targeting the static analysis on the coding standards compliance rules, developers can apply QA-MISRA from individual components to full systems as code is developed. This analysis speed becomes especially valuable for teams deploying automated CI/CD pipelines, to verify compliance quality on every branch check-in.

Zero False Positives & False Negatives

QA-MISRA produces zero false negatives and zero false positives on syntactic rules. When also coupled with the sound static analyzer Astrée, it then produces zero false negatives and very low false positives on semantical rules.

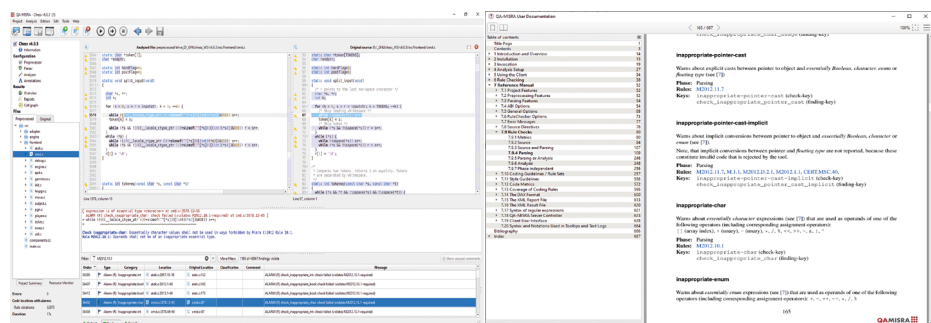
Zero false negatives mean more confidence the analysis has not missed problems. Lower false positives mean less wasted effort on assessing and discarding as 'false' a reported rule violation (alarm).

Delta Analysis

Delta Analysis provides immediate insight into changes between analysis runs. When this function is enabled, it highlights what's new, what has changed, and what remains unchanged, directly within the GUI. Rule violations can be easily filtered by their delta status, while detailed delta information is also included in analyzer logs and reports helping teams focus on what matters most and work more efficiently.

Contextual rule explanations help developers understand and learn

For each rule violation detected, QA-MISRA details alarm explanations to reduce the effort to understand it's root cause.

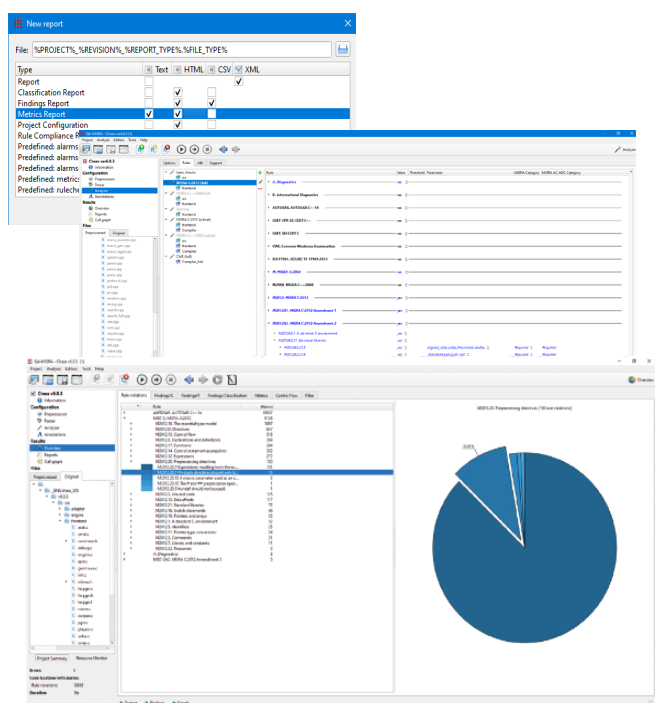


Configurable rulesets

Drill-down selection and combinations of different standards to individual rules and checks for each project, provide flexible configuration for projects with mixed C & C++, autogenerated code or manually written code. All alarm messages and rulesets can be selected, interactively filtered and explored in the GUI, or stored in configuration files for command line batch execution.

Offline detailed reports

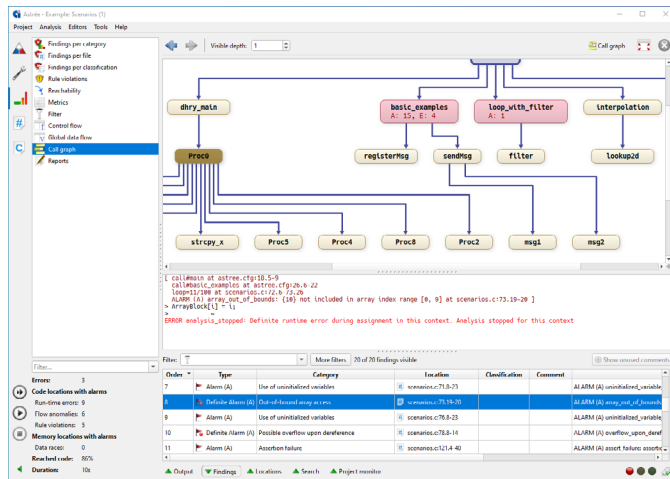
QA-MISRA generates configurable reports in multiple open standard formats (including SARIF) for offline compliance reports. This includes all alarm messages on source file level to be used in further automated processes or automatically distributed to offline users.



All alarm messages refer to either original source file (Source phase checks) or a single occurrence in the source (even if the original code is included in multiple pre-processed files (Parsing and Analysis phase checks), identifying sole root cause.

Call graph visualization

Folding call graphs, contextual rule findings, source code links and CSV exports provide graphical insights into code quality.



