Git storytime

back in my day...

2016-11-10
joe@flatiron.com

Quasi-technical
Lots of links in slide notes

Who remembers …?

Apple II in 1989 by Jordan Mechner
Pioneer of its time. “Rotoscoping”. At least 12 sequels, one of the most successful video game franchises ever

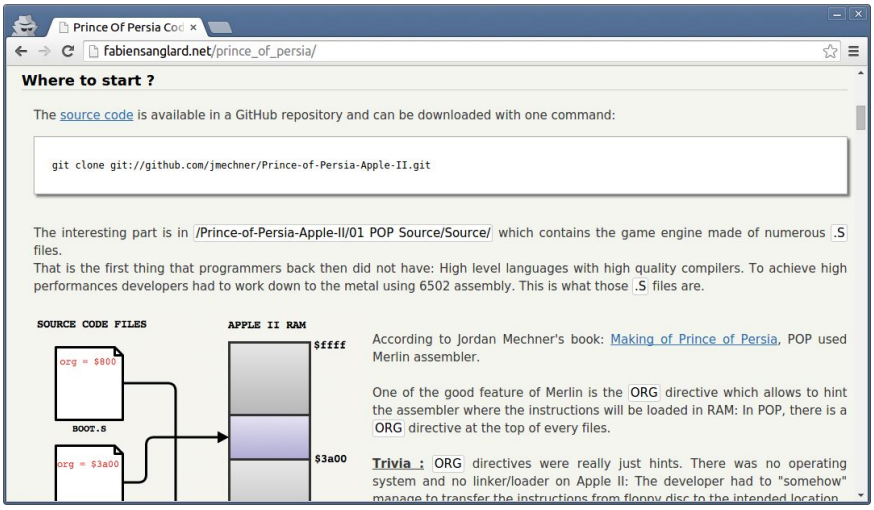
dude lost the code!

dude lost the code!

Why didn't I use Git?

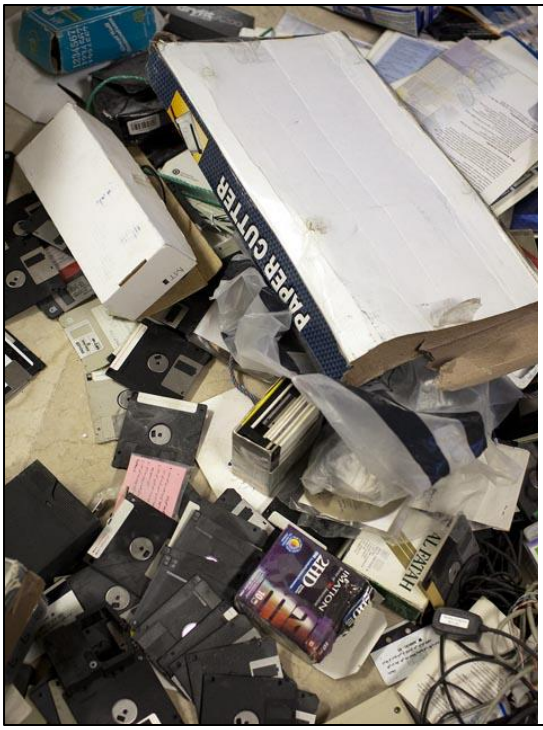
oh… it wasn't invented yet

Happy ending



https://github.com/jmechner/Prince-of-Persia-Apple-II
http://www.wired.com/2012/04/prince-of-persia-source-code/
http://fabiensanglard.net/prince_of_persia/

Version control Dark Ages



Version control
Dark Ages

Version control

Why talk about git?

Know your tools

Git is a humble tool.
How many people use Git? Pianist knows how a piano works. Might help demystify

Why talk about git?

Learn from its experience and design

One of the most successful software projects of all time. Anthropology

Why talk about git?

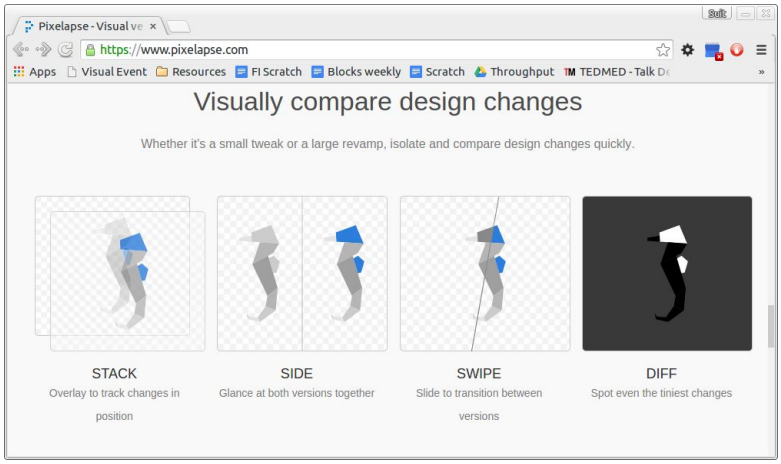
Contribute to our dialog about innovation

Git revolutionized how people thought about version control. In the same way Flatiron is revolutionizing how we think about healthcare data with innovations like abstraction and data linking

Git archaeology

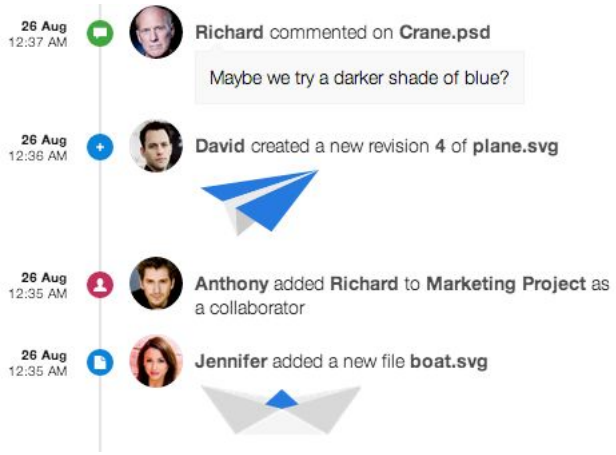
Most people talk about what git is. I want to talk about how it came to be (what it was)
Like an archaeologist, studying the past to understand how we came to where we are today

