

# Building Windows Version of OpenOffice.org on Linux

Madness, or The Way?

**Jan Holesovsky**

<kendy@suse.cz>

**Novell**<sup>®</sup>

# Content

- Motivation
- Possible speedups of the compilation
- The cross-compilation
- Did it work?

# Motivation

- On Linux, we can build OOo in ~30 minutes!
  - Yes, from scratch
  - Used to be ~25, but regressed recently thanks to changes in helpcontent2 :-(
  - On Win32 it is much more
- Windows builds are necessary
  - Lots of users...
- Developers don't need all the stuff
  - The packagers might differ in some needs though

# The 30 Minutes Linux Build

- Big, fat machine
  - 8 cores
    - > Allows heavy parallelism
  - 8G memory
    - > The build nearly does not have to touch disk ;-)
- Building less stuff
  - `--with-system-*`
- Tools for faster compilation
  - icecream
    - > `instsetoo_native$ build -all -P8 -- -P10`
  - Alternatively `ccache`

How To Apply Those To Win32?

# Big, Fat Machine

- More memory
  - Memory is cheap these days
  - But check that your Win32 edition can use it ;-)
    - > <http://msdn.microsoft.com/en-us/library/aa366778.aspx>
- Cygwin eats some of the performance
  - Emulated POSIX

# Building Less Stuff

- Divide et Impera!
  - Split build
    - > It's not necessary to build everything over and over again if you want to hack on eg. Writer
    - > openSUSE 11.1 uses that
  - Own release cycle for the 3<sup>rd</sup> party stuff
    - > Updated sporadically, but has to be built over and over again
    - > “solver” used to help a bit, but:
      - » Huge download size
      - » Incompatible among compilers
  - Own release cycle for helper tools
    - > dmake, transex3, makedepend, registry-related, ...
    - > Would help cross-compilation as well

# The Tools: Precompiled Headers

- Feature of the compilers
  - MSVC has it
    - > In OOo: --enable-pch given to ./configure
    - > Gcc has it as well
  - Header files are compiled to an interim format
  - For optimal use, all the headers have to be collected in one big header file; in OOo it is “precompiled\_<module>.hxx”
  - [http://wiki.services.openoffice.org/wiki/Precompiled\\_header\\_-\\_PCH](http://wiki.services.openoffice.org/wiki/Precompiled_header_-_PCH)
  - Build of Writer: 43 minutes



# The Tools: ccache

- Caches output of the previous compilations
  - Makes the 1<sup>st</sup> build a bit slower, next one is faster
  - The c/c++ file is preprocessed, the content of the file + the command line is hashed, and
    - > If the hash already exists in the cache, take the resulting .o, stdout and stderr and use it as if it was compiled
    - > If it does not, pass it to the compiler, write .o, stdout and stderr to the cache, and write the output as if the compiler itself did it
  - Originally for gcc, ported for MSVC in Cygwin
    - > <http://artax.karlin.mff.cuni.cz/~kendy/ccache/>

# The Tools: Icecream

- Sends tasks for compilation to another machines
  - Similarly to ccache, the source is preprocessed, and compiled, or sent for compilation somewhere else
    - > Has a scheduler that decides where to send the jobs
    - > Sends also the compiler to the remote hosts so that it is ensured that it is 100% same as on the original machine
  - Would be an ideal tool for the cross-compilation
    - > The task could be started on the Win32 machine, but distributed to Linux machines running MSVC in Wine!
    - > No changes needed for the OOO build system, just set CC and CXX to icecream
  - Unfortunately, the attempt to port it for MSVC failed so far :-(
    - <http://en.opensuse.org/Icecream>

More Fancy Stuff

The slide features a solid blue background. In the lower right quadrant, the text "More Fancy Stuff" is displayed in a white, sans-serif font. At the bottom of the slide, there are several horizontal, glowing white lines that create a sense of motion or depth, extending across the width of the slide.

# Real Cross-compilation (1)

- When I gave up on the Icecream port, why not try to tweak the OOO build system? ;-)
  - Copied MSVC, platform SDK, etc. to the Wine installation
  - Took 'solver' from Win32 and copied it to a tree compiled on Linux
  - Edited winenv.set.sh
    - > `CC="wine \"c:\\Program Files\\Microsoft Visual Studio 9.0\\VC\\bin\\cl.exe\""`
    - > Similarly other stuff
      - » Location of the includes, solver, COMPATH, SHELL, SOLARVER, ...
    - > But SOLARENV must point to the Linux solenv!
      - » To use the 'native' tools

# Real Cross-compilation (2)

- Edited several makefile.mk's
  - > Depending on if the tool is used in Wine or not, it has to use colons, or semicolons to delimit the includes
    - » COLON\_HACK := :  
\$(SRS)/shells.srs: \$(subst,z\$(COLON\_HACK), \$(SOLARINCDIR)) \  
\$/svx\$/globlmm.hrc
- Edited solenv/inc/unitools.mk and solenv/inc/wntmsci11.mk
  - > To use the Win32 tools in Wine, or native (Linux) tools for those Ooo-specific
- Successfully built Writer!, but with limitations
  - > Cannot build with debug
    - » <http://appdb.winehq.org/objectManager.php?sClass=version&iid=9569>
  - > Cannot sign the dlls

# The Crucial Question: Does It Help?

- Unfortunately no :-(
  - Native MSVC build time of sw: 78 minutes
  - Build time of sw with MSVC in Wine: 300 minutes
    - > 3.8 times more :-(
    - > Even with wineserver still running
  - [for comparison, PCH-enabled: 43 minutes!]

# Another Possibility: MinGW

- A gcc for Windows
- MinGW port of OOo is still in progress
  - Unfortunately still does build out-of-the-box
  - Provided several patches, and got sw to compile
    - > Took 126 minutes, 1.6 times more than MSVC
    - > No idea about run-time, parts still do not compile to get a full install
- The plus side is that it's much easier to use it for cross-compilation
  - It's gcc, after all ;-)
  - <http://www.dumbbell.fr/howto/win32-cross-compilation.en.html>

Conclusion



# Conclusion

- Without profiling & optimizing Wine, it does not make sense to use MSVC on Linux
  - Porting icecream to handle MSVC is viable
  - PCH is a clear winner, should be enabled by default!
- Split build would help tremendously
  - Helper tools as separate packages with an own release cycle
  - 3<sup>rd</sup> party libraries/tools as a separate package with an own release cycle
    - > Viable thanks to the current shorter OOO release cycle
    - > Would be even possible to provide binary versions
  - Separate I10n-related build tasks
- Big potential of MinGW

# Questions



**Novell®**