The NCBI C++ Toolkit

Release Notes (November 22, 2004)

Abstract

- Download Location
- Source Archive Contents
  - Source Code Archives
- New Development
  - CORELIB -- Portability and Application Framework
  - CONNECT -- Data streaming, Networking, and Dispatching
  - UTIL -- Miscellaneous Low-Level APIs
  - SERIAL -- Data Serialization (ASN.1, XML)
  - DATATOOL -- Code Generator and Data Converter Tool
  - CGI -- CGI and Fast-CGI Application Framework
  - HTML -- HTML Generation Library
  - Berkeley DB API (bdb) -- Much Enriched C++ API Based On BerkeleyDB
  - DBAPI -- Generic SQL Database Connectivity
  - ALGO/ALIGN -- Generic Alignment Algorithms
  - BLAST
  - ALNMGR -- Bio-sequence Alignment Manager
  - BIO-OBJECTS -- Bio-Object Specific Utility Functions (Not Involving OM++)
  - OM++ -- Object Manager -- For Retrieving and Processing Bio-Objects
  - OM++ LOADERS/READERS -- Data Retrieval Libraries for OM++
  - ID2
  - BIO-TOOLS
  - ALGO/PHY_TREE -- Phylogenetic and bio tree algorithms
  - BUILD FRAMEWORK (UNIX)
  - PTB -- Project Tree Builder for MSVC++ .NET
  - 3RD-PARTY PACKAGES
- APPLICATIONS
- Documentation
  - Document Location
  - Document Content
- Building on the MacOS
- Platforms (OS's, compilers used inside NCBI)
  - Unix
  - MS Windows
  - Mac OS X
- Caveats and Hints

- MacOS 10.X / CodeWarrior 9.1
- MacOS 10.2/GCC 3.3
- GCC 2.95
- GCC 3.0.4
- GCC 3.3
- GCC 3.4

• Last Updated

Release Notes (November 22, 2004)

Download Location

ftp://ftp.ncbi.nih.gov/toolbox/ncbi_tools++/2004/Nov_30_2004/

Source Archive Contents

Source Code Archives

- ncbi_cxx_unix--Nov_30_2004.tar.gz-- for UNIX'es (see the list of UNIX flavors below) and MacOSX/GCC
- ncbi_cxx_unix--Nov_30_2004.gtar.gz-- for UNIX'es (see the list of UNIX flavors below) and MacOSX/GCC
- ncbi_cxx_win--Nov_30_2004.exe-- for MS-Windows / MSVC++ 7.1 (self-extracting)
- ncbi_cxx_win--Nov_30_2004.zip-- for MS-Windows / MSVC++ 7.1
- ncbi_cxx_mac_cw--Nov_30_2004.tgz-- for MacOS 10.3.4 / CodeWarrior DevStudio for MacOS 9.2
- ncbi_cxx_mac_xcode--Nov_30_2004.tgz-- for MacOS 10.3.4 / xCode 1.[1-5]
- ncbi_cxx_mac_gcc--Nov_30_2004.tar.gz-- for MacOS 10.2, 10.3.4 / GCC 3.3.2

The sources correspond to the NCBI production tree sources from patch "RELEASE_NOV_2004", which in turn corresponds to the development tree sources from October 26-29, 2004.

New Development

Some of the new development that may introduce potentially backward-incompatible changes are marked with an asterisk(*).

CORELIB -- Portability and Application Framework

1. CNcbiApplication -- allow to treat standard I/O as binary on MS Windows. CArg_***File -- do set standard I/O streams to "binary" mode if so specified by the argument's description.
2. CNcbiApplication -- implemented setting up logfile from configuration file. Gave access to the logfile name to allow switching of diagnostics handlers.
3. CNcbiApplication -- changed standard exit codes to positive values. Exit codes must be in the range 0..255.
4. NCBI_GetRunpath() -- added implementation for MS-Windows.
5. StringToUInt8_DataSize() -- new function to converts data size constants (100MB, 256KB) to integer.