Version control



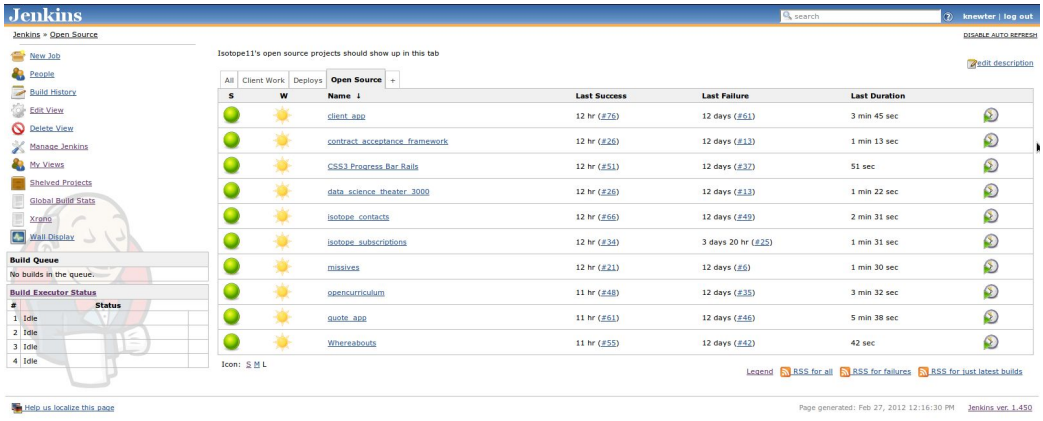
See changes

Version control



Encourage communication and collaboration

Version control



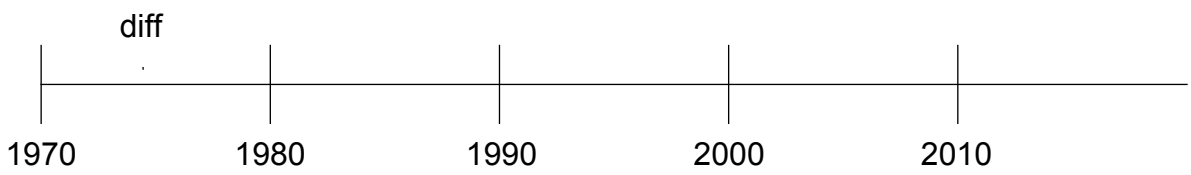
For most engineering organizations, a central system that the development ecosystem is built around.

Version control

- Understand changes over time
- Facilitate communication and sharing
- Hub of software development ecosystem
- Don’t lose Prince of Persia’s source code!!

Manages different revisions of content, in particular source code (SCM)

History



1974. Basically a dynamic programming research project at Bell Labs

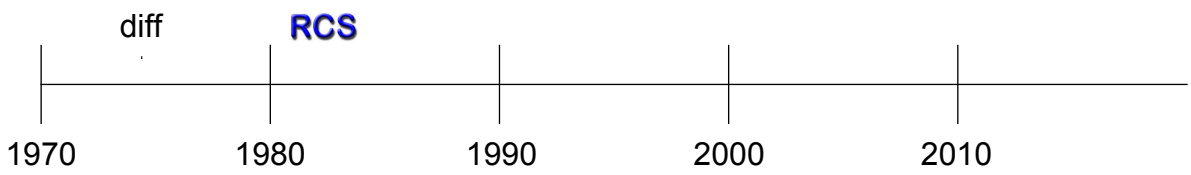
diff

```diff
--- /path/to/original   ''timestamp''
+++ /path/to/new  ''timestamp''
@@ -5,16 +11,10 @@
 compress anything.

 It is important to spell
-check this dokument. On
+check this document. On
 the other hand, a
 misspelled word isn't
 the end of the world.
@@ -22,3 +22,7 @@
 this paragraph needs to
 be changed. Things can
 be added after it.
+
+This paragraph contains
+important new additions
+to this document.
```

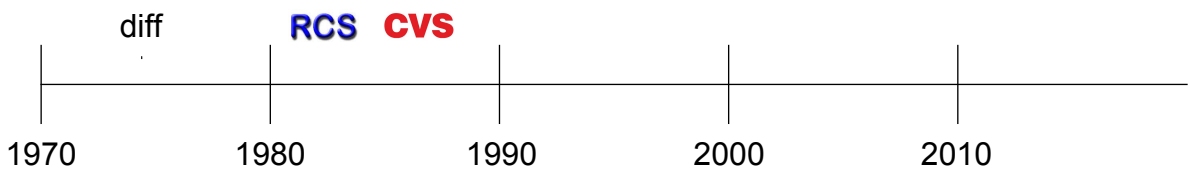
Very low-level tool. Sometimes used for ad hoc space-efficient storage of revisions

History



1982. Manage versions of a file by storing series of diffs. Like making manual copies

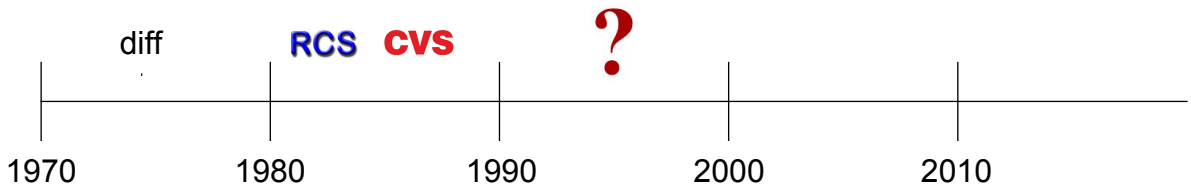
History



1986. Script on top of RCS to manage revisions of multiple files committed together. Introduced notion of a changeset

Original release:
https://groups.google.com/forum/message/raw?msg=mod.sources/eqze_AHbIK0/uE90wCq3ui4J

History



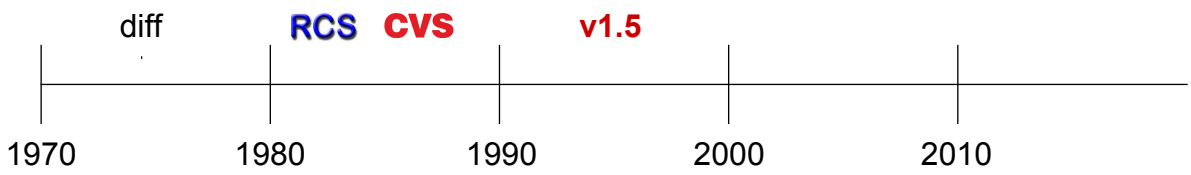
Early to late 90s, one of the greatest technical shifts of our time

