Astrée for C

Astrée is a static analyzer which signals all potential runtime errors, data races and further critical program defects. Astrée is *sound*, i.e., if no errors are signaled, this means there are no errors from the class of errors under investigation – the absence of errors has been proven. It reports program defects caused by unspecified and undefined behaviors according to the C99 norm, program defects caused by invalid concurrent behavior, and computes program properties relevant for functional safety.

Key features

- Analysis and verification at the module level, application level and whole-program level
- Safe and sound analysis results, based on a mathematically rigorous formal method
- Automatic OS-aware analysis for ARINC-653, OSEK, and AUTOSAR applications
- Full tracability of reported code issues
- Interactive result exploration
- Robust classification of findings
- Configurable report file generation
- Tracking and visualization of project progress and analysis revisions
- Client/server architecture featuring queue processing of analysis requests, and centralized user management and authentication
- Stand-alone tool with open interfaces and open file formats
- MATLAB integration and TargetLink coupling
- Automatic tool qualification according to safety standards
- Seamless RuleChecker integration:
  - Fast and easy to use
  - Enforcement of coding guidelines including MISRA-C:2004, MISRA-C:2012, and customized rule sets
  - No false positives and no false negatives on syntactical rules
  - Leverages Astrée results to guarantee zero false negatives and minimal false positives on semantical rules
  - Computation of code metrics: HIS metrics and customized metrics
  - Enforcement of metric thresholds

Error classes reported by Astrée include:

- integer/floating-point division by zero
- out-of-bounds array indexing
- erroneous pointer manipulation and dereferencing (null, uninitialized, and dangling pointers)
- data races (read/write or write/write concurrent accesses by two threads to the same memory location without proper mutex locking)
- inconsistent locking (lock/unlock problems)
- invalid calls to operating system services (e.g. OSEK-calls to `TerminateTask()` on a task with unreleased resources)
- integer and floating-point arithmetic overflows
- read accesses to uninitialized variables
- code Astrée can prove to be unreachable under all circumstances (note that this is not necessarily all unreachable code
- violations of optional user-defined assertions to prove additional runtime properties. These static assertions can be formulated with arbitrary side-effect free C expressions. When Astrée does not report an assertion failure alarm, the correctness of the asserted expression has been formally proven.
- violations of coding rules and code metric thresholds.
- non-terminating loops

Astrée is sound for floating-point computations and handles them precisely and safely. It takes all potential rounding errors into account.

Astrée computes data and control flow reports containing a detailed listing of accesses to global and static variables sorted by functions or variables, and caller/callee relationships between functions. The analyzer can also report each potentially shared variable, the list of asynchronous tasks accessing it, and the types of the accesses (read, write, read/write). The call graph is visualized and can be interactively explored.

Key benefits

Astrée considers all possible program executions with full control and data coverage and can report every source code expression and statement that may lead to a defect from one the supported defect classes. It relies on abstract interpretation – a provably correct formal method – and does not require the program under analysis to be instrumented, executed, or stimulated by test cases. The tool can be used on handwritten code, automatically gener-

Astrée for C

ated code, or by any combination thereof. Open interfaces and full batch mode execution make Astrée ideally suited to be used in continuous verification frameworks. Tool couplings, e.g., to dSPACE TargetLink, are available that provide a seamless integration in existing development environments. By using its Qualification Support Kit and AbsInt's Qualification Software Life Cycle Data Reports, Astrée can be automatically qualified according to all contemporary safety norms (e.g., ISO 26262 or DO-178B/C). Astrée enable users to

- reduce the risk of failure in the field by finding all potential run-time errors and data races,
- meet mandatory verification goals of contemporary safety standards,
- improve software quality, and
- reduce time-to-market.

Supported standards

- ISO/IEC 9899:1999

System requirements

- Windows: 64-bit Windows 7 SP1 or newer
- Linux: 64-bit CentOS/RHEL 6 or compatible
- 4 GB of RAM (16 GB recommended)
- 4 GB of disk space

Also available

The following AbsInt products are also available for this target:

- RuleChecker
- Qualification Support Kit
- Qualification Software Life Cycle Data Report

More information

- Visit our website: www.absint.com
- Speak with a product specialist: call +49 681 383 600

About AbsInt

AbsInt provides advanced development tools for embedded systems, and tools for analysis, optimization and verification of safety-critical software. Our customers are located in more than 40 countries worldwide. We have distribution agreements with major software distributors in Asia, North America, Middle East, and throughout Europe.

Our headquarters

Science Park 1
66123 Saarbrücken, Germany
Phone: +49 681 383 600
Fax: +49 681 383 60 20
Email: info@absint.com
Web: www.absint.com