6 CTime -- added new format letters support:'d' - day without leading zero, 'l' - milliseconds, 'r' - microseconds, 'p' and 'P' - AM/PM.

7 CTimeSpan -- new string conversion function AsSmartString().

8 (*) CTimeSpan -- renamed GetTotal*() -> GetComplete*().

9 CStopWatch -- added operator << to dump current stopwatch time to an output stream.

10 CDirEntry -- always delete trailing path separator except some special cases, like root dir and DOS disk names. Fixed NormalizePath() to prevent crashing on some path strings and corrected for the processing of root dirs variations.

11 Added ERound and ESign enumerators to use instead of senseless booleans.

12 (*) Case-insensitive comparisons now always treat letters on lowercase, in keeping with most (all?) standard library implementations.

13 CDiagCompileInfo -- new class to collect compile time information (__FILE__LINE__ and MODULE_NAME at this time). DIAG_COMPILE_INFO -- a convenience macro to make an instance of CDiagCompileInfo. CNcbiDiag::CNcbiDiag*() and CException::CException*() now accept CDiagCompileInfo instead of __FILE__ and __LINE__.

14 Added functionality to set/get module, class and function name in CNcbiDiag and CException.

15 SetDiagFilter() -- new function to setup diagnostic messages filtering based on source file path and/or module, class and function name

16 CNcbiApplication -- understands [DIAG].*_FILTER entries in config file to specify the diagnostic filtering rules.

17 CConfig -- new class to conveniently access a hierarchical tree of configuration parameters.

CONNECT -- Data streaming, Networking, and Dispatching

1 Added client API for the NCBI NetCache server.

2 ncbi_buffer.[ch] -- new methods BUF_Erase(), BUF_Append(), and BUF_Prepend (). Append/Prepend allow to insert a chunk of data into the buffer without making a copy of the data.

3 CConn_Streambuf -- added tracking of input position, for std::istream::tellp().

4 CONNECT_Init() -- now can be set up to be called automatically whenever a CONN-based C++ stream is constructed.

5 MEMORY_CreateConnectorEx(), CConn_MemoryStream -- new constructors, to allow building the connector on top of a BUF for the cases when an arbitrary data area is to be converted to an input/output stream in-place.

6 CConn_MemoryStream::ToString() -- added.

7 DisableOSSendDelay() -- new method for CSocket and SOCK API, to allow turning off the Nagle algorithm.

UTIL -- Miscellaneous Low-Level APIs

1 CStdPoolOfThreads -- fixed potential race conditions.

SERIAL -- Data Serialization (ASN.1, XML)

1 Corrected writing namespace name of NamedType in XML output stream.

2 RPC clients (for ID1, Entrez2, etc.) propagate persistent low-level exceptions.

3 Fixed the "delayed" object parsing. Allow to disable delayed parsing via registry or environment.

4 Provided thread-safe initialization in s_SerFlags().

DATATOOL -- Code Generator and Data Converter Tool

1 Improved diagnostics in datatool when parsing ASN.1 spec that contains an identifier with a wrong name (DATATOOL crashed formerly with no meaningful message).

CGI -- CGI and Fast-CGI Application Framework

1 Improve handling of unopenable FastCGI watch files.

2 Ensure that FastCGI IPC file descriptors don't get propagated to child processes, which could do no good with them.

3 CCgiApplication -- standard I/O streams to work in binary mode, to override MS-Windows default that opens them in text mode.

4 CGI API's exceptions -- re-structurize to distinguish between user- and server- errors by the exception type.

HTML -- HTML Generation Library

1 CHTMLText -- allow to disable internal buffering, at the cost of losing some functionality related to the recursive tag mapping; "disabled" is the default now.

2 Modified StripTags() to strip mapping tags <@...@> before stripping HTML tags. Added possibility to remove single closing tags.