History

diff

1970 19 0

History



1995. Client/server

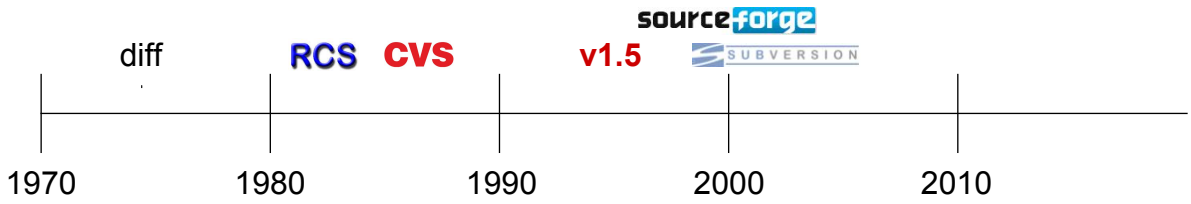
CVS 1.5

Release notes:

It uses reliable transport protocols (TCP/IP) for remote repository access

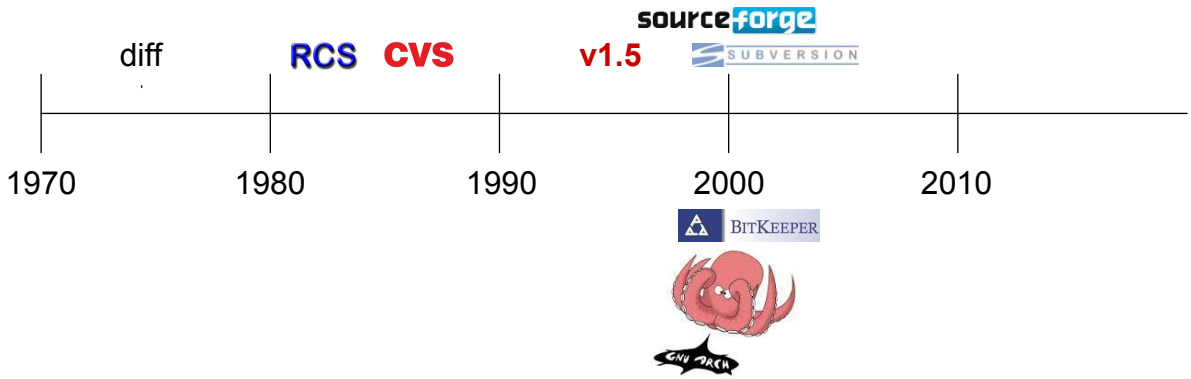
Reasonably good for development by centralized teams

History



Success! 1999 SourceForge (CVS hosting). 2004 Subversion v1. CVS “done right”

History



The first distributed version control systems

Distributed version control

Some key characteristics:

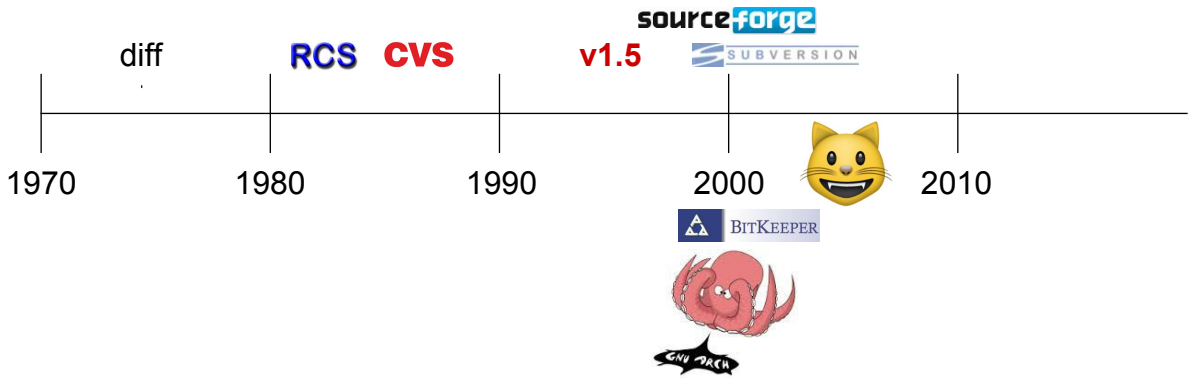
- Change set-based (instead of file-based)
  - No file locking
- Non-linear history
- Most operations are local
  - Full repository locally
- Optimized for merges

Meet Linus

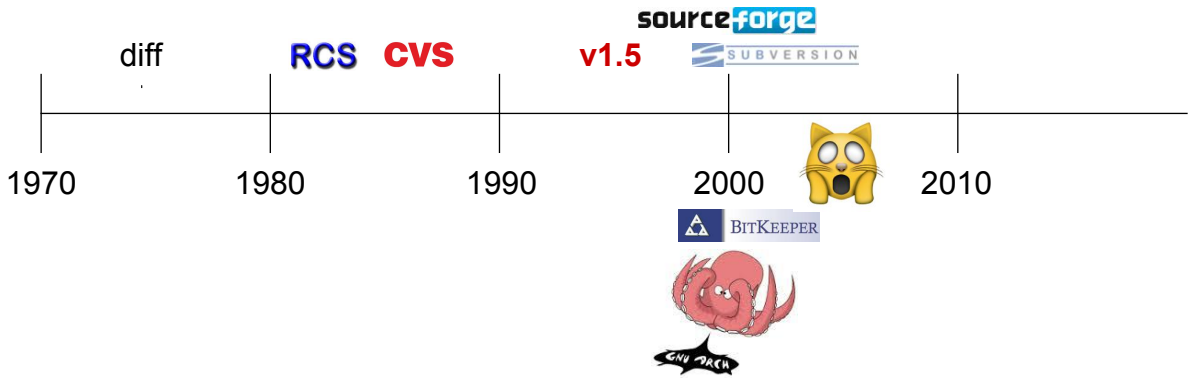
- 46 / M / Oregon
- Likes walks on the beach
- Databases are for brain deads
- Creator of Linux
  - Most popular open source project
  - Ever.

