SAMBA Opening Windows to a Wider World

The Samba Development Process

Jeremy Allison
Samba Team

jra@samba.org

In the beginning was the ftp site

arvidsjaur.anu.edu.au

And the “patch” command..
(let's not forget diff also)

Q.A. Originally not a big concern

"If it compiles, it is good; if it boots up, it is perfect."

Linus Torvalds

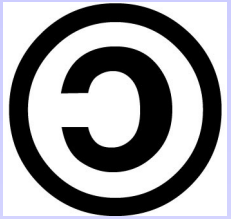
What's in a name ?

- In April 1994 a company called Syntax informed tridge they had a trademark on the name 'smbserver', and politely requested a name change.
  - smbserver became Samba.
- A good name gives a sense of identity for the developers.
  - Much easier to identify as “Samba developers” than “smbserver developers”.
  - Adds logo possibilities, T-shirts, merchandising possibilities.

What's in a license ?

- Originally under a “no commercial use” license, tridge was introduced to Linux and the GPL by Dan Shearer.
  - Adopted GPLv2+ for subsequent releases.
- Many opinions on the “correct” Free Software license to encourage contributions, but I firmly believe GPLv2+ was the correct decision to help build an early community of developers in what was a proprietary-dominated space.
  - 'Everyone shares' gives a sense of fairness and ownership to the project.

Who owns the code ?



- Tridge always accepted patches from anyone (as did Linus for Linux).
  - Copyright ownership wasn't even considered at the time.
  - Much more important now corporations are involved in Free Software. © assignment agreements are commonplace.
- By the time Samba became widely used, ownership was so widely distributed it was impossible for any one part to claim ownership rights on the code.
  - Gave a few problems later when corporations got involved.

The role of release manager

- The concept of a 'release manager' started with Samba 1.7 – July 1994.
  - For the first time tridge didn't just upload a tarball containing his current source code to the ftp site.
  - The original 'release' process was a script that tridge ran to ensure everything needed to build the code was included.
- I inherited the role of release manager for 1.9.
  - Started to add more steps to the process, ensuring compilation on more platforms, simple tests after install etc.

Source code control

- Samba finally entered the 'modern' age of tracking source code changes in May 1996.
  - Moved to cvs for Samba version 1.9.16.
  - Source code was always controlled, just by one person.
- Giant step forward in development.
  - Allowed remote checkouts of the current 'master' repository.
  - 'Branches' gave the ability for different features to be worked on simultaneously.
  - Basics of workflow still used today with git.

Source code control and the Samba Team

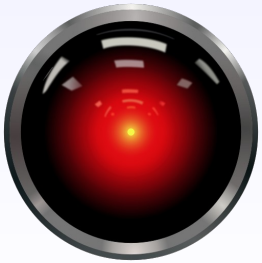
- The addition of source code control was the genesis of the idea of the “Samba Team”.
  - People with direct commit access into the source code repository were privileged.
  - Public shaming was used to prevent build breakage.
    - Only worked if people were ashamed :-).
- Goal set if you worked hard enough, submitted enough good patches, you could get direct commit access and join the 'Team'.
  - Partly used to avoid the work of merging large patches :-).

The end of the makefile – Autotools, then waf

- For Samba 2.0.0 – January 1999 – tridge spent *months* of time re-writing the build system.
  - No longer a hand configured 'Makefile' to match your particular system, now a set of automated tests (written in m4) for specific platform features.
- Revolutionized the ability to make code 'generic' and have underlying modules select the right code per platform.
  - Only the quota subsystem still a mess (to this day :-).
- After many arguments current popular build system is 'waf' – python based (autotools build still maintained).

Changing platforms – the rise of Linux

- Around the turn of the century, the primary platform for Samba changed from SunOS/Solaris to Linux.
- Many other minor platforms have since become almost impossible to maintain, build farm notwithstanding.
- We now have Linux-specific interfaces (per-thread creds).
  - Do we go...
  - The systemd route (Linux as the new POSIX !) ?
  - Functionality isolation "*I'm sorry Dave I can't do that on this platform*" ?