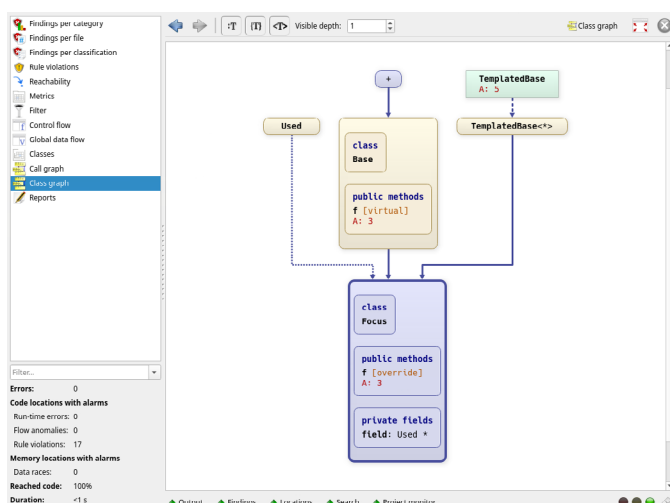
Code metrics

Code quality can be analysed through numeric metrics. In addition to the HIS metrics, many commonly used metrics are computed, and threshold values automatically checked.

C++ classes visualization

A filterable view in the GUI presents all classes and class relationships for the selected code, including those defined inside the C++ standard namespace.

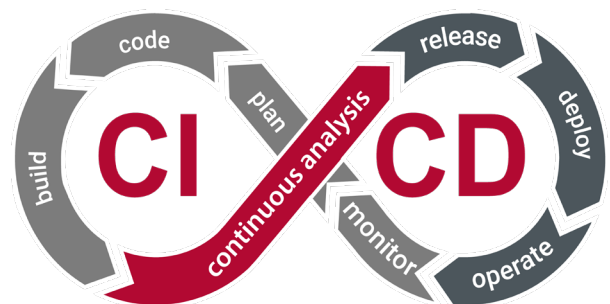
A class graph visualizes a selected class together with its relations to others. Possible relations are inheritance, template instantiation/specialization, and usage as type of field members. For each individual class, detailed information on the respective field and method members can be inspected.



Permanent floating licensing

Unlike most static analysis tools, the QA-MISRA client/server architecture supports centralized user licence management options. Licences can be permanent or subscription and node-locked to a single machine or floating network concurrent user. This provides greater flexibility for teams to queue processing of analyses, without limiting each licence to named individuals or size of source code.

Continuous integration



Analysis projects can be created and re-run in batch mode for full command line continuous integration. Analysis specification is defined in configuration files generated from the GUI or even automatically on-the-fly from XML and scripts.

This allows for a high degree of automated report generation and resulting actions (e.g. quality gates in DevOps or CI/CD pipelines).

Platforms & Integrations

QA-MISRA is available on Linux and Windows platforms. Pre-processing and parsing knowledge of the compilation environment is captured in project set-up and used in the analysis to support embedded software.

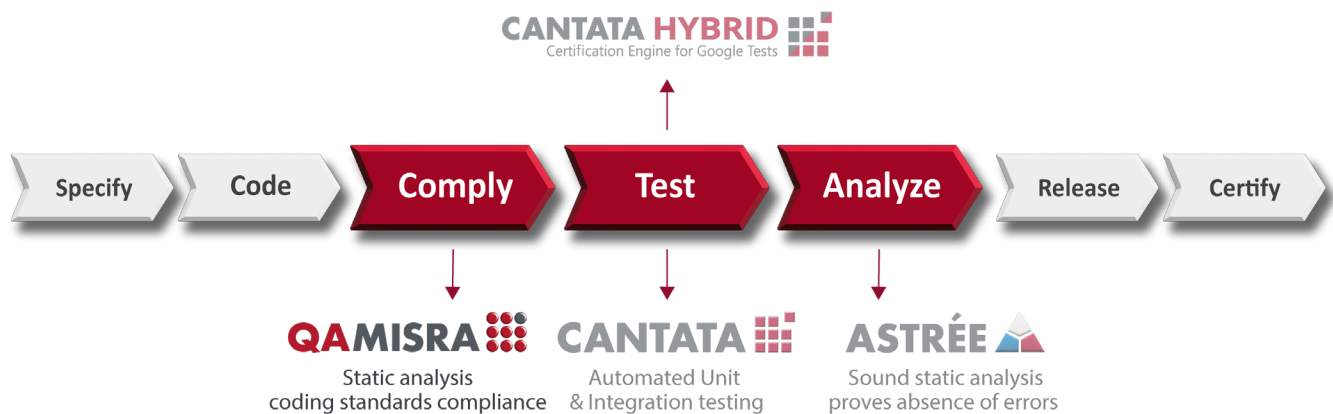
In addition to the GUI client and full command line interfaces QA-MISRA is integrated with the following:

- › Eclipse®
- › Arm® Keil® µVision® debugger
- › dSPACE® Targetlink®
- › MathWorks® MATLAB®/Simulink®
- › Visual Studio Code®
- › Jenkins®

Please refer to the QA-MISRA Release Notes for versions supported.

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 a 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 licence bundle option

QA-MISRA and Cantata share the same licence technology. This allows customers to obtain a bundled solution for both tools to share the same concurrent user licence 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 licence is converted to a bundle, there are very attractive combined prices available. Please contact us for more information.

Start free trial

Take QA-MISRA 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 licence

Learn more



qa-systems.com/qa-misra



QA Systems Group | sales@qa-systems.com | 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 licence 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.