git-annex

manage files with git, without checking their contents in

Richard Hartmann,
RichiH@{freenode,OFTC,IRCnet},
richih.mailinglist@gmail.com

2012-02-05

Outline

1. Intro
2. Use cases
3. Technical details
4. Outro

Outline

1. Intro
2. Use cases
3. Technical details
4. Outro

Who am I?

- Project & Network Operations Manager at Globalways AG
- freenode & OFTC staff
- Passionate about FLOSS
- I am not the author of git-annex, but an interested early adopter

What is git?

- Version control system
- Distributed
  - No need for central repository
  - Commit while offline
- **Full** history of all files in every checkout
- Best version control system available (imo...)

What is git-annex?

- Based on git
- No need to check files into git
- Still able to check files into git if you want
- Able to maintain full history, but does not do so by default
- Written with low bandwidth and flaky connections in mind
- Various work-flows

Outline

1 Intro

2 Use cases

3 Technical details

4 Outro

The Archivist

- Put data into git-annex
- Distribute data among any number of drives, tapes, remotes, etc
- Store offline media in a safe place
- Maintain full information about number and location of all copies

Media consumption

- Import podcasts, videos, and slides
- Sync or export to consumption devices
- Consume media
- Drop consumed media from annex
- Deletion propagates through all annexes over time

The Nomad

- Keep copies of data on www
- Optionally sync between several local devices for backup
- Add data locally and/or remotely while on the road
- Sync data between local and remote once at an Internet café or similar
- Perfect for photos while travelling

Outline

1 Intro

2 Use cases

3 Technical details

4 Outro

Internal workings 1/2

- Written in Haskell, so strong typing etc internally
- Uses rsync to transfer data
- Moves files into `.git/annex/objects`
- Makes files read-only
- Puts symlink in place of file
- Stores location data in branch `git-annex`
- User adds and commits symlinks to master branch

Internal workings 2/2

- Read-only files force you to `git annex unlock` prior to changing them
- Ensures that you will `git annex add` all unlocked files
- git-annex can then discard or keep old data, depending on setup

Data integrity

- Set minimal number of required copies per suffix, directory, etc
- SHA1, SHA2-{224,256,384,512} for integrity
- All remotes and special remotes can be verified
    - remotes verify locally and transmit the result
    - special remotes transfer all data to verify
- Verification takes required amount of copies into account
- `git fsck; git annex fsck`

Special remotes 1/2

- Stores data in non-git-annex remotes
- Tracks all data stored in special remotes
- Supports encryption for storage on untrusted machines/media
- Hook system lets you write to and read from arbitrary remotes

Special remotes 2/2

- bup
- directory
- rsync
- S3, Swift, etc
- Tahoe-LAFS
- web (media.ccc.de, Project Gutenberg, archive.org, etc)

Outline

1. Intro
2. Use cases
3. Technical details
4. Outro

Where to get it

- `cabal install git-annex --bindir=$HOME/bin`
- Native packages for
  - Debian
  - Ubuntu
  - FreeBSD
  - Arch Linux
  - NixOS

Further reading

- https://github.com/RichiH/talks
- http://git-annex.branchable.com/
- http://www.slideshare.net/RichiH/

Thanks!

Thanks for listening!

Questions? Follow me outside when my time-slot is over.

See slide footer for further contact Information.