Enter the OEMs

- In the early days it was not certain whether Samba was a finished product or a technology kit of parts.
  - Early Samba releases attempted 'product polish'.
  - SWAT was a (bad) example of this.
- OEMs were very clear. They needed a set of technologies they could customize.
  - We still need work on modularity.

Two steps forward, one step back..

- Samba development faced many challenges around the turn of the century.
  - Multiple Windows clients.
    - DOS, Win9x., Windows NT, Windows 2000, New MacOS clients (Thursby).
- Every time we fixed a bug we faced the danger of adding bugs against another platform, or regressing in another area.
  - Testing the fix doesn't help. Every fix is correct for the bug it's addressing. We had no way of discovering what we might have broken.

Test driven development

- Originally driven by trying to understand the DOS wildcard matching algorithm.
  - We finally understood the only way to progress was to write client test code against a Windows server, then make Samba behave the same.
  - Uncovered some interesting bugs in both systems :-).
- Samba is now 90% test driven development.
  - Code only modified once we have proof Windows behaves this way.
  - The other 10% are Samba-specific features.
    - Constrained by our own legacy systems.

"*My Life Is Driven By Bugzilla..*"

- Project infrastructure is really important.
  - Currently SerNet runs ours.
  - Many projects use external hosting (SourceForge → GitHub).
- Bugzilla is the collective memory of the Samba project problems.
  - Several Team members get CC:ed on every change made to the bugzilla.
  - Extremely important to monitor the problems people are having, and the relative quality of a release.

The build farm

build samba

- Created in 2002 (approximately) by tridge and Andrew Bartlett.
- Allows any arbitrary host to download a Samba source checkout, build it and run tests – all as a non-privileged user.
- Results then uploaded to build.samba.org so developers can analyze the results.
- Allows “strange” platforms to be tested without Team members having access.
  - Less used now Linux is the 800lb gorilla in the POSIX space.

Static analysis - Coverity

- Luckily for Samba, we were chosen as a test case for Coverity static code analyzer.
  - Volker and I (and others) jumped on the analysis, fixed over 250+ bugs reported.
  - No security issues found (yet).
  - Some tridge code confused the analyzer.
  - Mostly error code untested paths.
- Coverity (occasionally) keeps the scan updated.
  - Ongoing effort to keep up to date – keeps quality of code high.

Spoiling the fun – the Microsoft Documentation release

- All of a sudden we had a magic box that would answer any protocol question.
- Initially changed the dynamics of protocol correctness, just a matter of looking at the docs.
- Eventually we realized that an imperfect oracle is no better than no oracle.
  - Went back to writing tests :-).

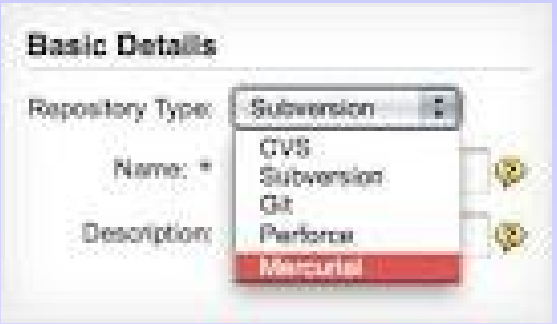
Formalizing the release process

- Credit must go to Karolin Seeger for this change.
- Formal control of all branches that are released to the public.
  - Bug reports required for all changes going into the release branches.
  - Two Samba Team engineer sign-off of all changes going into release branches.
  - (Preferably) regression tests for every change.
- Reliable, stable, timely release train has greatly improved Samba release quality.

Stability matters – freezing the VFS

- After Samba 3.4.x, commitment was made to keep the VFS API/ABI stable across a major release (3.x.0 → 3.x.final).
  - Turned out this was harder than it looks.
  - Several internal structures exposed into the VFS.
  - Cleaned up significantly in 4.0.x – API boundary clarified.
- Team member Richard Sharpe is the external VFS interface expert – wrote a wonderful white paper on writing a Samba VFS module.
  - Easy to use VFS is one of the reasons Samba is still vibrant and actively developed.