Linux one of the largest open source projects
Decentralized by nature
Large volume
Smart mofo
Linus decides to use BitKeeper (that’s why he’s smiling)

History


diff
RCS
CVS
v1.5
sourceforge
SUBVERSION
1970
1980
1990
2000
2010
BitKeeper
GNU ARCH

History



April 2005. :scream_cat: ; calamity ensues

Falling out with BitKeeper

BitKeeper was commercial and closed source, but offered a free license. Reverse engineering lost favor with vendor
https://lwn.net/Articles/130746/

Falling out with BitKeeper

From: **Linus Torvalds** <torvalds <at> osdl.org>
Subject: **Kernel SCM saga..**
Newsgroups: gmane.linux.kernel
Date: **2005-04-06** 15:42:08 GMT

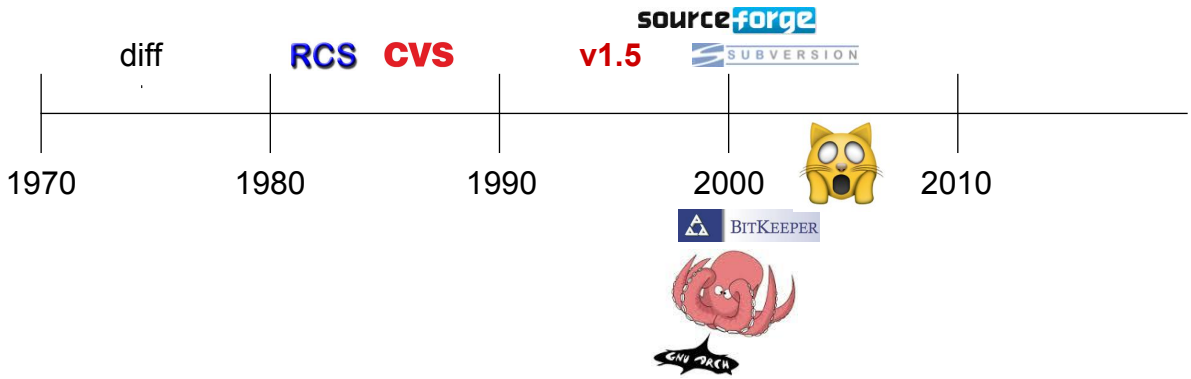
Ok,
as a number of people are already aware (and in some cases have been aware over the last several weeks), we've been trying to work out a conflict over BK usage over the last month or two (and it feels like longer ;). That hasn't been working out, and as a result, the kernel team is looking at alternatives.

[And apparently this just hit slashdot too, so by now _everybody_ knows]

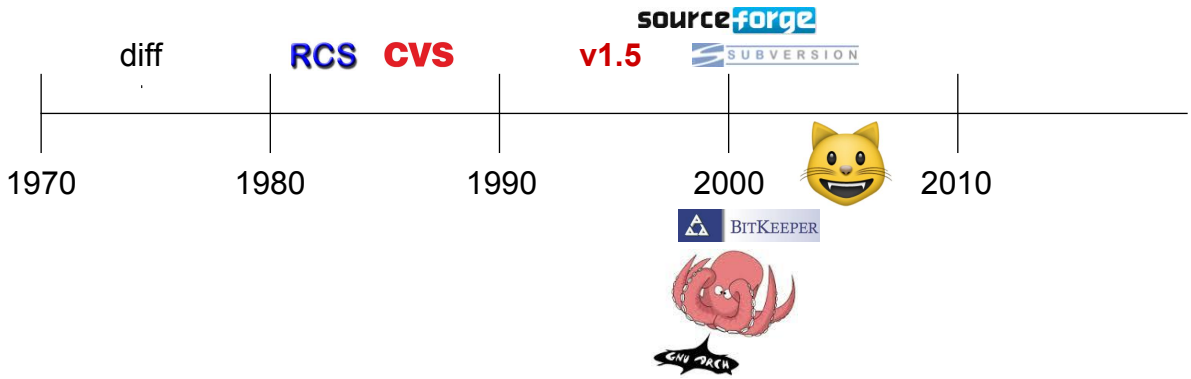
There are no alternatives to BitKeeper. Refuses to use a centralized VCS, every other DVCS sucks
Kernel will take a break for a few weeks

Like MTA on strike

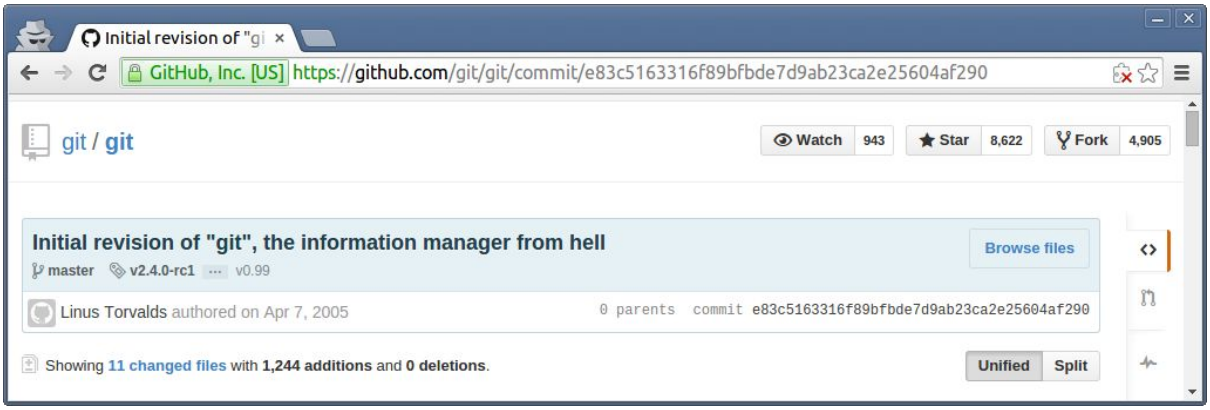
History


sourceforge
diff
RCS
CVS
v1.5
SUBVERSION
1970
1980
1990
2000
2010
BitKeeper
GNU ARCH

History


diff
RCS
CVS
v1.5
sourceforge
SUBVERSION
1970
1980
1990
2000
2010
BitKeeper
GNU ARCH

