Updates in 2020.1

Release date: April 28, 2020

This release includes the following enhancements:

New Versioning Convention

All Parasoft products, including C/C++test, now follow a new versioning scheme: *YYYY.release*.

Importing and Reviewing Requirements

You can now review information about requirements and test definitions imported from a requirements management system (RMS) in your IDE. C/C++test can visualize the correlations between imported requirements and test definitions in the Requirements view to help you create meaningful unit tests – without having to leave your IDE. See Indicating Requirement and Test Correlations.

Extended Automotive Compliance Pack

We've extended the Automotive Pack to help you achieve compliance with the automotive standards.

MISRA C:2012 Amendment 2

We've added new rules to the *MISRA C 2012* test configuration to fully cover the guidelines in MISRA C 2012 Amendment 2.

AUTOSAR C++ 14

We've added new rules to the *AUTOSAR C++14 Coding Guidelines* test configuration and updated the existing rules to extend support for AUTOSAR C++ 14 .

Increasing Code Coverage with Coverage Advisor

You can now effectively implement strategies for increasing code coverage with Coverage Advisor. Coverage Advisor provides guidance for covering gaps in code coverage by analyzing code and displaying available solutions of how to set up your tests to cover uncovered lines. The information about required dependencies and pre-conditions for parameters, global variables, and function calls is displayed in the Coverage Advisor view. This allows you to reduce the time and effort of manually creating meaningful test cases and appropriate stubs. See Working with Coverage Advisor.

Support for Working with Docker Containers

We've added support for working with Docker containers. You can now run analysis and execute tests utilizing a C/C++ toolchain or testing environment provided by a Docker container. This allows you to leverage C/C++test's capabilities in a containerized environment to ensure consistency across the team and multiple development cycles. See Analysis and Testing with a Docker Container.

Extended Support for CMake-based Projects

C/C++test now ships with an extension for CMake to streamline analysis and testing of CMake-based projects. You can now define C/C++test projects directly in the CMake build files to automatically generate C/C++test project files during the CMake build. See Integrating C/C++test into a CMake Build.

Support for Compilers

We've added support for the following compilers:

Compiler Name	Compiler Acronym
Vx-toolset for TriCore C/C++ Compiler 6.3	vxtc_6_3
Wind River Clang 8.0.x	wrclang_8_0

Support for Source Control Management Systems

We've added support for:

- Git 1.8, 1.9, 2.x
- SVN 1.10, 1.11, 1.12, 1.13
- Microsoft Team Foundation Server 2017, 2018, 2019

See Deprecated Support for Environments, Systems, and Configurations for information about deprecated and removed support for source control management systems.

New and Updated Test Configurations

We've added the following test configurations:

- Run Unit Tests in Container

We've updated the following test configurations:

- AUTOSAR C++14 Coding Guidelines
- CWE Top 25 + On the Cusp 2019
- Flow Analysis Aggressive
- Flow Analysis Standard
- High Integrity C++
- MISRA C 2012
- MISRA C++ 2008
- SEI CERT C Guidelines
- SEI CERT C Rules
- SEI CERT C++ Rules

New and Updated Code Analysis Rules

We've added new static analysis rules to extend coverage of compliance standards, with a special focus on the AUTOSAR C++ 14 and MISRA C:2012 Amendment 2; see New Rules and Updated Rules for the lists of new and updated rules.

Other Updates

- We've extended support for authentication via OpenID Connect. C/C++test now supports keystores with certificates for multiple users.
- We've upgraded the built-in Python runtime to Python 2.7. Python-related errors from Rule Wizard rules are now reported as setup problems.
- You can now configure C/C++test to return a non-zero exit code when a setup problem is reported. See Command Line Exit Codes.
- We've improved performance of test configurations where multiple duplicates of the same flow-based rule are enabled.

Deprecated Support for Environments, Systems, and Configurations

Support for the following environments and systems is now deprecated and will be removed in future releases.

Windows 7

Support for Windows 7 is deprecated, following the system's EOL.

32-bit Platforms

Support for Windows 32-bit and Linux 32-bit is deprecated. For details about the deprecation policy, contact your Parasoft representative.

Compilers

Support for the following compilers is deprecated:

- QNX GCC 4.2.x
- QNX GCC 4.4

IDEs

Support for Texas Instruments Code Composer Studio 6.0 is deprecated.

