Filesystems, oh my!

Dr. C. Scott Ananian <cscott@laptop.org>

A little story

- I'm old. I use hierarchically-structured paths *everywhere*
- But I couldn't convince the young kids they were *vital*
- So I prepared to **blind them with science**
- The rest of the story is at: http://wiki.laptop.org/go/Experiments_with_unordered_paths

tagfs

- What if 'cd foo/bar' put you in the same place as 'cd bar/foo'?
- Surprisingly many things Still Work.
- /home/cscott/mstone is the same as /home/mstone/cscott
- /etc/sysconfig/cscott/home is the same as /home/cscott/etc/sysconfig

Olpcfs

Design goals

- Filesystem w/ POSIX semantics
  - This is the codeword for “standard filesystem”. Windows, UNIX, and MacOS in various flavors have (more-or-less) POSIX-compliant filesystems.
  - Our first generation datastore was a “simple” proprietary wrapper: let's move forward!
  - Aim to provide ***best possible*** support for legacy applications

Design goals

- Content-addressable
  - Lots of attempts out there to create global distributed filesystems with unified namespaces – *let's not try this!*
  - Local arrangement & organization of documents is up to the individual user; all we need is an opaque tag to call it by.
  - Commercial support: XAM/Honeycomb (Sun)/Jackrabbit (Apache), etc, etc.

Design goals

- Versioned
  - Support exploratory learning by always allowing user to undo his most recent mistake.
    - “Continuous” versioning.
    - Snapshots don't work for this.
  - Groups of files may have independently modifiable *tagged versions* (“full persistence”)
    - Gives us our P_SF_CORE/P_SF_RUN support
    - Also useful when importing collaborative work

The olpcfs *filesystem*

- Transparent **versioning**
  - Reach back and study the past – then change it!
- Rich **metadata** via POSIX xattrs
  - Enhanced by mechanisms to treat metadata as 1st-class files
- Integrated metadata **indexing**
  - Unifies “Journal” and “files & folders” views

Demo

- http://wiki.laptop.org/go/Olpcfs has pointers to the source
- 2,500 lines of Python
  - Bdb and python-fuse packages
- FUSE impressions:
  - I prefer managing directory objects, rather than being given full pathnames

Sync'ing & sharing

- All objects have “XUID”
  - Content-addressable
- Distributed indexes of various scopes on top of local index
- Not all local objects may appear in filesystem tree!
  - Some may be imported into index only

Sync'ing & sharing

- XUID encapsulates *object plus metadata*
  - “Who's got this XUID?”
  - “I'll tell you which XUIDs I don't have if you'll tell me your XUIDs.”
- Independently-modified documents may result in tagged versions in filesystem after import

Cowfs

- Thin FUSE layer based on hardlinked fs
  - Born from a vserver disaster
- User-space locking == ++good
- Intercept when we open for writing, and break hardlink
- Use kernel caching
  - But we need direct writes

Rcsfs

- Kinder, simpler versioning filesystem
- Last time a VCS was made without directory support: 1982
- Extremely well-supported file format

The thin hero

- Mount over legacy filesystem (FAT, etc)
- When unmounted, it's just RCS directories
  - Lots of tools
- Files which are not mutated have no overhead

Just another cow, ma'am

- Rcsfs implementation based on cowfs!
  - Just check in previous version where you would break a link
- All versions are directly writable
  - Writing an existing version creates a branch
- User-space version vaccuuming

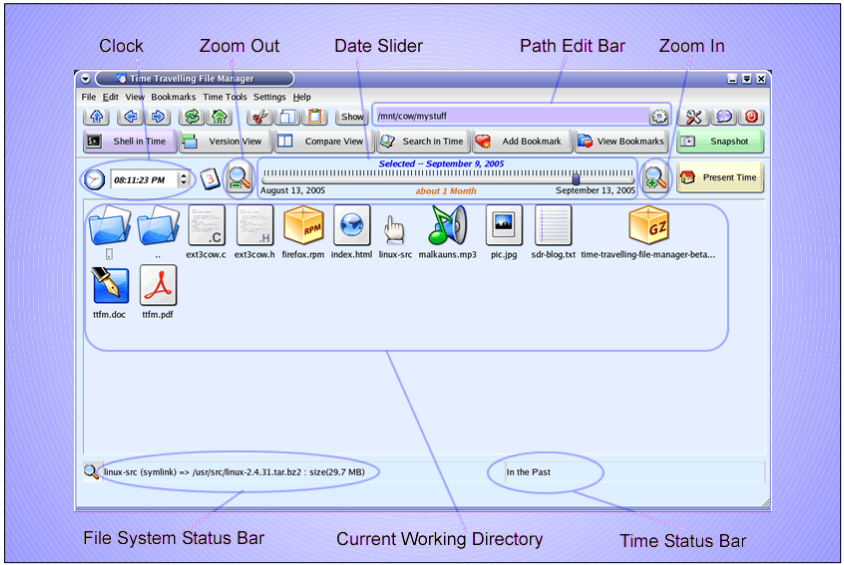
Fanotify

- Cowfs could be better: it could be a userspace client, not even a filesystem!
- Fanotify is being developed by Eric Paris as an antivirus system
  - Has ability to block I/O until approved by userspace
  - Blocking lets us do very interesting things!