Git v0.0



A tool for quickly merging patch series

https://github.com/git/git/commit/e83c5163316f89bfbde7d9ab23ca2e25604af290

Why git?

```
The name "git" was given by Linus Torvalds when he wrote the very
first version. He described the tool as "the stupid content tracker"
and the name as (depending on your way):

 - random three-letter combination that is pronounceable, and not
   actually used by any common UNIX command.  The fact that it is a
   mispronunciation of "get" may or may not be relevant.
 - stupid. contemptible and despicable. simple. Take your pick from the
   dictionary of slang.
 - "global information tracker": you're in a good mood, and it actually
   works for you. Angels sing, and a light suddenly fills the room.
 - "g*dd*mn idiotic truckload of sh*t": when it breaks
```

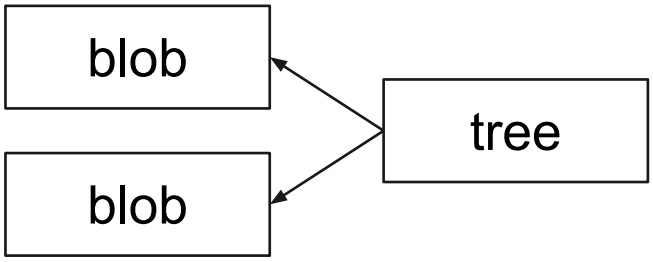
Object database

blob

```
$ git cat-file -p d670460b4b4aece5915caf5c68d12f560a9fe3e4
test content
```

Data identified by its content hash

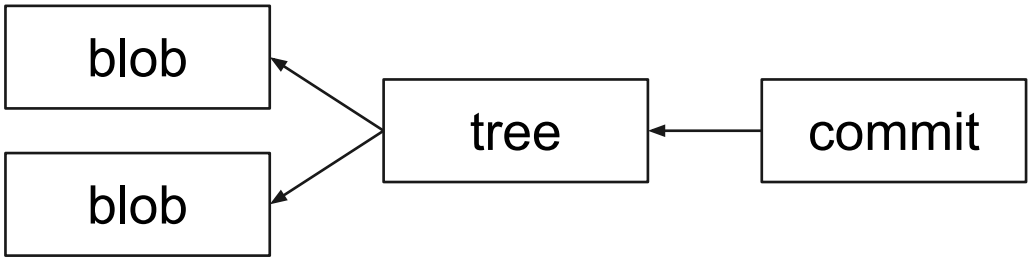
Object database




```
$ git cat-file -p master^{tree}
100644 blob a906cb2a4a904a152e80877d4088654daad0c859      README
100644 blob 8f94139338f9404f26296befa88755fc2598c289      Rakefile
040000 tree 99f1a6d12cb4b6f19c8655fca46c3ecf317074e0      lib
```

DAG

Object database




```
$ git cat-file -p fdf4fc3
tree d8329fc1cc938780ffdd9f94e0d364e0ea74f579
author Scott Chacon <schacon@gmail.com> 1243040974 -0700
committer Scott Chacon <schacon@gmail.com> 1243040974 -0700

first commit
```

That’s the basics

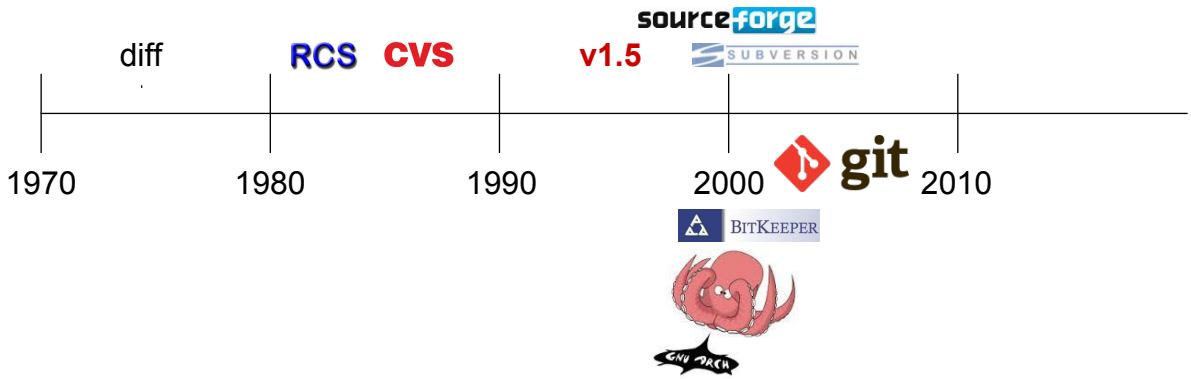
DAG represented by a content addressed object database. Took Linux a day

Git v0.0 usage

```
$ init-db
$ edit file
$ update-cache --add file
$ edit file
$ show-diff
$ update-cache file
$ T=$(write-tree)
$ P=$(cat .dircache/HEAD)
$ C=$(echo "My commit" |
      commit-tree $T -p $P)
$ echo $C >.dircache/HEAD
```

git commit

History


diff
RCS
CVS
v1.5
sourceforge
SUBVERSION
1970
1980
1990
2000
2010
git
BitKeeper
GNU ARCH

Plumbing and porcelain

Two types of commands:
Porcelain is user-facing commands
Plumbing is low-level internals

No formal separation
Git v0 had no porcelain (but see Cogito)

Git 1.5 released

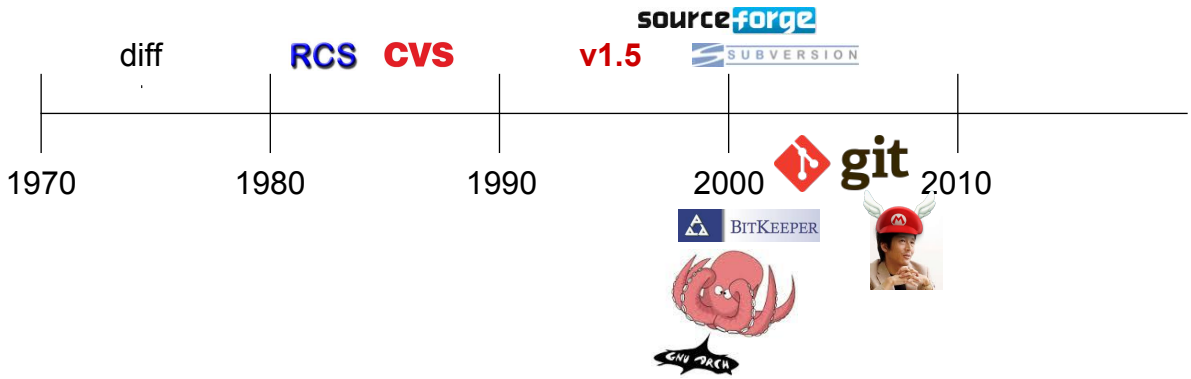
But not by Linus.

