

Offenburg (Germany) / Tampere (Finland), 1 December 2015

Please be informed that a new Testwell CTC++ version 8.0 has been released.

Testwell CTC++ v8.0 available

CTC++ v8.0 is a major upgrade from v7.3.1 (Oct 2014) --> v7.3.3 (May 2015, Windows platform only) --> v8.0 (this one).

The major enhancements in CTC++ v8.0 are:

- * Greatly renewed the HTML coverage report layout. More intuitive and easy to read color coding.
- * Introduced 'line coverage'. Based on execution flow analysis the lines executed/not executed are marked with green/red background in the HTML report.
- * Introduced annotations. They are means to comment the source code with regard to testing, and which comments show in CTC++ reports.
- * Introduced capability to report instrumented header files as their own file entities, extracted from the actual code files.
- * In ctcxmlmerge tool supporting statement coverage calculation on merged functions (when similar control flows structures).

Plus various other enhancements and bug fixes. See below the detailed CTC++ v8.0 VERSION.TXT excerpt.

Support customers can download the new version from Verifysoft download pages in the normal manner. Please contact us for further information.

Klaus Lambertz Olavi Poutanen Verifysoft Technology GmbH Testwell



Testwell CTC++ Version 8.0 - page 2

-----Excerpt of VERSION.TXT:

Testwell Oy Verifysoft Technology GmbH 27 November 2015 CTC++ System Version 8.0

This file describes the changes in successive versions of CTC++. The latest version is described first.

Version 8.0 (27 November 2015)

Since the previous "all platforms version" v7.3.1 there has been released a v7.3.3 (May 2015) version on Windows. It was a kind of test version on some new features. But those features grew and matured, and as result it came new major version CTC++ v8.0.

This version.txt explains the changes $v7.3.1 \rightarrow v8.0$. If you already have had a CTC++ v7.3.3/Windows, it already had some of the new features.

This revision has the following version numbers in its components:

Preprocessor Run-time libraries	8.0	<pre>(was: 7.3.1; seen by -h option) (was: 7.3; seen by 'ident' command applied on the library in some environments)</pre>
Postprocessor	8.0	<pre>(was: 7.3; seen by -h option and in the listings)</pre>
Header file ctc.h	8.0	(was 7.3; seen in the file)
Configuration file ctc.ini	8.0	(was: 7.3[.1]; seen in the file)
CTC++ to HTML Converter	5.1	(was: 4.1; seen by -h option)
CTC++ to Excel Converter	3.2	(was: 3.0; seen by -h option)
CTC++ XML Merger utility	3.0	(was: 2.2; seen by -h option)
ctc2dat receiver utility	3.3	(was: 3.2; seen by -h option)

and the following version numbers in its Windows platform specific components:

Visual Studio IDE Integration

4.2 (unchanged; seen by clicking the Tw-icon in the dialog program and selecting "About...")

CTC++ Wrapper for Windows 3.3 (was: 3.2; seen by -h option)

and the following version numbers in its Unix platform (Linux, Solaris, HPUX) specific components:

CTC++ Wrapper for Unix 1.4 (unchanged; seen by -h option)



<u>Testwell CTC++ Version 8.0 - page 3</u>

This CTC++ v8.0 version contains enhancements and bug fixes:

In the CTC++ preprocessor (ctc):

