Census of public source code history

Original Author: Audris Mockus
Presenters: John Sadik, Parker Collier, Anthony Roman

Outline

- Background
- Motivation
- Methodology
- Results

Background - Version Control Systems

VCS allows users to track changes in a code base, these changes are marked with an amount of metadata like comments, author name, and time of change.

Multiple users can develop a single code base concurrently.

Background - Example

Code Before

```
int i = n;

while (i−−)

printf (” %d”, i);
```

Code After

```
//print n integers iff n≥
0

int i = n;

while (−−i > 0)

printf (” %d”, i);
```

Changes

one line deleted

two lines added

two lines unchanged

Other attributes: date: 2014-05-29 01:25:30, developer id: audris, branch: master, Comment: “*Fix bug 3987 - infinite loop if n ≤ 0*”

Background

Allows for local code development, often a critical requirement.

There are many frameworks such as SCCS, CVS, ClearCase, SVN, Bzr, Hg, and Git.

VCS is almost universally adopted for software development.

Motivation - Why Public Source Code

It allows for comparison between different groups of software project and technologies, the same information will be recorded similarly across all version control systems.

A census on the current state of public code informs us on the current trends in software development.

- This is useful for large bodies like businesses and governments to better grasp the current state of the industry.

Motivation - Why Code

Code is functional knowledge.

- it can be analyzed quickly and precisely by a computer, rather than manually like many scholarly or literary works

Open source code.

- Large collaborative efforts allow for many people to contribute their knowledge and is representative of current innovation and interest.
- Legacy systems contain knowledge from millions of man hours, and have real discussion and documentation attached to them.

Motivation - Why Global Properties of Code

We need to understand the contextual information about code.

- How much has the code been reused? Is this is original author?
- what trends and technology are being used?

A full population is needed to gather this information. Cloud based VCS systems allow for the collection of all repositories on their platform.

Motivation - Why Now?

VCS systems are at an all time high for popularity, especially git based systems.

- large and comprehensive data sets can be pulled from the most popular cloud based VCS systems
  - i.e. Bitbucket, Github, and Gitlab

Methodology - Overview

1. Find VCS repositories
2. Copy or clone repositories
3. Gather all the metadata (i.e. commits, authors, dates, comments)
4. Extract and index all versions of each file
5. Establish links across repositories to create Universal Version History

Methodology - Universal Version History

- Code files are often copied and reused
- This can lead to code being counted and analyzed multiple times
  - To avoid this problem, the authors construct a “Universal Version History”
- To establish this Universal Version History they establish links between files
  - Based on identical content and similar content

Methodology - Discovery and Automation

- Look for sites with many projects (e.g. GitHub), ecosystems (e.g. Mozilla), famous projects (e.g. Mysql), projects in wide use, or published surveys of projects
- Use a spider for automation
  - Use a search engine and grab project home directory URLs
  - Search for VCS-specific URL patterns (e.g. git[:.])

Results - 2010

Forge	Type	VCSs	Files	File/Ver.	Uniq.	Space	From
Large cmpny.	Var.	>200	3,272K	12,585K	4,293K	remote	1988
SourceForge	CVS	121K	26,095K	81,239K	39,550K	820GB	1998
code.google	SVN	42K	5,675K	14,368K	8,584K	remote	1996
github.com	Git	29K	5,694K	18,986K	7,076K	154GB	1988
repo.or.cz	Git	1,867	2,519K	11,068K	5,115K	43GB	1986
gitorious.org	Git	1,098	1,229K	4,896K	1,749K	20GB	1988
Debian	Git	1,662	1,058K	4,741K	1,863K	19GB	1988
Savannah	CVS	2,946	852K	3,623K	2,345K	25GB	1985
objectweb.org	CVS	93	1,778K	2,287K	528K	17GB	1999
SourceWare	CVS	65	213K	1,459K	761K	10GB	1989
netbeans	Hg	57	185K	23,847K	492K	69GB	1999
rubyforge.org	SVN	3,825	456K	807K	256K	4.9GB	2001
Mozilla	Hg	14	58K	210K	105K	1.6GB	2000

Results - 2010 cont.

Forge	Type	VCSs	Files	File/Ver.	Unique F/V	Space	From
git.kernel.org	Git	595	12,974K	97,585K	856K	205GB	1988
OpenSolaris	Hg	98	77K	1,108K	91K	9.7GB	2003
FreeBSD	CVS	1	196K	360K	75K	2.5GB	1993
Kde	SVN	1	2,645K	10,162K	527K	50GB	1997
gnome.org	SVN	566	1,284K	3,981K	1,412K	1GB	1997
Freedesktop	CVS	75	139K	784K	375K	4GB	1994
Gcc	SVN	1	3,758K	4,803K	395K	14GB	1989
Eclipse	CVS	9	729K	2,127K	575K	11GB	2001
OpenJDK	Hg	392	32K	747K	60K	15GB	2008
Mysql-Server	Bazaar	1	10K	523K	133K	6GB	2000
PostgreSQL	CVS	1	6K	108K	105K	0.5GB	1994
ruby-lang	SVN	1	163K	271K	56K	0.6GB	1998
Perl	Git	1	11,539	103K	42K	0.2GB	1988
Python	SVN	1	8K	89K	76,454	0.8GB	1991

Results - 2015

- 30+ million projects on GitHub
  - Doubled from 2014
- 0.7 million projects on BitBucket
  - Increased 80% from 2014

Example Applications

- Software Supply Chain
  - Tracks down vulnerable code to reduce risks
- Provide software engineering research base
  - Code/Expertise Search
  - Automatic documentation
  - Risk Models

Challenges and Needs

- Challenges
  - Operational data are treacherous
  - Continuously changing
  - Measuring or defining accuracy
  - Potential for misinterpretation
- Hardware Needs
  - Constructing and analyzing large graphs
  - Storage requirements (1PB)
- Data Approaches
  - Contextualization
  - Augmentation
  - Correction