Distributed Version Control: moving to git



- Ex-Team member Jerry Carter evaluated all the distributed version control systems, decided on git as the next generation source code control for Samba.
  - Prescient choice – git is *the* winner in Free Software dvcs.
- Chosing git theoretically removes the “central master repository” for Samba.
  - Practically this is not the case – we still use it as a CVS/SVN on steroids.
  - Associated tools should make the leap to review-driven development much easier.

Tracking the code

- For a long time Samba only allowed personal © on contributions.
  - Made things difficult for corporate employed developers (typical California contracts).
  - Made things hard for OEMs who needed assurance that code contributions were legal.
- We recently introduced a similar form to the one used by the Linux kernel, where a developer asserts ownership and rights to submit under the license.
  - Commits should now contain a “Signed-off-by:” assertion.
  - Not all Team members using this correctly :-(.

Review-driven development

- Formal change proposed (and rejected), but then implemented in practice:
  - All code must be reviewed by two Team members.
    - One can be original author.
  - Code should be proposed as a [PATCH]: subject on the samba-technical mailing list.
  - Reviewer push, not author (not always followed in practice).
- Reviewer must add Reviewed-by: Tag to the commit.
- Complaint was this slowed down development.
  - Hasn't been much of an issue in practice.

The Micro-Patch revolution

- Concept created by Volker Lendecke, the idea is to break a changeset into a large number of minimal changes, each of which is easily reviewed.
  - The changesets continue to pass all regression tests on every single micro patch (this is sometimes very hard to achieve).
- Forces coders to really *THINK* about every single change, and allows much easier access for reviewers.
  - Minus is that it can slow down development.
  - I create large patches that achieve the required functionality, then spend time breaking them down.
    - Some Team members find this process frustrating.
    - Need a very detail oriented mind to make this work.

Attack of the fuzzers

- Codenomicon (and others) provide a fuzzing protocol product that attack network protocols based on knowledge of internal protocol structure.
- Run against Samba as proof of effectiveness of their product (good thing we're treated as a test case :-).
  - Devastating results on first runs (many, many, crashes).
  - No security holes found (yet).
- Similar to Coverity, they are willing to keep running against Samba as free advertising for their services.

Scaling up and out – ctdb, IBM and SwiftTest

- IBM has been pushing Samba (SONAS) into large-scale environments (tens of thousands of connected clients).
  - ctdb is the core engine allowing this to work.
- Client load test generators such as SwiftTest allow large scale simulation of client loads.
  - Not practical for most Team members to try and reproduce locally.
- Simple correctness changes can wreck large-scale performance.
  - Outstanding problem, without good regression tests.

Library spin-offs – talloc, tdb, ctdb, tevent

- Samba has created many externally useful technologies, adopted by other projects.
- Goal is to spin off mature libraries into separate projects.
  - Prevents Team members from making 'one more change' to external code.
    - The talloc_reference() disaster.
- Difference of opinion on what projects are ready to leave the Samba house and live on their own.
  - talloc, tdb certainly.
  - ctdb, tevent maybe.

Challenges

- Code complexity.
- A Windows-less world.
- A big ball of mud.
- Scaling large, shrinking small.
- A Cloudy future ?

Managing Code Complexity

- Samba is now too complex for one person to understand fully.

A Windows-less world

- None of the core Samba Team developers depend on Windows. For *ANYTHING*.

A big ball of mud

- A Samba split into separate components with clean interfaces would be much easier to develop.

Scaling large, scaling small

- We need to keep Samba running on Raspberry PI to IBM mainframe systems.

Remaining relevant in a "Cloud Storage" World ?

Questions and Comments ?

Email: jra@samba.org

Slides available at:

ftp://samba.org/pub/samba/slides/sambaxp-2013-development.odp