- New: Introduced configuration parameter WARNING_LEVEL with possible values none (like -no-warnings option), warn (normal warnings), info (some new warnings of tool limitations).
- New: introduced #pragma CTC ANNOTATION annotation_text. The annotation text comes visible to CTC++ reports (.txt, .xml, .html).
- Bug fix: When "CTC SKIP"; ... "CTC ENDSKIP"; is used to skip (not to instrument) code in a function, the starting "CTC SKIP"; is no more counted to a statement.
- Enhancement: On primary host platforms fine-tuned the default ctc.ini files to work more properly with some rarely used compiler options.
- Enhancement: For newer Borland compiler versions, taught ctc.exe to work properly with so called BOM marker, which the compiler can internally generate to a C-preprocessed file.
- New: Introduced new 'ASSEMBLY_COMMENT_CHARS=chars' configuration
 parameter. Helps in handling new kind of assembly dialects, e.g. where
 ';' does not start (assembly-)comment.
- Change: If there is 'const' in a function, e.g. in context const int xsize = X > Y ? 10 : 20; the decision in the ternary is no more instrumented. If it were instrumented, possible later use of xsize in an array declaration would cause a compilation error.
- Bug fix: Declaration in condition starting with 'constexpr' (C++11) was instrumented incorrectly resulting in a compilation error. For example,
 - if (constexpr T t = f()) ... (also in while and for). Now it is instrumented like other declarations in a condition.
- Bug fix: Under some circumstances, deriving the datafile name from symbolfile name was erroneous. E.g., if the symbolfile name was "D:\symbols\MON.sym", its datafile name came "D:\datbols\MON.dat". Now it comes correctly "D:\symbols\MON.dat".
- Bug fix: In some (build system generated) response files there could occur the following kind of file names "dirl\dir2"\file.xxx. Of these ctc.exe erroneously parsed dirl\dir2\ file.xxx. Now they are parsed to dirl\dir2\file.xxx or to "dirl\dir2\file.xxx", if there is real reason to use "...".



Testwell CTC++ Version 8.0 - page 4

- New/Enhancement: When C code, functions with old-style (K&R1) parameter declarations are now instrumented. For example:

int foo()
 int i;

{...function body;}

Previously this type of K&R1 code was forbidden, not instrumented, and sometimes could cause nondeterministic behavior.

- Bug fix: Some extreme case "..." safequarding in the ctc-generated temporary file names (e.g. if they needed to be created to a directory with space in the path).
- Bug fix: In Unixes, some extreme case corrections in handling adjacent
 '\' and ''' and '\'' characters on the command line for preserving the intended value in ctc-constructed intermediate commands. E.g. in
 -DMACRO=\'\\\'xxx\\\'\', -DMACRO=\'xyz\\\'

In step v7.3.3 --> v8.0:

- Bug fix: In a C++11 template use, something like 'using NewName = OldName<...>;' and when later seeing '...NewName<...>...', ctc parsed the code erroneously, with various error consequences.
- Change: Now each 'asm' (and its various synonyms) are counted as one statement, like a ';'. Now e.g. a function with only inline asm lines no more gets zero statement count.
- Change: If a compound statement or a function body contains only #-starting line(s), it is counted to one statement, like an empty {}.
- Bug fix: The following kind of function definitions
 'TT (fname) (parlist) {body}' were not processed right (before v7.3.3 not instrumented, in v7.3.3 wrong function name). When the return type was a simple predefined type like 'int' or when the function name was not in parenthesis, there was no problem.
- Bug fix: A template class or function that had a template parameter, something like 'template<template <class> A = B> class C {...}', was parsed erroneously causing varying consequences.

In the CTC++ run-time library:

- Functionality unchanged, but assumes a v8.0 compliant ctc.ini file.



<u>Testwell CTC++ Version 8.0 - page 5</u>

In the CTC++ postprocessor (ctcpost):

- New: For helping the ctc2html utility to generate HTML report with line coverage information, some new information is added to textual execution profile report. They are (and have meaning):
 - -- each if/for/while/do/switch/try statement block ending-} (either
 when there is or is not explicit '}' in user's code) show as
 "}+" (code after } was entered) or "}-" (code after } was not
 entered) line.
 - -- labels show now as "L:" (L was "flown to", and possibly also "gotoed") or "L:+" (L was not "flown to" but it was "gotoed") line.
 - -- case labels show now as "case n:" (this case was explicitly "jumped to") or "case n:+" (this case was not explicitly "jumped to" but it was anyway "flown to").
 - -- "default:" or "default:+" lines (same logic as in "case n:[+]"). Same information is produced to XML form report. These additional ± markings are not added, if statement coverage is not available on the function.
- New: Annotation concept introduced for commenting. In the user's code there can be line "#pragma CTC ANNOTATION text", where the "text" shows in the generated report (and later in the HTML report).
- Enhancement: Made the utility more robust against some crashes (they could occur in some pathological cases, if only parts of a function were instrumented and real control structure was not known).
- Change: In statement coverage flow analysis, if previous probe was hit, and statement list following it was started, and next probe is not hit (e.g. due to exception occurred or control remained in some called code), all statements upto next probe are counted to hit statements. Previously they were excluded from hit statements.

