Software Heritage: why and how
Building the Universal Archive of Source Code

Roberto Di Cosmo
roberto@dicosmo.org
June 7th, 2018
1. Software is everywhere and nowhere
2. Software source code in Science
3. Are we loosing trace of our knowledge?
4. The Software Heritage initiative
5. Using the Software Heritage archive
6. Building for the long term
7. Conclusion
Source code is executable and human readable knowledge

a growing part of our Cultural Heritage
Software source code is *special*

Harold Abelson, Structure and Interpretation of Computer Programs

“Programs must be written for people to read, and only incidentally for machines to execute.”

Quake 2 source code (excerpt)

```c
float Q_rsqrt( float number )
{
    long i;
    float x2, y;
    const float threehalves = 1.5F;
    x2 = number * 0.5F;
    y = number;
    i = * ( long * ) &y; // evil floating point bit level hacking
    i = 0x5f3759df - ( i >> 1 ); // what the fuck?
    y = * ( float * ) &i;
    y = y * ( threehalves - ( x2 * y + y ) ); // 1st iteration
    // y = y * ( threehalves - ( x2 * y + y ) ); // 2nd iteration, this
    // can be removed
    return y;
}
```

Net. queue in Linux (excerpt)

```c
/*
 * SFB uses two B[i][n] : L X N arrays of bins (L levels, N bins per level)
 * This implementation uses L = 8 and N = 16
 * This permits us to split one 32bit hash (provided per packet by rxhash or
 * external classifier) into 8 subhashes of 4 bits.
 */
#define SFB_BUCKET_SHIFT 4
#define SFB_NUMBUCKETS (1 << SFB_BUCKET_SHIFT) /* N bins per Level */
#define SFB_BUCKET_MASK (SFB_NUMBUCKETS - 1)
#define SFB_LEVELS (32 / SFB_BUCKET_SHIFT) /* L */

/* SFB algo uses a virtual queue, named "bin" */
struct sfb_bucket {
    u16 qlen; /* length of virtual queue */
    u16 p_mark; /* marking probability */
};
```

Len Shustek, Computer History Museum

“Source code provides a view into the mind of the designer.”
~ 50 years, a lightning fast growth

Apollo 11 Guidance Computer (~60,000 lines), 1969

"When I first got into it, nobody knew what it was that we were doing. It was like the Wild West."

Margaret Hamilton

Linux Kernel

... now in your pockets!

are we taking care of all this?
Software is spread all around
Software is fragile
Software lacks its own research infrastructure

Photo: ALMA(ESO/NAOJ/NRAO), R. Hills
Outline

1. Software is everywhere and nowhere
2. Software source code in Science
3. Are we loosing trace of our knowledge?
4. The Software Heritage initiative
5. Using the Software Heritage archive
6. Building for the long term
7. Conclusion
Collberg’s report from the trenches

Analysis of 613 papers
- 8 ACM conferences: ASPLOS’12, CCS’12, OOPSLA’12, OSDI’12, PLDI’12, SIGMOD’12, SOSP’11, VLDB’12
- 5 journals: TACO’9, TISSEC’15, TOCS’30, TODS’37, TOPLAS’34

all very practical oriented

The basic question

Can we get the code to build and run?
The result

This can be debated (see http://cs.brown.edu/~sk/Memos/Examining-Reproducibility/), but...

... that’s a whopping 81% of non reproducible works!
Pressure to make research code available is now raising

Evaluation of software artefacts (optional)

- tools are usable, in line with expectations
- started as a contest in 2011 (ESEC/FSE) (winner Vouillon and Di Cosmo)
- now going mainstream: POPL’17, POPL’16, ECOOP’16, OOPSLA’16, CGO’16, VISSOFT’16, PLDI’16, CGO’15, PPoPP’15, VISSOFT’15, ISSTA’15, OOPSLA’15, PLDI’15, POPL’15, CAV’15, ECOOP’15, FSE’15, ISSTA’14, OOPSLA’14, PLDI’14, ECOOP’14, FSE’14, SAS’13, OOPSLA’13, ECOOP’13, FSE’13, FSE’11
Some people claim that having (all) the source of the code used in an experiment is not worth the effort (see “Replicability is not Reproducibility: Nor is it Good Science”, Chris Drummond, ICML 2009)

