AMD

LLVM STABLE RELEASES

TOM STELLARD

OCTOBER 28, 2014

Agenda

AMD

- Why stable releases?
- Use Case: Mesa3D.
- Stable release history.
- Stable release process.

Why Stable Releases?

AMD

- Make life easier for LLVM users.
  - Especially operating system distributions.
  - There are a large number of users maintaining their own stable trees.
  - Often with limited input from LLVM developers.
  - Limited testing as well.
- Shorter release cycle for major releases.
  - Fixes deemed too risky or low importance can be pushed off until next stable release.
- Improved testing coverage.
  - Released versions of LLVM get more widespread use than SVN.
  - 2+ months of extra testing for releases.

Why Stable Releases?

AMD

- Keeps projects using released versions involved.
  - It's very common for projects to stablize with a released version of LLVM.
  - Stable releases give them the opportunity to contribute fixes.

Who Benefits from Stable Releases?

AMD

- Operating System distributors:
  - Linux: Ubuntu, Fedora, Debian, Gentoo, etc.
  - FreeBSD
- Other open source projects
  - These projects rely on released versions of LLVM and can't always ship their own LLVM copy.
  - Examples: Mesa, POCL
- Proprietary software developers who use LLVM.

Stable Release Use Case: Mesa3D

AMD

- Mesa3D is an Open Source GPU driver project.
- Contains drivers for a several different GPUs at varying levels of completeness.
- 3 month release cycle.
- Depends on official LLVM releases packaged by Linux distributions.
- Drivers that use LLVM:
  - llvmpipe - Software rasterizer
  - r600g - OpenCL™ AMD Evergreen and Northern Islands GPUs.
  - radeonsi - OpenCL™ and OpenGL AMD Southern Islands GPUs and newer.

Stable Release Use Case: Mesa3D

AMD

- llvmpipe
  - New driver features not dependent at all on compiler changes.
  - Main challenge is bugs that force disabling of optimizations.
- r600g / radeonsi
  - Tight integration between driver features and compiler features.
  - Not practical to wait 6 months for bug-fixes.

- Mesa3D really needed a solution for getting bug-fixes into distributions packages.
- First solution: Send bugfix patches directly to package maintainers.
  - Very time consuming.
  - Ended up breaking Ubuntu's Clang 3.2 package.

Stable Release Attempt - 3.3.x

AMD

- First attempt at a stable release.
- Not enough testers to complete the release.
- Release qualification requirements were unclear.
- There were some positives:
  - Added extra number to LLVM version.
  - Release script improvements.
  - Raised interest in stable releases.

Stable Release - 3.4.x

AMD

- First stable release series.
- Challenges:
  - Dealing with extra number in LLVM version.
  - Deluge of merge requests at the last minute.
  - Rookie release manager.
  - C++11 not allowed.
- 3.4.1
  - Various bug fixes for targets.
  - C++11 implementation fixes.
- 3.4.2
  - Build fixes with gcc 4.9, which was released after 3.4.0.
  - Fixed soname of libLLVM-3.4.so

Planned Stable Release - 3.5.x

AMD

- Planned for November 2014.
- New SVN tag naming convention in place. Will help simplify process.
- Process changed: developers allowed to merge their own changes.

AMD

- Community members (developers or users) nominate a patch to be merged into the stable branch.
- Stable patch rules:
  - Must be approved by the code owner and the release manager.
  - In most cases should have already been merged to ToT.
  - Patches cannot break shared library ABI
  - e.g. A program built against libLLVM-3.4.0.so must be able to link with any libLLVM-3.4.x.so
- Once approved, community member or release manager merges patch into stable branch.

Process Challenges

AMD

- What's the difference between a bug and a feature?
  - Hard to figure out where to draw the line in some cases.
- Testing for ABI compatibility.
  - Current Tool: http://ispras.linuxbase.org/index.php/ABI_compliance_checker
  - Seems to have trouble with newer versions of gcc.
  - There were a few false positives in the 3.4.x release.
- Merge requests come in at the last minute.
- Merging patches can be difficult.
- Helping new testers get up to speed.

Process Improvements

AMD

- Patch nomination:
  - Separate mailing list for stable branch patches.
  - Annotating commits so they can be automatically merged by a post-commit hook.
- Who merges the patches: developers or release manager?
  - Release manager can ensure patches adhere to release rules.
  - Merge mistakes are a risk with the release manager.
  - It is less work for developers to have release manager merge.
  - Less work for release manager to have developers merge.
- Automated Testing
  - Public buildbots for stable branches.
  - Better automation for testers.

Stable Release Goals

AMD

- Developers get in the habit or merging most bug-fixes to the stable branch.
- Stable branch buildbots.
- Cheaper releases with more automated testing.
- Other ideas:
  - Longer lifetime for stable releases.
  - Validation on more targets.

How you can help

AMD

- Volunteer for release testing.
- Update release documentation.
- Improve release scripts.
- Search the bug tracker for patches that can be backported.

Questions

AMD

DISCLAIMER & ATTRIBUTION