Berkeley DB API (bdb) -- Much Enriched C++ API Based On BerkeleyDB

1 CBDB_Cache - added option to run a maintenance thread to do garbage collection (remove obsolete cache elements) and to cleanup old unused transaction log files. Naturally, it is available only in multi-thread applications.

2 CBDB_Cache - implemented flexible (traffic based) transaction checkpoint. Improved database stability.

3 Fixed bug in the IWriter implementation provided by CBDB_Cache.

4 CBDB_Cache - added new class factory configuration options to control the use of transactions and the level of I/O syncronicity.

5 CBDB_Env::SetLogFileMax() -- new method, to limit the size of log files.

DBAPI -- Generic SQL Database Connectivity

1 By default, rename DBLIB symbols in built-in FreeTDS in order to avoid a potential name clash with Sybase DBLIB (when linking the two statically together).

2 Fixed bug in binding NULL-valued CDB_Char, CDB_VarChar, CDB_Binary and CDB_VarBinary.

3 CVariant - Added SetNull() method.

ALGO/ALIGN -- Generic Alignment Algorithms

New general purpose tree algorithms:

1 CBandAligner -- a new class, derived from CNWAligner to provide a version of the band-limited global alignment algorithm. The applicability of the algorithm is limited

to a special but important case when the upper band limit can be given prior to running the algorithm. The running time and space are then both have an order of band multiplied by the sequence length.

2 A strategy of cutting short terminal exons in splign has been adjusted.

3 CCompartmentFinder -- a small intron length penalty was introduced to favor more compact models over more stretched ones.

BLAST

1 (*) Removal of unneeded functionality from the BlastSeqSrc API and addition of the IBlastSeqInfoSrc interface.

2 Bug fixes to gapped blastx/tblastn and ungapped blastx/tblastn for sum statistics.

3 (*) Updated code to use new version of scoremat.asn specification

4 Added structure group customization to PSSM engine to ignore query sequence. Improved input data validation and error handling in PSSM engine.

5 Made consistent the sorting of HSPs on entrance and exit of all public functions after the preliminary gapped alignment. Also, HSPs are sorted by score (not by e-value) after traceback.

6 Moved RPS blast initialization into the BLAST engine instead of forcing applications to manually do it, added support for concatenated queries to RPS blast.

ALNMGR -- Bio-sequence Alignment Manager

1 CAlnMap:

- Fixed the support for unaligned regions in GetRawSeg.
- Fixed the range of the Unaligned chunk in case it spans over multiple gaps.
- Extended Get{Aln,Seq}Chunks() with the ability to obtain [implicit] unaligned regions.
- Completed AlnRange swapping.
- Flags for the new functionality -- fUnaligned and fAddUnalignedChunks.

2 CAlnVec::GetColumnVector() -- added argument check.
CAlnVec::GetAlnSeqString() -- added translation.

3 CAlnMix -- the use (or not) of OM++ now only depends on whether a scope was provided at construction time.

4 CAlnMix::fPreserveRows -- new flag.

5 CAlnVwr -- PopsetStyle flags changed.

BIO-OBJECTS -- Bio-Object Specific Utility Functions (Not Involving OM++)

1 CSeq_id -- support several new prefixes in IdentifyAccession() method.

2 Splitted id2 and seqsplit libraries

3 CSeq_loc_CI -- added IsSetStrand().

4 CSeq_loc -- added ChangeToMix(), ChangeToPackedInt().

5 CDense_seg::FillUnaligned() -- new method, to create a dense-seg with all unaligned pieces (implicit inserts), if any, added between segments.