Sure, diversity is important, but:

- Source code is like the proof used in a theorem: can we really accept Fermat statements like “the details are omitted due to lack of space”?
- modern complex systems makes even the simplest experiment depend on a wealth of components and configuration options
- access to all the source code is not just necessary to reproduce, it is also useful to evolve and modify, to build new experiments from the old ones
Outline

1. Software is everywhere and nowhere
2. Software source code in Science
3. Are we loosing trace of our knowledge?
4. The Software Heritage initiative
5. Using the Software Heritage archive
6. Building for the long term
7. Conclusion
URL decay disrupts the web of reference

Web links are not permanent (even permalinks)

there is no general guarantee that a URL... which at one time points to a given object continues to do so

URLs used in articles decay!

Analysis of IEEE Computer (Computer), and the Communications of the ACM (CACM): 1995–1999
  - the half-life of a referenced URL is approximately 4 years from its publication date
D. Spinellis. The Decay and Failures of URL References.

How Do Astronomers Share Data?

Pepe, Goodman, Muench, Crosas, Erdmann

dx.doi.org/10.1371/journal.pone.0104798

PLOS August 28, 2014
Example: doi:10.1109/MSR.2015.10

- to find what 10.1109/MSR.2015.10 is, go to a resolver (e.g. doi.org)
- this returns http://ieeexplore.ieee.org/document/7180064/
- at this URL we find ...

Architecture of the DOI infrastructure

- DOI resolution can change
- content at URL can change
- no intrinsic way of noticing
- persistence based on good will of multiple parties
No catalog, no archive, no references, . . . and we are at a turning point

Looking at the past

- a lot of old software misplaced, lost, or behind barriers, but . . .
- most founding fathers are still here, and willing to share
- urgent to collect their knowledge

Only a few years left.

Looking at the future

- software development and use skyrockets: more programmers, and more code!
- essential to provide a universal platform for all the future software source code

Every year that goes by makes the problem worse.

it is urgent to take action!
1. Software is everywhere and nowhere
2. Software source code in Science
3. Are we losing trace of our knowledge?
4. The Software Heritage initiative
5. Using the Software Heritage archive
6. Building for the long term
7. Conclusion
Software Heritage

Our mission
Collect, preserve and share the source code of all the software that is available

Past, present and future
Preserving the past, enhancing the present, preparing the future
A principled infrastructure

Technology
- transparency and FOSS
- replicas all the way down

Content
- intrinsic identifiers
- facts and provenance

Organization
- non-profit
- multi-stakeholder

Software Heritage

Source files: 4,536,067,027
Commits: 1,024,675,748
Projects: 83,801,775
All the source code
All the source code: strategy

- Embargo
- Automation
- Focused Search
- Crowdsourcing

Online
Offline
Closed
Open
- full development history permanently archived
- origins: GitHub (automated), Debian (automated), Gitorious, Google Code, GNU
- ~200Tb raw contents, ~10Tb graph (7+Bn nodes, 60+Bn edges)
Merkle tree (R. C. Merkle, Crypto 1979)

Combination of
- tree
- hash function

Classical cryptographic construction
- fast, parallel signature of large data structures
- widely used (e.g., Git, blockchains, IPFS, …)
- built-in deduplication
Outline

1. Software is everywhere and nowhere
2. Software source code in Science
3. Are we loosing trace of our knowledge?
4. The Software Heritage initiative
5. Using the Software Heritage archive
6. Building for the long term
7. Conclusion
Reference archive for all software

A "wayback machine" for software source code … and intrinsic identifiers!


Demo time: let’s highlight some features…

Origin search

Directory browsing

Revisions as diffs
A glimpse at the technical roadmap

Features...

