



Offenburg (Germany) / Tampere (Finland), 18 March 2014

Please be informed that a new **Testwell CTC++ version 7.2** has been released.

This Testwell CTC++ v7.2 contains the following two major things:

- \* Introduction of test case concept: Facilitates collecting coverage data and reporting it per user-defined test cases.
- \* Introduction of ctcxmlmerge tool: Facilitates getting one combined coverage report over whole code base, which has been built and tested for different configurations. Previously, due to configuration-specific conditional compilation and macro values in the code files, CTC++ could see such files to be different, and supported coverage reporting only per one configuration at a time.

Additionally there are some bug fixes and enhancements to handle various demanding or "extreme corner" usage situations that have come up with the users.

Along with this basic CTC++ v7.2 the CTC++ Host-Target add-on (HOTA) has also been upgraded, from v5.0 to v5.1.

See below the CTC++ v7.2 VERSION.TXT excerpt for more detailed info.

As you are aware, as of 1.7.2013, the Testwell Tools, including CTC++, are now owned by Verifysoft Technology GmbH. But Testwell Oy (Finland) continues to support and develop the tools. Testwell also serves some Finland-based key customers.

Thus, if you are a customer, which Testwell serves directly, or if you still have maintenance in effect based on your purchase of it from Testwell, your download account to get this new version is still open in the same way as it used to be. If you are a Verifysoft customer, or ex-Testwell customer and renewed the maintenance via Verifysoft, you will get this upgrade with the procedures as you have in effect with Verifysoft.

Olavi Poutanen  
Managing Director  
Testwell Oy

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

Version 7.2 (13 March 2013)  
-----

This revision 7.2 of CTC++ has the following version numbers in its components:

Preprocessor	7.2 (was 7.1.2; seen by -h option)
Run-time libraries	7.2 (was 7.1; seen by 'ident' command applied on the library in some environments)
Postprocessor	7.2 (was: 7.1.1; seen by -h option and in the listings)
Header file ctc.h	7.2 (was: 7.1; seen in the file)
Configuration file ctc.ini	7.2 (was: 7.1; seen in the file)
CTC++ to HTML Converter	3.5 (was: 3.4; seen by -h option)
CTC++ to Excel Converter	3.0 (was: 2.0; seen by -h option)
CTC++ XML Merger utility	2.0 (new; seen by -h option)
ctc2dat receiver utility	3.2 (was: 3.1; seen by -h option)

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

Visual Studio IDE Integration	4.2 (was: 4.1; seen by clicking the Tw-icon in the dialog program and selecting "About...")
CTC++ Wrapper for Windows	3.1 (was: 3.0; seen by -h option)

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

CTC++ Wrapper for Unix	1.4 (was: 1.3; seen by -h option)
------------------------	-----------------------------------

This CTC++ v7.2 version contains enhancements and bug fixes:

In the CTC++ preprocessor (ctc):

- New: New configuration parameter OPT\_NO\_OPTFILE. Its argument is a list of compiler/linker options which should not be moved to a ctc-generated response file (if one needs to be generated), but be left on the generated command line. (Some compilers and linkers do not recognize certain options, if they reside in a response file.)
- Change: Speed-up when instrumenting files, which can be processed in less than one second. No more the one second delay, which was introduced in v6.5.5 for avoiding duplicate timestamps. Now the problematic situation (as it was at the time) is handled otherwise.

- Enhancement: Inline member functions in nameless classes, structs and unions are now instrumented (previously they were not), e.g.

```
class {...int f() {...}...} x;  
typedef struct {...void memb() {...}...} S;
```

The names of such functions are shown now in the listings as follows

```
<unnamed-type>::f()  
<unnamed-type>::memb()
```

- New: Removing values from a list type configuration parameter setting is now possible by the '-' marking. For example

```
OPT_ADD_COMPILE-/DCTC_HEAP_COUNTERS,/DCTC_CLIKE_MCOND
```

- Bug fix: When a string literal was immediately followed by a prefixed string literal, e.g. L"abc"L"def", this was broken by ctc (yielding non-compilable code).
- Bug fix: A ternary-?: preceded by a C++11 lambda with a throw specification was instrumented incorrectly. E.g.,  
[...] (...) throw(...) {...} () ? x : y  
Previously non-compilable code.
- Bug fix: The declarator ^, a VC++ (C++/CLI) extension, was handled erroneously in the following two cases:
  - If ^ occurred in a declaration in condition. e.g.,  
'if (T^ t = ...) ...', the instrumented code did not compile.
  - If ^ occurred in a trailing return type (C++11), e.g.,  
'auto f() -> String^ { ... }', the function (or lambda) was not instrumented.
- Change: When a datafile name is given in the DATAFILE configuration parameter, the extension ".dat" is added, if not given explicitly. (This is now consistent with adding the ".sym" extension to a symbolfile name. Note also that ctcpost accepts only such datafiles of which extension is ".dat".)
- Bug fix: ctc crashed in a situation when there were no ctc options specified on the command line and an intermediate RUN\_BEFORE\_ALL instrumentation script was called and it passed at least one option back to ctc.
- New: When C# code, statement lambdas, '(...)=>{...}' are now instrumented in the same way as C++11 lambdas.
- New: When C# code, accessors "get", "set", "add", and "remove", e.g. get {...}, are now instrumented like functions.
- Bug fix: When C# code, a ternary-?: preceded by the lambda operator '=>' was instrumented incorrectly. E.g., ... => i ? j : k ...  
Previously non-compilable code.