Source Control Management Systems

Support for the following SCMs is deprecated:

- AccuRev
- ClearCase
- CVS
- Serena Dimensions
- StarTeam
- Synergy CM
- Visual Source Safe

Java 6

Support for Java 6 and lower is deprecated. In consequence, support environments that require Java 6 or lower will be removed for future releases.

C/C++test Test Configurations

We've deprecated the following test configurations:

- Texas Instruments > Run TI CCS 3.x Application with Memory Monitoring
- Texas Instruments > Run TI CCS 3.x Tests
- ARM > Run ADS 1.2 Application with Memory Monitoring
- ARM > Run ADS 1.2 Tests
- Windows Mobile> Build Test Executable for Windows Mobile
- Windows Mobile> Build and Run Application with Memory Monitoring for Pocket PC
- Window Mobile> Build and Run Application with Memory Monitoring for Smartphone
- Window Mobile> Build and Run Test Executable for Pocket PC
- Window Mobile> Build and Run Test Executable with for Smartphone
- Window Mobile, Windows CE>Build and Run Application with Memory Monitoring for WMobile or Windows CE (ActiveSync)
- Window Mobile, Windows CE>Build and Run Test Executable for Windows Mobile or Windows CE (ActiveSync)

The deprecated test configurations are not available by default and can only be applied as user-defined test configuration. They are now shipped with C/C++test in the following location: [INSTALL_DIR]\configs\Deprecated.

Removed Support for Environments and Systems

IDEs

We've removed support for Texas Instruments Code Composer Studio 5.x.

Source Control Management Systems

We've removed support for Microsoft Team Foundation Server 2010.

Other Removed Features

- The Code Review module has been removed and is no longer supported.
- Using C/C++test in the command line with an Eclipse-Based Builder is no longer supported. The option for executing the specified build script prior to testing (`-buildscript %SCRIPT_FILE%`) has been removed.
- The following option has been removed form the Common tab in the test configuration settings: *After Testing> Commit added/modified files to source control if no tasks were reported*.
- Windows 8 is no longer supported.

Resolved Bugs and FRs

Bug /FR ID	Description
CPP-42109	cpptestcli should return non-zero exit code for set-up problems (e.g launching unit tests fails)
CPP-43406	VS Enable/Disable test case action does not refresh Test Case Explorer tree
CPP-44060	Error: expected ';' before '}' token for QCommandLineOption
CPP-44561	MISRA2004-19_10 (MISRAC2012-RULE_20_7-a) reports false positive on string concatenation in macro definition
CPP-44565	Improve auto-detection for Crosstool-NG GCC (5.x)
CPP-44578	class "std::enable_if<>" has no member "type"
CPP-44736	MSVC /permissive- should not disable friend class injection
CPP-44739	Incorrect configuration of /MP option for MSVC
CPP-44742	MISRA2004-16_8 (MISRAC2012-RULE_17_4-a) reports false positives when the 'default' statement that does not contain the 'return' is not the last label in the 'switch'
CPP-44744	GNU __underlying_type operator incorrectly handled by xharness
CPP-44760	Inconsistent use of malloc/free in runtime and cpptest_driver.c.h
CPP-44761	CppTest stream redirection API doesn't work if runtime is build with unicode.
CPP-44796	tiarm compiler configs shall pass '--define' option to the linker
CPP-44850	armclang - ignore command line with -cc1
CPP-44862	Make sure that every variant of dsStrToFloat returns same result
CPP-44865	CODSTA-190 (CERT_C-FLP37-c) reports false positives on calls to functions from string library with names other than 'memcmp' and 'bcmp'
CPP-44871	FORMAT-07 reports false positives on '=' characters used in messages of the #error directives
CPP-44872	FORMAT-21 reports false positives on '!' characters used in messages of the #error directives
CPP-44874	CODSTA-112 (MISRAC2012-RULE_18_8-a) contains 'Drawbacks' section that is no longer valid
CPP-44918	AUTOSAR-A6_5_1-a (HICPP-6_2_1-a) reports false positive on loops that do not use containers
CPP-44923	GLOBAL-UNUSEDFUNC (AUTOSAR-M0_1_10-a) reports false positive on conversion operator() defined in 'hash' struct
CPP-44946	Invalid stub call instrumentation of constructor call with braced initializer as argument
CPP-45054	Autovalidation of CPPTEST_POST_CONDITION_MEM_BUFFER doesnt work for NULL pointers
CPP-45067	PB-23 reports false positives on calls to the std::initializer_list()
CPP-45092	HICPP-6_2_1-a (AUTOSAR-A6_5_1-a) reports false positives on 'for' loops that can not be replaced by 'for-range'