6 CDense_seg::FromTranscript() -- new method, to facilitate initialization from the alignment transcript string.

7 Auth-list.names marked for delayed parsing to reduce memory usage by entries with huge publication sets.

8 CSeq_id_Handle -- new methods GetMapper() and MatchesTo(), fixed the matching of Genbank, Embl, and Ddbj Seq-ids.

9 CSeq_id_Mapper::GetInstance() - made thread-safe.

OM++ -- Object Manager -- For Retrieving and Processing Bio-Objects

1 Added mapping of code-breaks and anticodons in feature iterator.

2 Preserve fuzz from the original location or use it to indicate truncated intervals in annotation iterators.

3 Added flag for handling unresolved IDs (search/ignore/fail) and method for external annotations search in SAnnotSelector.

4 Fixed behaviour of CSeq_loc_Mapper to match results produced by iterators.

5 Added new constructors to annotation iterators, marked many constructors and methods as deprecated.

6 Implemented CScope::AddSeq_annot().

7 Fixed processing of 'partial' flag in mapped features.

8 Features truncated while mapping now get correct Int-fuzz values to mark truncation place.

9 Fixed thread safety in several places: CAnnot_Index::x_InitIndexTables(), CObjectManager::GetInstance().

10 CSeqMap::End() -- fix to allow reverse iteration from the end.

11 CSeqVector -- use 0xff to represent gaps in NCBI2na encoding rather than throw an exception.

12 Added methods for working with gaps in CSeqVector: IsInGap(), SkipGap(), SkipGapBackward(), GetGapSizeForward(), GetGapSizeBackward().

OM++ LOADERS/READERS -- Data Retrieval Libraries for OM++

1 Added support for "orphan" annotations - for annotations-only data loaders.

2 (*) One of overloaded GetRecords() methods was renamed to GetDetailedRecords() to avoid name conflicts.

3 Fixed default implementation of GetIds().

4 Allow the caching ID1 reader to cache intermediate gi.

5 Try to deal with withdrawn and private blobs without exceptions in GenBank loader to reduce problems with old GCC.

6 Added support for SNP_graph annotations.

ID2

1 ID2-Reply-Get-Blob-Id.blob-id -- made optional to report errors.

2 (*) SeqSplit specs -- modified to allow splitting non-gi sequences. WARNING: If you use split_cache application you'll have to resplit your data in the cache since SeqSplit specification was changed.

3 ID2S locations and Seq-id selections made consistent and orthogonal.

4 Added support for loading split descriptors in CSeqdesc_CI.

5 Added support for split assembly history.

6 Updated splitter library for new SeqSplit specs.

BIO-TOOLS

1 sequence::GetTitle -- when generating protein titles, include extra protein names only when the new flag fGetTitle_AllProteins is given.

2 sequence::GetTitle -- pick gene names more reliably in the case of overlaps.

3 seq_loc_util.[ch]pp -- added seq-loc operations (Add/Subtract/Merge).

4 TestForOverlap() -- added processing of multi-strand locations.

5 TestForOverlap() -- to perform all calculations with Int8 and check for overflow when returning int. x_TestForOverlap() allows to get the Int8 value.

ALGO/PHY_TREE -- Phylogenetic and bio tree algorithms

CBioTree- New node design, improved usability.

BUILD FRAMEWORK (UNIX)

1 Configure now supports an explicit --without-64 option for use on bi-arch systems that produce 64-bit binaries by default.

2 The Toolkit now automatically regenerates makefiles as needed, so it should generally no longer be necessary to run reconfigure.sh after editing or even adding the configurable ("*.in") makefiles.

PTB -- Project Tree Builder for MSVC++ .NET

1 PTB to rely more on UNIX makefile tree when generating MSVC projects: parse the contents of Makefile.in and process requested projects and subfolders only.

2 Take into account LIB_OR_DLL flag in makefiles: if a specific project is marked as DLL, then do not build it as static library in the DLL builds.

3 Made it possible to specify which header files go into a specific MSVC project -- the headers can now be listed in the "HeadersInInclude" and "HeadersInSrc" entries of section "[AddToProject]" in Makefile.*.msvc.