In step v7.3.3 --> v8.0:

- Change: Header files (when instrumented, contain complete functions) are now extracted from their code files, coverage hits summed up (if many similar), and reported as separate MONITORED HEADER FILE item. (Option -nhe, no header extract, keeps the old behavior)
- Bug fix: In XML form report the const-if() and similar decisions were reported as if the decision had always been true. But it could have been also false.
- Change: "Statement coverage not available" shows now at textual profile listing and in HTML:
 - -- function level: "N.A.", previously "100% (0 / 0)"
 - -- file level: TER% is calculated of those function from which it was available and remark "(N.A. for n functions)", previously no remark that statement coverage of some functions was not available
 - -- overall summary level: like file level (Similar changes in XML report)



<u>Testwell CTC++ Version 8.0 - page 6</u>

In the CTC++ Wrapper for Windows (ctcwrap):

- Bug fix: "ctcwrap -hard ..." now works also if the specified compiler is given with path that contains spaces.
- Enhancement: In "ctcwrap -hard ..." use the machinery now works better if CTCHOME is not set or it is not in PATH.
- Change: Some new sanity checks on command line arguments and clarified error messages.

In CTC++ to HTML converter (ctc2html):

- New: In the HTML form report introduced 'line coverage'. It shows as color-coded lines what code lines were/were not executed.
- Change: Greatly rearchitectured and improved the look-and-feel of the HTML report, notably the source code mapped Execution Profile HTML page. It has structural coverage counters as before, but the color-coding is redesigned and there is new line coverage color-coding on the source code lines.
- New: Introduced (to advanced users) capability to modify the used style sheet for fine-tuning used colors and other graphical details in the generated HTML pages.
- New: Some new --enable-XXX=... options added.
- New: --enable-XXX=0/1 options can alternatively be given as --disable-XXX/--enable-XXX on command-line.
- Change: Syntax-highlighted presentation is now default in displaying the source code (with option to disable it).

In step v7.3.3 --> v8.0:

- New: Identifying simple #if 0 ... #endif sections and in line coverage color painting showing the lines in grey (indicating disabled code)

In the CTC++ XML Merger utility:

- Change: Adapted to the new ctcpost v8.0 generated .xml reports.
- Change: When inside a function there is an #include, which brings code (that was also instrumented), the #included file is no more pulled out to a separate MONITORED HEADER FILE element in the report. Instead the #included code is reported as part of its includer function.



<u>Testwell CTC++ Version 8.0 - page 7</u>

In step v7.3.3 --> v8.0:

- Change: Statement coverage on functions is recalculated if the merged functions are similar (has same code). File and summary levels correspondingly.

In the CTC++ to Excel Converter (ctc2excel):

- Bug fix: corrected a bug in lines having FILE OVERALL column. All columns after this were shifted to the left, and FILE word was lost.

In step v7.3.3 --> v8.0:

- Change: Option -efs argument can now be given also as "c" or as plain c. Previously it had to be 'c' (char ' in some keyboards is difficult).

General:

- The v7.x level symbolfile and datafile is not compatible to CTC++ v8. The code needs to be reinstrumented and tests rerun for getting v8 level coverage report.
- CTC++ User's Guide upgraded to v8.0 level (ctcug.pdf).

Version 7.3.1 (3 October 2014)

For this version, please have a look to http://www.verifysoft.com/ctcpp731.pdf