OOoCon 2004 - Berlin

Making the source more manageable for new Developers

Chris Halls

Debian OpenOffice.org Team / Credativ

Agenda

- Problems faced by new developers
- Solutions already available
- Some ideas for new solutions
- Discussion

OOo from the outside

- I often talk to other developers in the Debian community who hack on almost anything ...

- except OpenOffice.org

  - *Many bug reports in Debian's bug tracker*
  - *Recent security bug not fixed by Debian's security team*
  - *powerpc crash bug tagged 'help' for 1 year*
  - *OOo is NMU free :)*

Volunteers scratching itches

- More people would get involved if it was easier for them to 'scratch their itch'
- Volunteers have a limited amount of time
  - *We need to help them get productive quickly!*
- Small tasks need to be easy to work on

Problems facing new developers

- Very large source downloads
- Heavy requirements to build the source
  - *Build time*
  - *Disk space*

Problems already partly solved

- Solver binaries
- Use of system libraries

The Solver

- Contains output from compilation process for all projects
  - *Reduces compilation time*

- Work on any project by downloading the solver and compiling just the project you are working with

- Available for
  - *Linux x86/powerpc*
  - *Solaris*
  - *Windows*

Problems the Solver doesn't solve

- Solver reduces compilation time
- Effectively means you download the complete OOo twice
  - *No reduction in download sizes*
  - *Need lots of disk space*
- Only available for a small subset of available platforms and compiler versions

Using system libraries

- System version of library is selected at configure time *(--with-system-XXX)*

- *stlport4*
- *zlib*
- *jpeg*
- *expat*
- *freetype*
- *libxml*
- *python*
- *icu*
- *db3*
- *sablot*
- *odbc*
- *curl*
- *nas*
- *neon*
- *sane*

Using System Libraries

- Reduces OOo compilation time
- Reduces disk and memory footprint
- Can take advantage of updates and optimisations provided by system library builder
    - *libneon security bugs*
- But system libraries are not used by default
    - *Inexperienced developers do not always make use of them*

Ideas for improvements

- Buiding OOo using installed files and extra modules generated from the solver
  - *Use libraries from installed version*
  - *Precompiled build tools*
- More wild ideas
  - *Splitting the OpenOffice alias*
  - *Building translations separately from the code*

Using Installed OOo Libraries

- Shared libraries for OOo already available in *<prefix>/program*

```
acceptor.uno.so        libicui18n.so           libspa645li.so
bridgefac.uno.so       libicule.so             libspell645li.so
connector.uno.so       libicuuc.so             libspl645li.so
corereflection.uno.so  libj645li_g.so          libsrtrs1.so
implreg.uno.so         libldap50.so            libstore.so
introspection.uno.so   liblng645li.so          libsts645li.so
invocadapt.uno.so      liblocaledata_en.so     libsvg645li.so
invocation.uno.so      liblocaledata_es.so     libsvl645li.so
...
```

- If we had the headers available, we could compile OOo using these libraries

Precompiled build tools

- The build tools do not need to be compiled as part of OOo and are often not part of the finished product
  - *dmake*
  - *transex3*
  - *rsc2*
  - *cppumaker*
  - *javamaker*
  - *rdbmaker*
  - *svidl*
  - *bmp*

Precompiled build tools

- Contributors such as translators do not want the complete compiled source code - they want to generate resources using these tools
- People looking to work on a specific small fix in a library just want to be able to work on the library without building all the tools
- Having tools available could make cross-compilation easier
- Could we generate a new openoffice-tools installation package from the solver?

module-dev packages

- We could follow the convention of generating *module* and *module*-dev packages
- *module* package contains libraries & other files shipped for install
- *module*-dev package contains headers needed for building binaries that use this library
- Convention in Linux world:
  - libraries in /usr/lib
  - include files in /usr/include/<module>

module-dev details

- Contents of *module*-dev
  - Header files from each include/*prj directory*
  - Build tools?

- *.Set file would need to include existing OOo installation in include & library paths

- Debugging symbols not included in releases
  - *We could generate module-dbg package with library containing debug symbols*

Dividing up the OpenOffice alias

- The OpenOffice alias does not have to be checked out in one go

```
$ cvs co -c
[...]
OpenOffice   -a helpcontent transex3 i18n api offuh sdb dbaccess connectivity
             sfx2 offmgr res idl desktop sd sch sip goodies svx vcl rsc toolkit
             UnoControls forms setup2 scp scptools instsetoo readlicense extras
             wizards so3 sj2 ie sal boot sc scaddins basic basctl sw starmath
             lingu solenv dmake rscpp xml2cmp jtools config_office chaos inet
             uui ucbhelper store ldapber cppu unoidl cppuhelper javaunohelper
             jurt bridges remotebridges stoc cpputools registry codemaker
             rdbmaker vos unoil tools svtools std2 io eventattacher unzip
             unotools extensions external configmgr sot automation comphelper
             jut officecfg sandbox fileaccess xmlhelp xmloff sax framework sim2
             ucb drafts udkapi offapi examples ridljar bonobo lingucomponent
             package linguistic XmlSearch product odk help dtrans udkwww apiwww
             DocumentProperties xmlscript chart salhelper idlc mkdepend
             soltools sysui shell testshl psprint padmin testtools

[...]
udk_src      -a solenv sal dmake xml2cmp config_office cppu unoidl cppuhelper
             javaunohelper jurt bridges remotebridges stoc cpputools registry
             codemaker rdbmaker vos unoil ridljar salhelper idlc mkdepend
             soltools io jut sandbox store udkapi product testtools logging
             ext_log4j
```

Large or small alias?

- Advantages of the large alias
  - *Inter module versioning is simpler – no special care must be taken to make sure modules are in sync*
  - *Fits the Sun environment, where solvers are available via NFS mounts in the internal network*

- Disadvantages
  - *Huge source size*
    - *1.0.3 - 180MB*
    - *1.1.2 – 215MB*
    - *1.9 m52 – 226MB*
  - *Duplication of external libraries already on the system*

Separating translations

- Translators don't care about compiling binaries and programmers don't care about all the translations
- Many projects recognise this and package translations separately
- In OOo, all languages are bundled together in .src (1.1.x) / localize.sdf
- Can L10N_FRAMEWORK be used with precompiled tools instead of a complete source tree?

Ideas for improvements

- Buiding OOo using installed files and extra modules generated from the solver
  - *Use libraries from installed version*
  - *Precompiled build tools*
- More wild ideas
  - *Splitting the OpenOffice alias*
  - *Building translations separately from the code*