4 Added dependency on DATATOOL into ASN projects, so they are forced to re-built when DATATOOL changes.

5 Process configurable sources (*.in) a la UNIX "configure" utility.

6 Tune the projects and defines for each build configuration individually and independently of other configurations (so if you have some 3rd-party lib available in only some configurations it will be used in those).

3RD-PARTY PACKAGES

1 Prepare an easy-to-build bundle of 3rd-party packages (in source codes) for MSVC++, put it to a public FTP along with the NCBI C++ Toolkit sources.

APPLICATIONS

1 NetCache -- new server-side daemon to share temporary data between several hosts. Mostly to be used for keeping CGI session context and immediately reusable (temporary yet expensive to re-calculate) data.

2. AlnVwr (a demo of various alignment viewers to illustrate the usage of CAlnMap and CAlnVec). Added Seq-align as input type; its segs must be of type Dense-seg. Implemented a demo view using the new CAlnPos_CI class.
3. AlnMrg (a demo program to illustrate the usage of CAlnMix). The default output now is Seq-align, but Dense-seg can be chosen by the "-dsout" command-line argument. Added viewers, mostly to be able to view translated Dense_segs (DS + m_Widths).
4. objmgr_demo.
   - **a** Added more options to control demo application: -range_loc -overlap -by_product -desc_type -count_subtypes
   - **b** Added output of CSeq_loc_Mapper result.
   - **c** Added bidirectional tests for CSeqMap_CI.
   - **d** BDB cache options tuned for better performance.
5. seqvec_bench. Test for "block" fetching with GetSeqData().
6. test_objmgr_data[_mt]. Allow showing the processing time in verbose mode.
7. id1_fetch_simple. Allow to change ID1 service name via command line arguments.

Documentation

Document Location

The documentation is available online at http://www.ncbi.nlm.nih.gov/books/bv.fcgi?call=bv.View..ShowTOC&rid=toolkit.TOC&depth=2 as a book titled "The NCBI C++ Toolkit". This is an online searchable book.

The C++ Toolkit book also provides PDF version of the chapters; although these are not up to date in this release. The PDF version can be accessed by a link that appears on each page.

The older HTML documentation has been deprecated and is no longer being updated, and "The NCBI C++ Toolkit" online book at the previously listed URLs is the official documentation.

Document Content

Documentation has been grouped into chapters and sections that provide a more logical coherence and flow. New sections and paragraphs continue to be added to update and clarify the older documentation or provide new documentation. The chapter titled "Introduction to the C++ Toolkit" gives an overview of the C++ Toolkit. This chapter contains links to other chapters containing more details on a specific topic and is a good starting point for the new comer.

The DOXYGEN source browser is used to complement the traditional docs with structured DOXYGEN-generated "Library Reference" ones based on the in-source comments. You can access the DOXYGEN source browser for the NCBI code from:

http://www.ncbi.nlm.nih.gov/IEB/ToolBox/CPP_DOC/doxyhtml/

The above link is also available under the "Browsers" that appears on each page.

You can also access the CVS code repository via a Web interface. These links also appear in the sidebar box on each page.

A C/C++ Symbol Search query appears on each page of the online Toolkit documentation. You can use this to perform a symbol search on the public or in-house versions of LXR, Doxygen and Library. The Library search runs a CGI script that lists the library name where the symbol (such as function) is defined in.

The current release notes as well as past release notes are now in an appendix in the C++ Toolkit Book and a link to the current release notes appears on each page of the online Toolkit document.

Building on the MacOS

We now build all the libraries and most of the applications including the Genome Workbench (gbench). We do not build any of the many test or demo apps (except testvalidator). We also build the FLTK library's GUI editor, fluid.

All apps are built as application bundles except gbench_plugin_scan and datatool which are built as command line apps. Any of the applications can be built as command line apps by tweaking the build scripts or the Codewarrior projects.