- (done) **lookup** by content hash
- **browsing**: "wayback machine" for archived code
  - (done) [http://archive.softwareheritage.org/api](http://archive.softwareheritage.org/api)
  - (done) [http://archive.softwareheritage.org/browse/search](http://archive.softwareheritage.org/browse/search)
- (done) **download**: `wget` / `git clone` from the archive
- (done) **deposit** of source code bundles directly to the archive
- (todo) **provenance** lookup for all archived content
- (todo) **full-text search** on all archived source code files

... and much more ...

you have the world’s software development graph at your hands!

your tools could be here!

Roberto Di Cosmo

www.dicosmo.org
Demo links

Paper points to lost source code on gitorious

- https://www.openaire.eu/search/publication?articleId=dedup wf_001::cd996f0b6236b90659f84f99feb62bcc
- https://gitorious.org/parmap
- https://archive.softwareheritage.org/browse/search/?url=%22gitorious.org/parmap%22
Supporting more accessible and reproducible science

A global library referencing all software used in all research fields

- completes the infrastructure for Open Access in science
- provides intrinsic persistent identifiers needed for scientific reproducibility
- enables large scale, verifiable software studies
Deposit Scientific Software

Deposit software in HAL


Generic mechanism:
- SWORD based
- review process
- versioning

How to do it:
- today: deposit .zip file
- tomorrow:
  - provide SWH id and metadata
  - provide SWH id, metadata is extracted
  - ...

Intrinsic PIDS for referencing content now available

see http://bit.ly/swhpids and the forthcoming iPres 2018 article
The way to go to archive and reference scientific software

All features of Software Heritage for free

- **intrinsic IDs** (integrity, not just DIOs!), browse, download (now)
- metadata, licenses, provenance analysis (plagiarism detection), classification (wip)
- and many more (powerful connections with SE and Industry)

Coverage and uniformity

- **one** archive for **all** domains (industry included)
- you can reference **any** software, not just the deposited one
  
  *thanks D. Katz for pointing this out*
- **git-compatible** identifiers greatly simplify workflows

Sustainability

**one** infrastructure  **independent** non profit foundation  **worldwide** mirrors

... doors are open!
Big Code = Big data + AI

Large scale *repeatable* software studies…

- vulnerability detection
- dependency analysis
- pattern elicitation
- automatic classification …

… need a uniform representation

Software Heritage has **one data model** for all forges/VCS…

… yes, we do **data normalization** of software evolutiona!

Coming soon to a platform near you!
Outline

1. Software is everywhere and nowhere
2. Software source code in Science
3. Are we loosing trace of our knowledge?
4. The Software Heritage initiative
5. Using the Software Heritage archive
6. Building for the long term
7. Conclusion
Growing Support

Landmark Inria Unesco agreement, April 3rd, 2017

Sharing the vision

Contributing to the mission

Roberto Di Cosmo  www.dicosmo.org
The next steps

The Software Heritage Foundation
- independent
- long term mission
- multistakeholder

The community
- academia: Open Access, research
- industry: better software
- cultural heritage: all the software history

The mirror network
- resilience
- biodiversity

“Let us save what remains: not by vaults and locks which fence them from the public eye and use in consigning them to the waste of time, but by such a multiplication of copies, as shall place them beyond the reach of accident.”

Thomas Jefferson
You can help!

Take the research challenges
- efficient tracking of development streams
- machine learning/classification
- ...

Contribute to the development see http://forge.softwareheritage.org

★★ listers/loaders for other unsupported forges, VCS
★★ Web UI improvements

Funding
- pester companies to become sponsors: sponsorship.softwareheritage.org
- give your own contribution: www.softwareheritage.org/donate

Spread the word!
- help research teams use the archive
- tell everybody about Software Heritage

Roberto Di Cosmo www.dicosmo.org
Outline

1. Software is everywhere and nowhere
2. Software source code in Science
3. Are we loosing trace of our knowledge?
4. The Software Heritage initiative
5. Using the Software Heritage archive
6. Building for the long term
7. Conclusion
Software Heritage

www.softwareheritage.org  @swheritage
Grand opening, June 7th, UNESCO headquarters!

Library of Alexandria of code
- recover the past
- structure the future

A CERN for Software
- build better software
  - for industry
  - for society as a whole