CPP-45190	Fix: GCC allows nonliteral type as constexpr function parameter
CPP-45193	MISRA2004-16_10 (AUTOSAR-M0_3_2-a) rule reports false positive on for range loop
CPP-45227	Non-compilable instrumented code for double curly braces function argument
CPP-45228	Non-compilable instrumented code for std::getline
CPP-45239	HICPP-6_2_1-a (AUTOSAR-A6_5_1-a) reports false positive on the 'for' loop that uses a loop counter in the body
CPP-45240	cpptestcc instrumentation error for C++ code with generic lambda
CPP-45272	VS 'cpptestcli -fail' does not return non-zero exit codes
CPP-45283	MISRA2004-16_7 (MISRA2008-7_1_2_a) reports false positive when a parameter can not have a pointer to const type
CPP-45312	OPT-23 (JSF-122) is inconsistent with JSF-121 (getter/setter with more than 2 statements should not be inlined)
CPP-45320	CERT_C-EXP46-a does not match to the CERT EXP46-C requirement
CPP-45352	Cannot use type alias in method definition
CPP-45375	Explicit conversion operator spuriously ignored for static_cast operation
CPP-45498	C stub file and CPP testrunner file may cause linking problem (Visual Studio)
FA-7416	False positives for MISRA2012-RULE-18_1_a and MISRA2012-DIR-4_1_a (BD-PB-ARRAY)
FA-7445	BD-PB-OVERFNZT rule reported when terminating char array as 0 and '\0'
FA-7473	Cannot create CFG for a function using address of a global variable in the context of arithmetic and conditional operations.
FA-7539	BD-PB-ARRAY reports false positive out of bounds
FA-7571	C++test reports BD-PB-NOTINIT bogus violation when initialization happens inside loop.
FA-7619	BD-PB-WRAPESC potential false positive on recursive calls
FA-7626	BD-API-VALPARAM reports false violation for #include <complex>
XT-33567	Problems with 'parasoft-suppress' comment suppressions containing '//' inside suppression reason.
XT-37692	Engine license generated by IDE can be different depends on Java or IKVM version.

New Rules

Rule ID	Description
AUTOSAR-A12_8_2-a	User-defined copy and move assignment operators should use user-defined no-throw swap function
AUTOSAR-A14_5_3-a	A non-member generic operator shall only be declared in a namespace that does not contain class (struct) type, enum type or union type declarations
AUTOSAR-A8_4_4-a	Multiple output values from a function should be returned as a struct or tuple
AUTOSAR-A8_5_4-a	Avoid overloading constructors with std::initializer_list
AUTOSAR-M0_1_2-ac	Avoid conditions that always evaluate to the same value