Git 1.5 considered by many to be the first usable release (vastly improved porcelain)

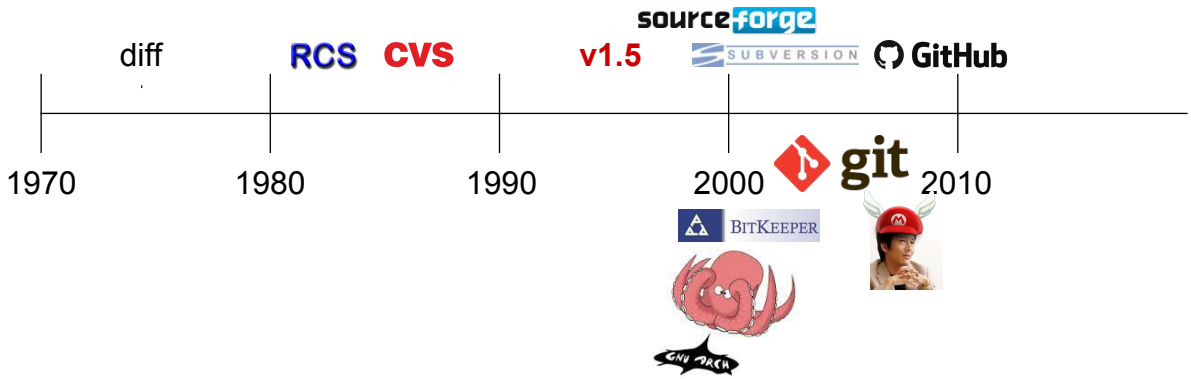
Linus contributed performance, excellent plumbing. Junio made a lot of headway in usability and porcelain

http://comments.gmane.org/gmane.comp.version-control.git/6475

History


diff
RCS
CVS
v1.5
sourceforge
SUBVERSION
1970
1980
1990
2000
git
2010
BitKeeper
GNU ARCH

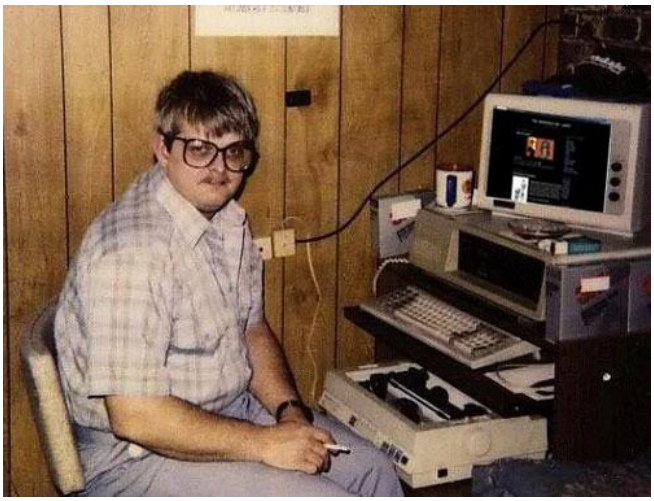
History



2009. Inflection point to full on tipping point

Not officially associated with Git. Free for open source projects

Open source zero

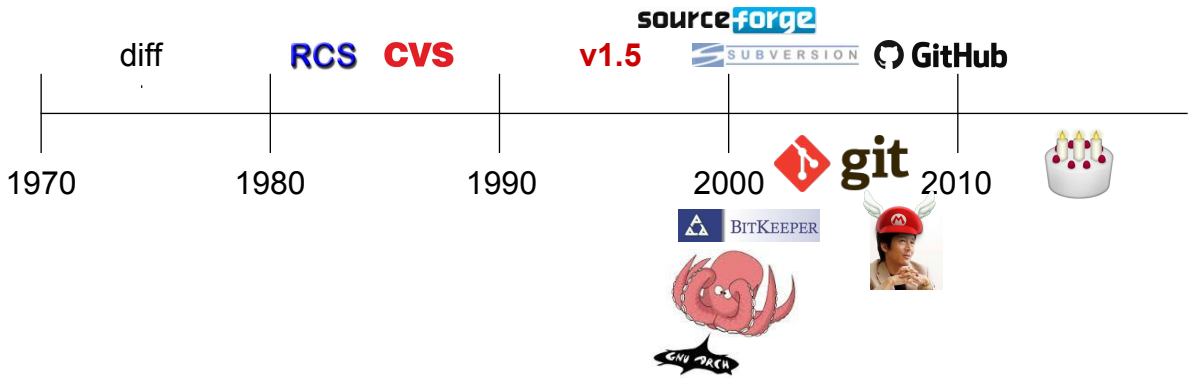


from stodgy basements (e-mail patches) to sexy garages

Open source hero

open source went from the exception to the norm. This guy is the everyman

History



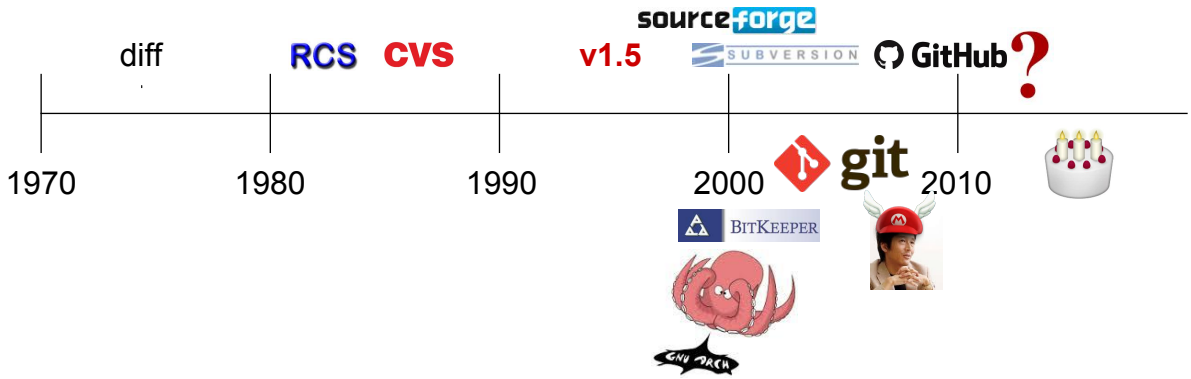
April 2015. 10 years!
The most popular VCS
One of the most widely used software engineering tools
Linus' contribution to Git has had as big an impact as Linux