Build procedure is still the same: use an AppleScript editor to open and run the script files makeLibs.met and makeApps.met. You must use a script editor capable of opening a script file larger than 32K, such as Apple's Script Editor v2.0, or Smile. Script Editor v1.9 will not work since makeLibs.met just got too big. The command line tool osascript also works.

Projects include targets to compile with BSD/Apple headers and libraries and with MSL headers and libraries, but plugins (for gbench) built with MSL do not link properly, and so, because of lack of interest to keep up with changes, some source code does not currently compile with MSL. Hopefully this will become easier to work with and get fixed with Codewarrior v.9.

Targets to be compiled can be controlled by including an empty file or folder in the compilers:mac_prj folder with the name 'Build' followed by the keywords of the targets you want built. The keywords are: MSL, BSD, Debug and Final. For example, to build only the BSD Debug targets use: "Build BSD Debug", to build both BSD debug and release (final) versions: "Build BSD". The default is to build everything.

If you install the C++ toolkit under a different name than "ncbi_cxx" or in a different location than your home directory, you can edit the script's properties, pRootFolderName and pRootFolderPath, to override these defaults. Note: these paths, and those mentioned below, must be entered in mac format (e.g. disk:Users:username:) not Unix format (e.g. /Users/username/). The disk name (and its following colon) may be omitted.

Certain third party libraries (see Table 18) are required to build some parts of the C++ toolkit. The scripts will try and find them if they are in your home directory, or you can specify where they were installed using properties at the beginning of the script.

Table 18

Third Party Libraries

Library	Property	Example
FLTK 1.1.5rc2 (w/ NCBI patches)	pFLTKRootFolder	"usr:local:src:fltk-1.1.5rc2:"
BerkeleyDB 4.2.52	pBdbRootFolder	"Users:myhome:mylibs:db-4.2.52"
SQLite 2.8.15	gSqliteFolder	
dlcompat	pDLRootFolder	"usr:"

You do not have to build the FLTK or BDB libraries separately. This is done by the scripts and Codewarrior along with the toolkit libraries. Just unpack the source bundles in your home directory or where ever you have specified in the appropriate properties. The root folders for FLTK and BDB do not have to have any particular names. If there is more than one version the scripts will grab whichever is last alphabetically (e.g. fltk-1.1.4r2 will get used instead of fltk-1.1.3).

The scripts normally halt on any Codewarrior compilation errors. If you want them to continue and save errors, set the next script property, pSaveContinueOnErrors, to true. Compilation errors for a project will be saved in a file in the same folder as the project being built, with a name in the following format: projectName-targetNumber.errs (e.g. xncbi-2.errs).

The Genome Workbench's configuration file(s) is stored in the users Library:Application Support:gbench folder.

CFM builds are not supported. OS 8 or 9 are not supported. We know 10.2 works. We think 10.1 still works, and 10.3 might work.

Platforms (OS's, compilers used inside NCBI)

This release was successfully tested on the following platforms -- but may also work on other platforms. Since the previous release, some platforms were dropped from this list, just because we do not use them here anymore, and some were added (these new platforms are highlighted using bold font). Also, it can happen that some projects would not work (or even compile) in the absence of 3rd-party packages, or with older or newer versions of such packages -- in these cases, just skipping such projects (e.g. using flag "-k" for make on UNIX), can get you through.