BD-SECURITY-SENSLOG	Avoid passing sensitive data to functions that write to log files
BD-SECURITY-TDALLOC	Validate potentially tainted data before it is used to determine the size of memory allocation
CODSTA-205	Do not cast an array to the pointer to a structure of a larger size than the size of the array
CODSTA-206	The '_Noreturn' function specifier should not be used
CODSTA-207	Thestdnoreturn.hheader file should not be used
CODSTA-208	Thestdalign.hheader file shall not be used
CODSTA-209	The facilities that are specified as being provided bystdatomic.hshould not be used
CODSTA-210	The '_Thread_local' storage class specifier should not be used
CODSTA-211	The facilities that are specified as being provided bythreads.hshould not be used
CODSTA-212	The 'rsize_t' type should not be used
CODSTA-213	The '_Alignas' alignment specifier and the '_Alignof' operator should not be used
CODSTA-214	The '_Atomic' type specifier and the '_Atomic' type qualifier should not be used
CODSTA-215	The '*_STDC_WANT_LIB_EXT1_*' macro should not be defined to the value other than '0'
CODSTA-216	The '_Generic' operator should not be used
CODSTA-217	The 'errno_t' type should not be used
CODSTA-218	Do not use following macros: RSIZE_MAX, L_tmpnam_s, TMP_MAX_S
CODSTA-219	Do not use the functions defined in Annex K of ISO/IEC 9899:2011 standard
CODSTA-MCPP-47	Avoid overloading constructors with std::initializer_list
CODSTA-MCPP-48	Multiple output values from a function should be returned as a struct or tuple
CODSTA-MCPP-49	User-defined copy and move assignment operators should use user-defined no-throw swap function
CODSTA-MCPP-50	A for-loop that loops through all elements of the container and does not use its loop-counter shall not be used
CWE-532-a	Avoid passing sensitive data to functions that write to log files
HICPP-12_5_6-a	User-defined copy and move assignment operators should use user-defined no-throw swap function
MISRA2008-0_1_2_aa	A project shall not contain infeasible paths
MISRA2012-DIR-4_1_k	Avoid integer overflows
MISRA2012-DIR-4_13_f	Do not release a lock that has not been acquired
MISRA2012-RULE-1_4_a	The '_Generic' operator should not be used
MISRA2012-RULE-1_4_b	The '_Noreturn' function specifier should not be used
MISRA2012-RULE-1_4_c	Thestdnoreturn.hheader file should not be used
MISRA2012-RULE-1_4_d	The '_Atomic' type specifier and the '_Atomic' type qualifier should not be used
MISRA2012-RULE-1_4_e	The facilities that are specified as being provided bystdatomic.hshould not be used
MISRA2012-RULE-1_4_f	The '_Thread_local' storage class specifier should not be used
MISRA2012-RULE-1_4_g	The facilities that are specified as being provided bythreads.hshould not be used
MISRA2012-RULE-1_4_h	The '_Alignas' alignment specifier and the '_Alignof' operator should not be used
MISRA2012-RULE-1_4_i	Thestdalign.hheader file shall not be used

MISRA2012-RULE-1_4_j	The '*_STDC_WANT_LIB_EXT1_*' macro should not be defined to the value other than '0'
MISRA2012-RULE-1_4_k	The 'rsize_t' type should not be used
MISRA2012-RULE-1_4_l	The 'errno_t' type should not be used
MISRA2012-RULE-1_4_m	Do not use following macros: RSIZE_MAX, L_tmpnam_s, TMP_MAX_S
MISRA2012-RULE-1_4_n	Do not use the functions defined in Annex K of ISO/IEC 9899:2011 standard
MISRA2012-RULE-21_21	The library function 'system' ofstdlib.hshall not be used
MISRA2012-RULE-8_3_c	All declarations of an object or function shall have compatible types
MISRAC2012-DIR_4_13-f	Do not release a lock that has not been acquired
MISRAC2012-DIR_4_1-k	Avoid integer overflows
MISRAC2012-RULE_1_4-a	The '_Generic' operator should not be used
MISRAC2012-RULE_1_4-b	The '_Noreturn' function specifier should not be used
MISRAC2012-RULE_1_4-c	Thestdnoreturn.hheader file should not be used
MISRAC2012-RULE_1_4-d	The '_Atomic' type specifier and the '_Atomic' type qualifier should not be used
MISRAC2012-RULE_1_4-e	The facilities that are specified as being provided bystdatomic.hshould not be used
MISRAC2012-RULE_1_4-f	The '_Thread_local' storage class specifier should not be used
MISRAC2012-RULE_1_4-g	The facilities that are specified as being provided bythreads.hshould not be used
MISRAC2012-RULE_1_4-h	The '_Alignas' alignment specifier and the '_Alignof' operator should not be used
MISRAC2012-RULE_1_4-i	Thestdalign.hheader file shall not be used
MISRAC2012-RULE_1_4-j	The '*_STDC_WANT_LIB_EXT1_*' macro should not be defined to the value other than '0'
MISRAC2012-RULE_1_4-k	The 'rsize_t' type should not be used
MISRAC2012-RULE_1_4-l	The 'errno_t' type should not be used
MISRAC2012-RULE_1_4-m	Do not use following macros: RSIZE_MAX, L_tmpnam_s, TMP_MAX_S
MISRAC2012-RULE_1_4-n	Do not use the functions defined in Annex K of ISO/IEC 9899:2011 standard
MISRAC2012-RULE_21_21-a	The 'system()' function from the 'stdlib.h' or 'cstdlib' library shall not be used
MISRAC2012-RULE_8_3-c	All declarations of an object or function shall have compatible types
TEMPL-18	A non-member generic operator shall only be declared in a namespace that does not contain class (struct) type, enum type or union type declarations