[Git Chronicle]

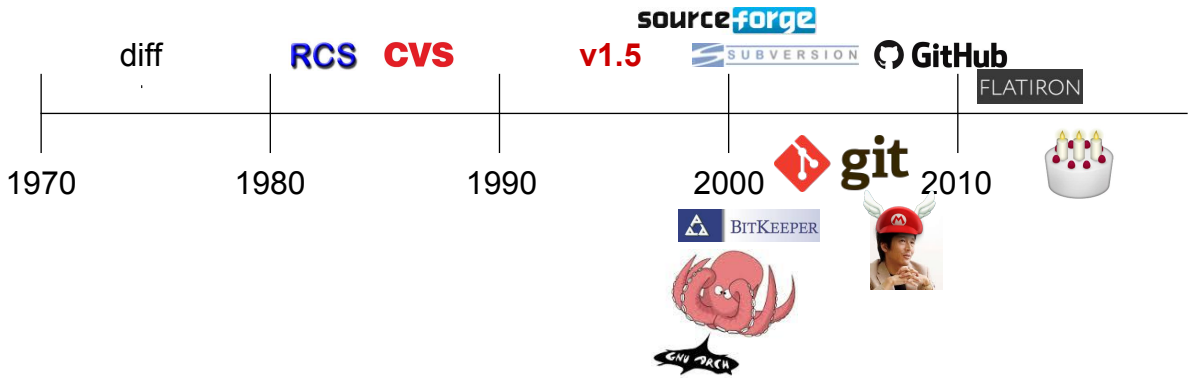
[10 Years of Git]

[Evolution of Version Control in Open Source]

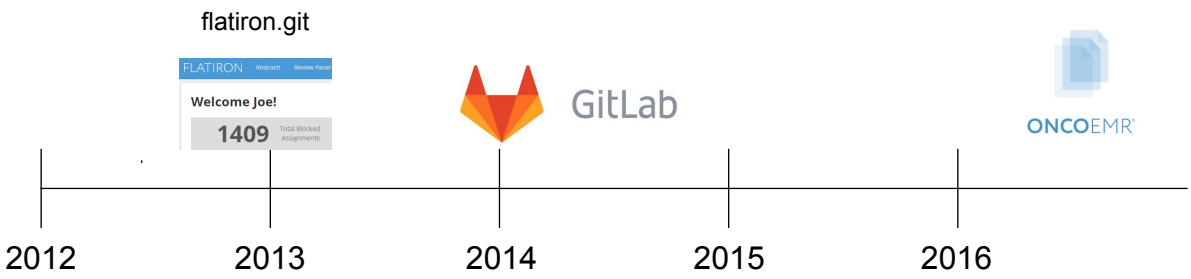
History


diff
RCS
CVS
v1.5
sourceforge
SUBVERSION
GitHub
?
1970
1980
1990
2000
2010
git
BitKeeper
GNU ARCH

History


diff
RCS
CVS
v1.5
sourceforge
SUBVERSION
GitHub
FLATIRON
1970
1980
1990
2000
2010
git
BitKeeper
GNU ARCH

History at Flatiron



GitHub post-mortem - https://docs.google.com/document/d/1qik7tmZmKweL0YoNTHfxjfYtP8TjTuE44L8FnKFRPcc/edit

Git like a startup

- Minimum viable product (Lean Startup)
- Perfect is the enemy of good
- Nothing is precious

Git internals are rife with inconsistencies (stash reflog, submodules)
Porcelains abandoned, natural selection of code

Rapid prototyping
Resourcefulness

Git not like a startup

- Not user-oriented
- Development in the open
- No attempt to get big

No A/B tests. UI an afterthought; still many inconsistencies
No stealth mode. Co-maintainers
Built to further their own ends. No monetization

TODO write a conclusion

- Share a passion
- What's *our* git?

Spark (DJ), React (JoeD), Celery (Paul / James), Fortigate (Ben)

Leftovers

contrib/

Documentation/ (esp. api)

git-bigfiles

code fearlessly

mercurial

Leftovers

subcommands. polyglot.
8e49d50388211a0f3e7286f6ee600bf7736f4814
. 215a7ad1ef790467a4cd3f0dcffbd6e5f04c38f7

git stash. git log (composition). git rebase

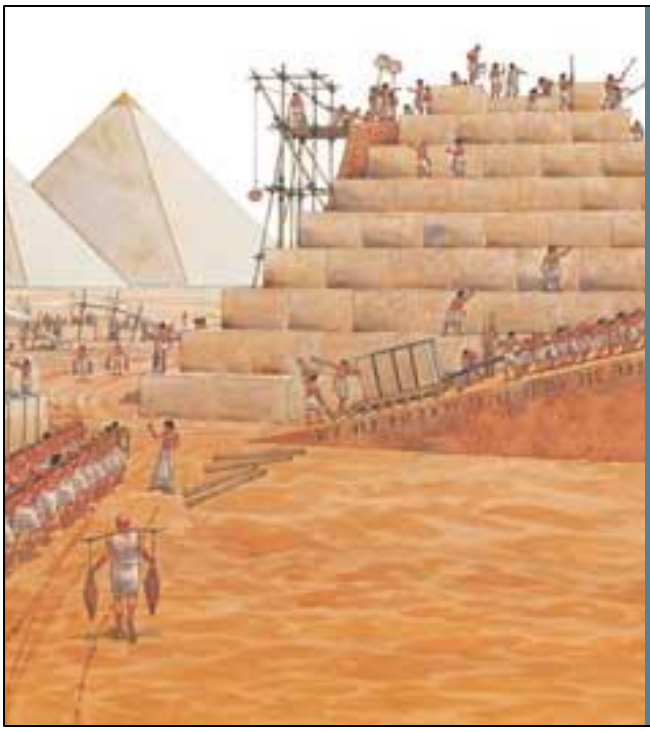
Leftovers

refs

reflog

rebase DAG

merge commits



Forking B.G.E.