In the CTC++ run-time library:

- New: Introduced a test case concept. Coverage data can be collected, and later reported by `ctcpost`, per user-defined test cases. No changes to current "test case free" usage conventions.

In the CTC++ postprocessor (`ctcpost`):

- New: Information line of the number of verbose notices (e.g. if some coverage data was discarded), if any given, is now shown in the (profile, untested, timing) listing headers. This helps in support cases. (As before, the notices themselves are written to `stderr`.)
- New: Comment lines in response files are now supported. The comment character is '#' in column 1.
- Change: Option `-V` (upper-case V) used alone (or together with one or more `-c` options) shows now also the `ctcpost` version information.
- Change: Removed "SOURCE" from the file-TER line in profile reports

```
***TER  0 % (  0/  6) of SOURCE FILE file.cpp  (was before)
***TER  0 % (  0/  6) of FILE file.cpp          (is now)
```
- Enhancement: Made `ctcpost` more robust against some rare conditions (that could come up e.g. when instrumenting without C-preprocessing, although basically "unsupported", and in some weird use of `#includes`), which caused `ctcpost` to crash.
- New: Added some new elements to the XML report, in `#include` handling, primarily for the needs of the new `ctcxmmerge` utility.
- Bug fix: In XML reports, function end-brace location is now reported consistently in situations when the file was instrumented for function coverage and when the file was instrumented for higher coverage but reported in function coverage view.
- Bug fix: When Java code, statement coverage was calculated wrong in `'break label;'` under certain conditions.

In Host-Target add-on (HOTA):

- Enhancement: Upgrade v5.0 --> v5.1 released along with this basic CTC++ v7.2. Essentially only the test case support is added. The HOTA v5.0 users need not upgrade to the new v5.1 version unless the new test case features are used. The HOTA is a separate delivery package and it has its own `version.txt`.

In the ctc2dat utility:

- New: When reading text encoded counter values, and if the target has been a 64 bit machine, and if a value would come over 32 bit, max 32 bit value is used. Previously wrap-around happened. Note: when summing up the value to the datafile, the value remains at max 32 bit, if the value would become bigger (unchanged behavior).

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

- Bug fix: In "ctcwrap -hard ..." use corrected behavior if compiler/linker name was given with different casing as it actually was in the file system.
- Change: Some technical rearrangements in preparation to future needs.

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

- Change: The ctc options that are in a possible user-defined CTCOPTS environment variable are now noticed, too, to the addition of the ctc options that are given at the ctcwrap command itself.

In CTC++ to HTML converter (ctc2html):

- Enhancement: The tool accepts now also ctcxmlmerge generated profile listings.
- New: Added to advanced --enable-XXX options means to set threshold% on statement coverage TER% warnings (displayed in red color if under threshold%), e.g. --enable-stmtthreshold=100.

In Visual Studio IDE integration:

- New and changes: E.g. Visual Studio 2013 IDE integration support. Read more from the version.txt from %CTCHOME%\Vs\_integ\version.txt

In the CTC++ to Excel Converter utility:

- New: Added option -u (untested, for displaying only not fully-tested functions) and option -efs 'c' (excel field separator, for specifying the used excel field separator as wanted)
- Enhancement: Accepts now also ctcxmlmerge generated inputs.

In the CTC++ XML Merger utility:

- New: This utility, `ctcxmlmerge v2.0`, is introduced. It reads one or more XML form coverage reports, merges them and writes out a textual Execution Profile Listing. Instrumented header files are reported as separate files. This utility replaces the previous `ctcmerge` utility. This utility is now available on Windows, Linux and Solaris host platforms, but not on HP/UX.

General:

- CTC++ User's Guide upgraded to v7.2 level (`ctcug.pdf`).
- Change: On Windows platform removed installation variants Server and Workstation installation. Full Installation variant remains.
- Change: The Flexlm license control behavior is changed so that after a successful license check-out the working connection is not recorded into registry (at Windows) or to file `~/.flexlmrc`. Thus, in subsequent license check-outs the connection is established in the same way as in the first time, i.e. normally based on the environment variable `LM_LICENSE_FILE` or based on the `FLEXLM_LICENSE_FILE` setting in the CTC++ configuration file `ctc.ini`.

Version 7.1.2 (13 May 2013)

-----

For this version, please have a look to  
<http://www.verifysoft.com/ctcpp712.pdf>