



Center for Systems And
Software Engineering

UCC v.2009.10

Release Notes

1. Introduction

This document provides the release notes for the UCC v.2009.10. Unified CodeCount (UCC) is a unified and enhanced version of the CodeCount toolset. It is a code counting and differencing tool that unifies the source counting capabilities of the previous CodeCount tools and source differencing capabilities of the DiffTool (which is now replaced by UCC). It allows the user to count, compare, and collect logical differentials between two versions of the source code of a software product. The differencing capabilities allow users to count the number of added/new, deleted, modified, and unmodified logical SLOC of the current version in comparison with the previous version. With the counting capabilities, users can generate the physical, logical SLOC counts, and other sizing information such as comment and keyword counts of the target program.

This release supports both counting and differencing various languages including C/C++, C#, Java, SQL, Ada, Perl, ASP, ASP.NET, JSP, CSS, HTML, JavaScript, VB, and VbScript. This release also counts but does not diff PHP source files.

2. Compatibility Notes

Like the CodeCount toolset, UCC v.2009.10 is released in C++ source code, thus it allows users to compile and run on various platforms. The development team has tested the tool on Windows using MS Visual Studio and on Unix/Linux using the g++ compiler.

Unlike the CodeCount toolset, the UCC v.2009.10 does not support FORTRAN, Assembly, PL/1, COBOL, Pascal, and Jovial.

3. Requirements

Minimum Software Requirements:

- Compiler: a compatible C++ compiler that can load common C++ libraries including IO and STL, such as MS Visual Studio 2005, g++, and Eclipse.
- Operating systems: expectedly any platforms that can compile and run a C++ application. The tool has been tested on Windows 9x/Me/XP/Vista, Unix, Linux, Solaris, and Mac OS X.

Minimum Hardware Requirements:

- RAM: 512 MB. Recommended: 1024 MB.
- HDD: 100 MB available. Recommended: 200 MB available.

4. Features

- 1) **Counting Capabilities.** UCC allows users to measure the size information of a baseline a source program by analyzing and producing the count for:
 - logical SLOC
 - physical SLOC
 - comment
 - executable, data declaration, compiler directive SLOC
 - keywords
 - *complexity measures*: mathematic functions, logarithms, calculations, assignments, etc.
- 2) **Differencing Capabilities.** UCC allows users to compare and measure the differences between two baselines of a source program. These differences are measured in terms of the number of logical SLOC added/new, deleted, modified, and unmodified. These differencing results are saved to both plain text .txt and .csv files.
- 3) **Counting and Differencing Directories.** UCC allows users to count or compare source files by specifying the directories where the files are located. This capability eliminates difficulties in creating the file list that users may have encountered in the previous versions of the CodeCount toolset.
- 4) **Various Programming Languages Supported.** The counting and differencing capabilities accept the source code written in C/C++, C#, Java, SQL, Ada, Perl, ASP, ASP.NET, JSP, CSS, HTML, JavaScript, VB, and VbScript. In addition, the tool can also count (but not compare) PHP source files. The tool detects the language of each file using its file extension (see Feature #9)
- 5) **Command Arguments.** The environment file containing user's settings (e.g., c_env.dat file) in the CodeCount tools is no longer used. Instead, the tool accepts user's settings via command arguments.
- 6) **Duplication.** For each baseline, two files are considered duplicates if they have same content or the difference is smaller than the threshold given through the command line switch *-tdup*. As such, two files may be duplicated although they have different filenames.

Duplicates in each baseline are counted, and their counting results are saved into files named "Duplicates-A-<LANG>_outfile.dat" and "Duplicates-B-<LANG>_outfile.dat", where LANG is the name of the programming language used. As such, one or more files are generated as a result of the duplication feature.

The output file named DuplicatePairs.txt shows the matching pairs and a list of duplicate files for each baseline. The matching pairs are displayed in two columns, Baseline A and Baseline B. The duplicate files are essentially the files that are

counted and shown in the files “Duplicates-A-<LANG>_outfile.dat” and “Duplicate-B-<LANG>_outfile.dat”.