Forking G.E.

Just build one pyramid! Straight out of science fiction

Distributed version control

Optimizes for different workflows:

- Decentralized development
  - Lots of “forks”
- Lots of merges
- Operations against history

Notably absent is stuff about writing actual code
Who would have a workflow like this?

Falling out with BitKeeper

Let the flames begin.

Linus

PS. Don't bother telling me about subversion.

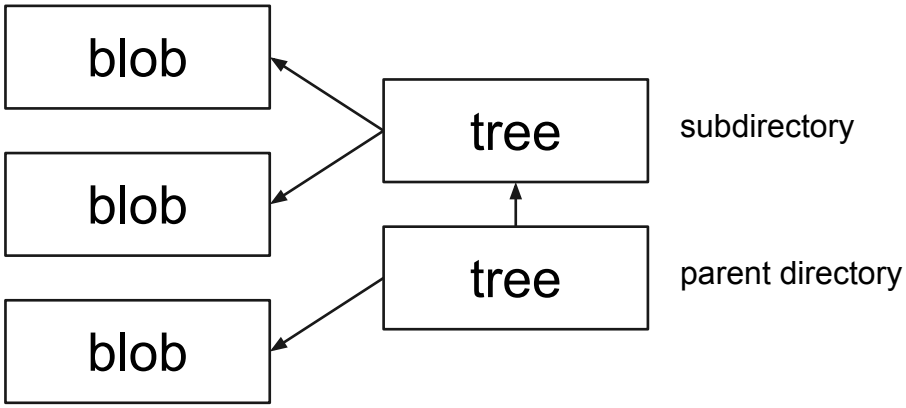
Object database

- Stored as files
  - .git/objects
- Content addressable
  - SHA1
  - Deduplication (compression)
  - Simple graph representation

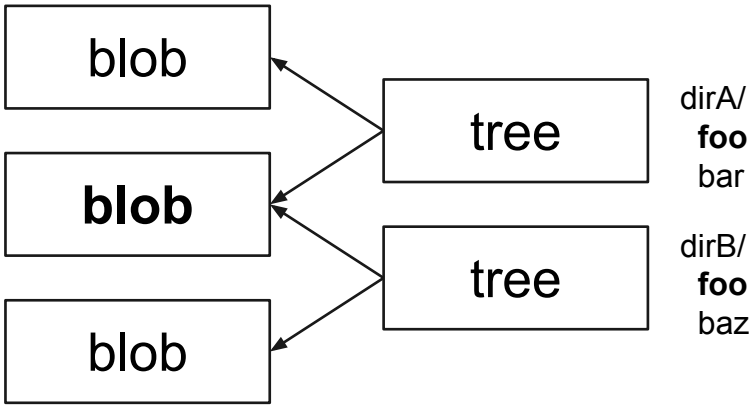
Varnish Notes from the Architect

Linus hates databases
Special key-value store

Object database

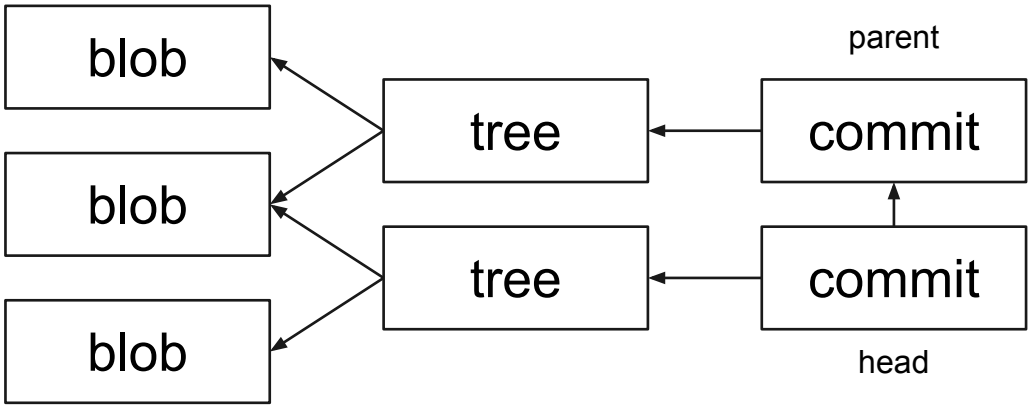


Object database



Similar to COW

Object database



Plumbing and porcelain

Two types of commands:

- Porcelain is user-facing commands
- Plumbing is low-level internals

No formal separation

Git v0 had no porcelain (but see Cogito)

Low-/high-level interfaces

Most porcelains were short-lived, but they all contributed some idea

Very low-level tool. Sometimes used for ad hoc storage of revisions (same thing I said about diff)

History

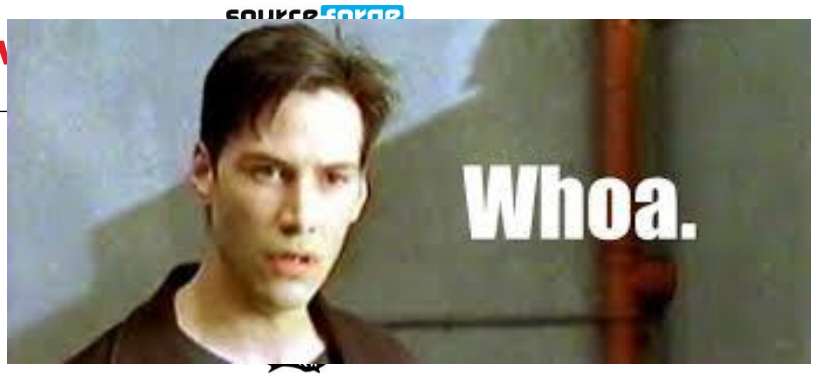
diff

RCS

CV

1970

1980



Git Tech Talk @Google

Good timing, Git 1.5 recently released

https://www.youtube.com/watch?v=4XpnKHJAok8

Happy birthday!

Git Chronicle

10 Years of Git

Evolution of Version Control in Open Source

Object database

Three main types:

- blob (file content)
- tree (directory)
- commit (change set)

Git Internals
Git for Computer Scientists
Fast Intro to Git Internals

Index

- Staging area to build a tree object
  - For the next commit
- Performance optimization
  - Binary representation for fast operations against working tree

Vast majority of today's users only care about the first
Much confused topic. In some ways an artifact of the past