User-facing tools

What not to do

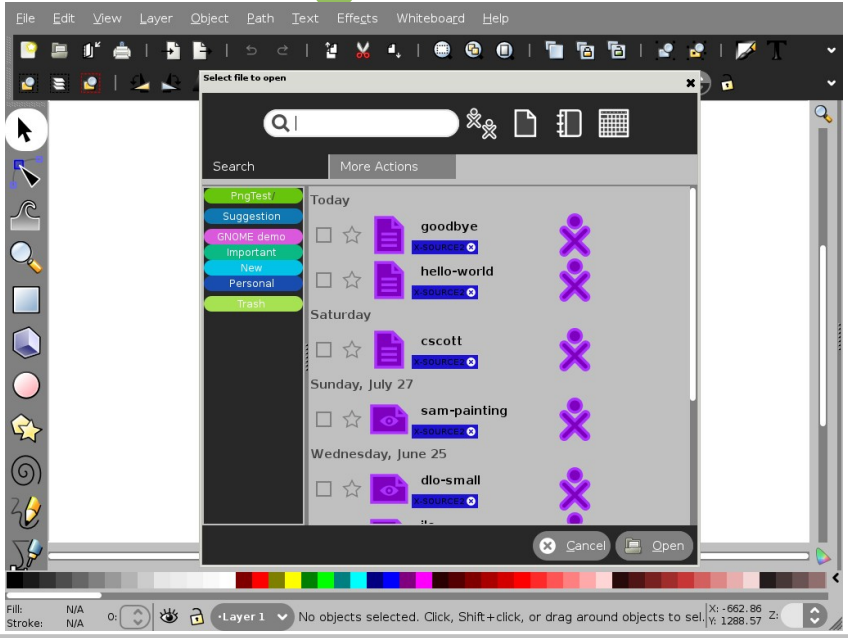
- A “time-travelling file manager”



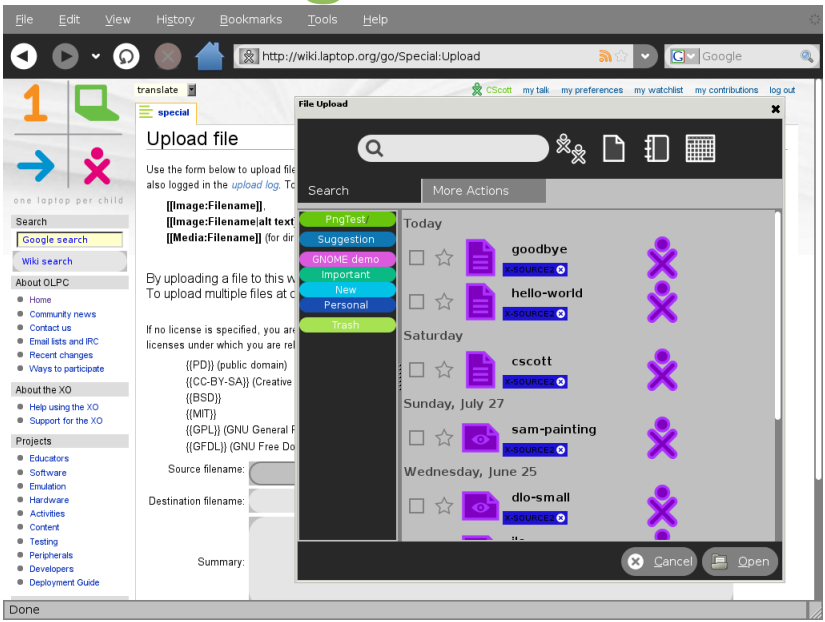
Legacy apps rock!

- There are so many of them
- GTK already has a nice abstract GtkFileChooser interface
- Nothing says that your GtkFileChooser has to be in-process...

Embedding demo (Inkscape)



Embedding demo (Firefox)



Bitfrost security

- Because the journal “file chooser” is out-of-process, untrusted apps don't need full access to user files
- The journal displays files, and then arranges to make available *only the selected one*
- We're also in the loop for saving, and can add metadata, etc.

THE END

(p.s. I need a job)