- 7) **Matching.** Two files are matched if and only if they have the same filename regardless of which directories they belong to. Two files that have the same filename are matched if they have least uncommon characters in their directory names. This feature allows users to handle to the situation where files are moved from one directory to another or the directory structure is changed.
- 8) **Complexity Count.** When the counting capabilities are executed (the command without the `-d` switch), the tool produces complexity counts for C/C++, C#, Java, Ada, Perl, and SQL files. The complexity counts include the number of math, trig, logarithm functions, calculations, conditionals, logicals, preprocessors, assignments, and pointers. Complexity results are saved to the file “outfile_cplx.txt”.
- 9) **File Extensions.** The tool determines the language used in a source file using file extension. This release supports the following languages and file extensions:

Languages	File Extensions
C/C++	.cpp, .c, .h, .hpp, .h, .cc, .hh
C#	.cs
Java	.java
SQL	.sql
Ada	.ada, .a, .adb, .ads
Perl	.pl, .pm
ASP, ASP.NET	.asp, .aspx
JSP	.jsp
CSS	.css
HTML	.htm, .html, .shtml, .stm, .sht, .oth, .xhtml
PhP	.php
JavaScript	.js
VB	.vb, .frm, .mod, .cls, .bas
VbScript	.vbs

5. Changes and Upgrades

This section describes changes and upgrades to the tool since Release 2009.01.

- 1) Newly added languages include ASP, ASP.NET, JSP, CSS, HTML, VB, and JavaScript.
- 2) The number of characters allowed in each physical is 10,485,760, which was changed from 1,024.
- 3) Rational ClearCase files supported: The Rational ClearCase application appends version information to the filename, starting from ‘@@’. The command line switch

-cf allows the UCC tool handle the original filename instead of the ClearCase-modified filename.

- 4) Improved the accuracy of the counting and differencing functions for Ada and SQL.
- 5) For web files, changed the language name from {xxx in yyy} to {xxx/yyy} where xxx is the name of the language embedded in the source file of language yyy. For example, “Java in JSP” is changed to “Java/JSP”.
- 6) Upgraded the counting and differencing functions for Perl to handle the ‘#’ character in regular expressions and arrays.
- 7) Improved the performance in processing excessively long logical SLOC. If a logical SLOC has the number of characters larger than a specified threshold (default to 10,000), the additional characters are truncated. The user can specify this threshold by using the command line switch *-trunc*.
- 8) The command line switch *-trunc* specifies the threshold percentage for duplicated files of the same name. This specifies the maximum percent difference between two files of the same name in a baseline to be considered duplicates. By default, this threshold is zero, so the files must be identical by logical SLOC – spacing and comments are not considered.
- 9) Added physical to logical line ratio and number of files accessed into the output files.
- 10) The counts and complexity of duplicated files are printed separately for Baseline A and Baseline B.
- 11) Differencing results for duplicated files that are unmatched across baselines are printed in a separate file.
- 12) An error log file is generated to report all errors occurred.
- 13) A progress indicator was implemented to indicate the percentage of completion.

As a result of defect fixes and upgrades, the logical SLOC count produced by UCC is generally **1% ± 0.2%** less than the logical SLOC count produced by the CodeCount toolset. This value was determined by performing an experiment on open source applications that range from 40 KSLOC to 60 KSLOC.

6. Known Issues and Limitations

No	Issues
1	For JavaScript code, the tool does not count the statement that is not terminated by a semicolon.
2	<p>When the command option <i>-dir</i> is provided along with a file spec and there is no file found for some of the types specified by the file spec, the tool shows a file-not-found message. This does not affect the final result.</p> <p>For example, if the following command is run</p> <pre>UCC -dir .\proj1 *.cpp *.h</pre> <p>and if the directory <i>proj1</i> contains some .cpp files but it does not have any .h files, the tool shows a file-not-found message.</p>
3	The tool only detects and handles C# and VB as code-behind languages for the ASP.NET.