Unix

Table 19

Unix OS's and Supported Compilers

Operating System	Architecture	Compilers
Linux-2.4.23 (w/ LIBC 2.2.5)	INTEL	GCC 2.95.3, 3.0.4, 3.4.0
Linux-2.4.26 (w/ LIBC 2.3.2)	INTEL	GCC 3.4.0
Linux-2.6.7 (w/ LIBC 2.3.3)	INTEL/64	GCC 3.4.0
Linux-2.4.23 (w/ LIBC 2.2.5)	INTEL	ICC 8.0
Solaris-8	SPARC	WorkShop 6 update 2 C++ 5.3 Patch 111685-13 (64-bit, 32-bit)
Solaris-8	SPARC	GCC 3.0.4
Solaris-9	INTEL	WorkShop 6 update 2 C++ 5.3 Patch 111685-13
Solaris-9	INTEL	GCC 3.3.3
IRIX64-6.5	SGI-Mips	MIPSpro 7.3.1.2m (64-bit, 32-bit)
FreeBSD-4.5	INTEL	GCC 3.0.4
FreeBSD-4.10	INTEL	GCC 3.4.2
Tru64 (OSF1) V5.1	ALPHA	GCC 3.3.2
Tru64 (OSF1) V5.1	ALPHA	Compaq C++ V6.5-014

MS Windows

Table 20

MS Windows and Supported Compilers

Operating System	Compilers
MS Windows	MSVC++ 6.0 Service Pack 5. No longer supported.
MS Windows	MSVC++ 7.1. See documentation on MS Visual C++.NET.

Mac OS X

Table 21

Mac OS, and Supported Compilers

Operating System	Compilers
MacOS 10.2, 10.3.4	GCC 3.3
MacOS 10.3.4	CodeWarrior 9.2

Caveats and Hints

MacOS 10.X / CodeWarrior 9.1

1. Not all of the test or demo applications are built.
2. The source code for the latest release of FLTK (1.1.x), BerkeleyDB (4.x) and SQLite (2.x) should be present. See the installation instructions for details.

MacOS 10.2/GCC 3.3

At least the GCC 3.3 update for Dec. 2002 Developers Tools required from Apple.

GCC 2.95

1. Poor MT-safety record.
2. Relatively incomplete/incorrect (comparing to modern compilers) STL implementation.
3. It is going to be deprecated in NCBI rather soon -- as soon as we have any significant trouble with its maintenance.

GCC 3.0.4

1. Destructor of constructed class member is not called when exception is thrown from a method called from class constructor body (fixed in GCC 3.3).
2. STL stream uses locale in thread unsafe way which may result to segmentation fault when run in multithread mode (fixed in GCC 3.3).
3. Long-file support for C++ streams is disabled/broken (first broken in 3.0, fixed in 3.4).

GCC 3.3

Other than the feature described below, GCC 3.3.2 has been very good for us; it has a lot of very ugly bugs finally fixed.

Painfully slow linking in debug mode on Linux with GCC-3.3 compiler:

1 Starting with binutils 2.12 linker tries to merge constant/debug strings marked for merging in object files. But it seems it does this job very inefficiently - I've seen messages about it in internet.

GCC starting with version 3.2 marks section of string constants ready for merging, and also has an option to disable this flag in object files (-fno-merge-constants). Adding this flag to compilation stage allows to avoid slow linking.

GCC 3.3 also sets merge flag for debug sections and unfortunately there is no option to disable this flag. As a result, linking of debug executables significatly slower than with gcc 3.0.4.

The slowdown rate depends on size of debug strings section and it's non-linear, so bigger projects will suffer more of this bug (N^2).

Binutils 2.15 fixes this. The link time still 2 times slower than without symbol merge, but the resultant executable is about two times smaller in size, and no compiler patching is necessary. We are still testing it in-house.

We had to patch GCC 3.3 in-house with the fix described at http://lists.boost.org/MailArchives/boost/msg53004.php.

2 Long-file support still broken (first broken in 3.1).

GCC 3.4

1 The "Painfully slow linking..." (see GCC3.3 [1] above) is still an issue. -- Again, had to patch it in-house to speed it up, a la GCC 3.3.

2 At least on Linux, ifstream::readsome() does not always work for large files, as it calls an ioctl that doesn't work properly for large files.

3 At least on Linux, GCC 3.4.[1,2] optimizer (very rarely) generates incorrect code when comparing enumerated values in else-ifs. (Fixed in 3.4.2)

Last Updated

This section last updated on November 24, 2004