Updated Rules

Rule Category	Rule IDs
AUTOSAR C++14 Coding Guidelines	AUTOSAR-A15_0_2-a, AUTOSAR-A15_1_4-a, AUTOSAR-A15_5_1-b, AUTOSAR-A18_5_2-a, AUTOSAR-A18_5_3-c, AUTOSAR-A27_0_2-b, AUTOSAR-A27_0_2-c, AUTOSAR-A2_13_4-a, AUTOSAR-A3_1_6-a, AUTOSAR-A5_0_1-f, AUTOSAR-A5_2_5-a, AUTOSAR-A5_2_5-c, AUTOSAR-A5_3_2-a, AUTOSAR-A6_5_1-a, AUTOSAR-A8_4_12-c, AUTOSAR-A8_4_13-b, AUTOSAR-A8_4_2-a, AUTOSAR-M0_3_1-b, AUTOSAR-M0_3_1-d, AUTOSAR-M0_3_1-f, AUTOSAR-M0_3_1-i, AUTOSAR-M0_3_2-a, AUTOSAR-M16_0_6-a, AUTOSAR-M16_0_8-a, AUTOSAR-M5_0_16-a, AUTOSAR-M5_0_16-b, AUTOSAR-M7_1_2-b, AUTOSAR-M9_3_3-a
Bug Detective	BD-API-NEGPARAM, BD-API-REVNEGPARAM, BD-PB-ARRAY, BD-PB-CHECKRETGEN, BD-PB-NP, BD-PB-OVERFRD, BD-PB-OVERFWR, BD-PB-PTRARR, BD-RES-BADDEALLOC, BD-RES-LEAKS
SEI CERT C	CERT_C-API01-a, CERT_C-ARR30-a, CERT_C-ARR38-a, CERT_C-ARR38-b, CERT_C-ARR39-a, CERT_C-CON30-a, CERT_C-CON40-a, CERT_C-CON43-a, CERT_C-DCL13-a, CERT_C-DCL40-a, CERT_C-ENV01-c, CERT_C-ENV30-a, CERT_C-ERR33-c, CERT_C-ERR33-d, CERT_C-EXP08-b, CERT_C-EXP10-b, CERT_C-EXP30-b, CERT_C-EXP34-a, CERT_C-EXP47-a, CERT_C-FIO22-a, CERT_C-FIO37-a, CERT_C-FIO38-a, CERT_C-FIO41-b, CERT_C-FIO42-a, CERT_C-FLP37-c, CERT_C-INT10-a, CERT_C-MEM00-e, CERT_C-MEM12-a, CERT_C-MEM31-a, CERT_C-MSC19-a, CERT_C-MSC19-b, CERT_C-MSC37-a, CERT_C-POS54-c, CERT_C-PRE01-a, CERT_C-STR05-a, CERT_C-STR30-a, CERT_C-STR31-a, CERT_C-STR31-b, CERT_C-STR38-a, CERT_C-WIN30-a
SEI CERT C++	CERT_CPP-ERR56-a, CERT_CPP-ERR57-a, CERT_CPP-EXP50-b, CERT_CPP-FIO51-a, CERT_CPP-MEM51-d, CERT_CPP-MSC52-a, CERT_CPP-STR50-c, CERT_CPP-STR50-g, CERT_CPP-STR51-a
Coding Conventions	CODSTA-166_a, CODSTA-166_b, CODSTA-185_a, CODSTA-188, CODSTA-190, CODSTA-195, CODSTA-CPP-54, CODSTA-CPP-78
Coding Conventions for Modern C++	CODSTA-MCPP-10_a, CODSTA-MCPP-35
Common Weakness Enumeration	CWE-119-a, CWE-119-d, CWE-119-e, CWE-125-a, CWE-125-c, CWE-476-a, CWE-770-a, CWE-772-a, CWE-787-a, CWE-787-d
Exceptions	EXCEPT-21
High Integrity C++	HICPP-12_5_4-a, HICPP-5_1_2-b, HICPP-5_2_1-a, HICPP-5_2_1-c, HICPP-5_3_3-c, HICPP-6_3_2-a, HICPP-9_1_1-a
Joint Strike Fighter	JSF-069, JSF-114, JSF-115, JSF-117_b, JSF-118, JSF-122, JSF-151.1, JSF-174_a, JSF-174_b, JSF-204.1_b, JSF-206
MISRA C 1998	MISRA-107_a, MISRA-107_b
MISRA C 2004	MISRA2004-12_2_b, MISRA2004-16_10, MISRA2004-16_7, MISRA2004-16_8, MISRA2004-19_10, MISRA2004-19_16, MISRA2004-20_4
MISRA C++ 2008	MISRA2008-0_3_1_a, MISRA2008-0_3_1_b, MISRA2008-0_3_1_g, MISRA2008-0_3_1_h, MISRA2008-0_3_2, MISRA2008-16_0_6, MISRA2008-16_0_8, MISRA2008-18_4_1, MISRA2008-5_0_16_a, MISRA2008-5_0_16_b, MISRA2008-5_0_1_b, MISRA2008-7_1_2_a, MISRA2008-8_4_3, MISRA2008-9_3_3
MISRA C 2012 (Legacy)	MISRA2012-DIR-4_12, MISRA2012-DIR-4_13_a, MISRA2012-DIR-4_1_a, MISRA2012-DIR-4_1_b, MISRA2012-DIR-4_1_g, MISRA2012-DIR-4_1_h, MISRA2012-DIR-4_7_b, MISRA2012-RULE-13_2_b, MISRA2012-RULE-17_4, MISRA2012-RULE-18_1_a, MISRA2012-RULE-18_1_c, MISRA2012-RULE-1_3_d, MISRA2012-RULE-1_3_e, MISRA2012-RULE-1_3_g, MISRA2012-RULE-20_13, MISRA2012-RULE-20_7, MISRA2012-RULE-21_17_b, MISRA2012-RULE-21_19_a, MISRA2012-RULE-21_3, MISRA2012-RULE-21_8_c, MISRA2012-RULE-22_1, MISRA2012-RULE-22_5_a, MISRA2012-RULE-22_5_b, MISRA2012-RULE-7_4, MISRA2012-RULE-8_13_a
MISRA C 2012	MISRAC2012-DIR_4_1-a, MISRAC2012-DIR_4_1-b, MISRAC2012-DIR_4_1-g, MISRAC2012-DIR_4_1-h, MISRAC2012-DIR_4_12-a, MISRAC2012-DIR_4_13-a, MISRAC2012-DIR_4_7-b, MISRAC2012-RULE_13_2-b, MISRAC2012-RULE_17_4-a, MISRAC2012-RULE_18_1-a, MISRAC2012-RULE_18_1-c, MISRAC2012-RULE_1_3-d, MISRAC2012-RULE_1_3-e, MISRAC2012-RULE_1_3-g, MISRAC2012-RULE_20_13-a, MISRAC2012-RULE_20_7-a, MISRAC2012-RULE_21_17-b, MISRAC2012-RULE_21_19-a, MISRAC2012-RULE_21_3-a, MISRAC2012-RULE_21_8-c, MISRAC2012-RULE_22_1-a, MISRAC2012-RULE_22_5-a, MISRAC2012-RULE_22_5-b, MISRAC2012-RULE_7_4-a, MISRAC2012-RULE_8_13-a
Optimization	OPT-23
Possible Bugs	PB-23, PB-27, PB-67, PB-72

Removed Rules

The following rules have been removed to enhance the accuracy of results:

Rule Category	Rule IDs
AUTOSAR C++14 Coding Guidelines	AUTOSAR-A27_0_2-d, AUTOSAR-A27_0_2-g
SEI CERT C	CERT_C-EXP46-a, CERT_C-MSC32-a, CERT_C-MSC32-b, CERT_C-MSC32-c, CERT_C-STR34-a, CERT_C-STR34-e, CERT_C-STR34-f
SEI CERT C++	CERT_CPP-STR50-a, CERT_CPP-STR50-d
MISRA C 2012 (Legacy)	MISRA2012-RULE-21_8_d
MISRA C 2012	MISRAC2012-RULE_21_8-d