Software Heritage
Building the Universal Software Archive for Open Science

Roberto Di Cosmo
roberto@dicosmo.org
May 14th, 2018
1. Software is everywhere around us
2. Software source code for Science!
3. The Software Heritage initiative
4. Status
5. Building for the long term
Software is Pervasive

At the heart of *our society*
- communication, entertainment
- administration, finance
- health, energy, transportation
- education, research, politics
- ... 

At the heart of *technology*
- house appliances $\approx 10M$ SLOC
- phones $\approx 20M$ SLOC, *cars* $\approx 100M$ SLOC
- Internet of things, ...
Software is Knowledge

Key mediator for accessing all information (c) Banski

Information is a main pillar of our modern societies.

Absent an ability to correctly interpret digital information, we are left with [...] "rotting bits" [...] of no value.

Vinton G. Cerf IEEE 2011

Software is an essential component of modern scientific research

[...] the vast majority describe experimental methods or software that have become essential in their fields.

Top 100 papers (Nature, October 2014)

Software embodies our Knowledge and Cultural Heritage
"The source code for a work means the preferred form of the work for making modifications to it."

— GPL Licence

Hello World

**Program (source code)**

```c
/* Hello World program */

#include<stdio.h>

void main()
{
    printf("Hello World");
}
```

**Program (excerpt of binary)**

```
4004e6: 55
4004e7: 48 89 e5
4004ea: bf 84 05 40 00
4004ef: b8 00 00 00 00
4004f4: e8 c7 fe ff ff
4004f9: 90
4004fa: 5d
4004fb: c3
```
Software Source Code is *special*

Harold Abelson, Structure and Interpretation of Computer Programs (1st ed.) 1985

“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 threerad = 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 * ( threerad - ( x2 * y * y ) ); // 1st iteration
  // y = y * ( threerad - ( x2 * y * y ) ); // 2nd iteration, this
  // can be removed
  return y;
}
```

Net. queue in Linux (excerpt)

```c
/*
 * SFB uses two 8[N][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.”
"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

Fashion victims

- many disparate development platforms
- a myriad places where distribution may happen
- projects tend to migrate from one place to another over time

Where is the place …

where we can find, track and search *all* source code?
Like all digital information, FOSS is fragile

- inconsiderate and/or malicious code loss (e.g., Code Spaces)
- business-driven code loss (e.g., Gitorious, Google Code)
- for obsolete code: physical media decay (data rot)

Where is the archive...

where we go if (a repository on) GitHub or GitLab.com goes away?
Outline

1. Software is everywhere around us
2. Software source code for Science!
3. The Software Heritage initiative
4. Status
5. Building for the long term
We face a science crisis

"Sub-prime science"? (Nicholas Humprey)
- inconsistencies
- data corruption, fraud
- non reproducible findings… (picture from Nature, Sep. 2015)

The world starts noticing

October 2013

John Oliver, *Science* May 2016
How we built our scientific knowledge

The experimental method

- make an observation
- formulate an hypothesis
- set up an experiment
- formulate a theory

And then we reproduce and verify.

Reproducibility is the key

*non-reproducible single occurrences are of no significance to science*

Karl Popper, *The Logic of Scientific Discovery*, 1934
Reproducibility in the digital age

For an experiment involving software, we need

- open access to the scientific article describing it
- open data sets used in the experiment
- source code of all the components
- environment of execution
- stable references between all this

Remark

The first two items are already widely discussed!

... what about software?
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 workflow
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!
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


Scholar roster of broken links

An example from Astronomy

<table>
<thead>
<tr>
<th>Domain</th>
<th>links (broken)</th>
<th>.html</th>
<th>.txt</th>
<th>.dat</th>
<th>.gz</th>
<th>.tar</th>
<th>.fits</th>
<th>tilde</th>
</tr>
</thead>
<tbody>
<tr>
<td>cs.harvard.edu</td>
<td>802 (110)</td>
<td>336 (70)</td>
<td>0</td>
<td>0</td>
<td>4 (2)</td>
<td>5 (4)</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>hescan.gsfc.nasa.gov</td>
<td>640 (33)</td>
<td>423 (27)</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www.stsci.edu">www.stsci.edu</a></td>
<td>498 (61)</td>
<td>205 (29)</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>15 (10)</td>
</tr>
<tr>
<td>ascii.harvard.edu</td>
<td>471 (152)</td>
<td>212 (99)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 (1)</td>
</tr>
<tr>
<td>nsc.ipac.caltech.edu</td>
<td>427 (194)</td>
<td>135 (76)</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>cfa-www.harvard.edu</td>
<td>352 (68)</td>
<td>277 (52)</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>54 (17)</td>
</tr>
<tr>
<td>archive.stsci.edu</td>
<td>336 (58)</td>
<td>57 (39)</td>
<td>2</td>
<td>1 (1)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www.ipac.caltech.edu">www.ipac.caltech.edu</a></td>
<td>285 (14)</td>
<td>209 (12)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www.mnf.caio.eu">www.mnf.caio.eu</a></td>
<td>211 (21)</td>
<td>12 (6)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7 (5)</td>
</tr>
<tr>
<td>space.mit.edu</td>
<td>193 (10)</td>
<td>50 (5)</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2 (1)</td>
</tr>
<tr>
<td><a href="http://www.astrosipia.edu">www.astrosipia.edu</a></td>
<td>186 (4)</td>
<td>103 (1)</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td><a href="http://www.nesc.org">www.nesc.org</a></td>
<td>186 (38)</td>
<td>54 (22)</td>
<td>1 (1)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4 (1)</td>
</tr>
<tr>
<td>nrao.ipac.caltech.edu</td>
<td>163 (5)</td>
<td>38</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www.seds.org">www.seds.org</a></td>
<td>156 (2)</td>
<td>106 (1)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>kea-www.harvard.edu</td>
<td>125 (37)</td>
<td>42 (17)</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>26 (16)</td>
</tr>
<tr>
<td>physics.nist.gov</td>
<td>125 (3)</td>
<td>63 (2)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www.nasa.gov">www.nasa.gov</a></td>
<td>120 (3)</td>
<td>50 (2)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www.uspaeus.es">www.uspaeus.es</a></td>
<td>118 (35)</td>
<td>23 (19)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 (1)</td>
<td>0</td>
<td>1 (1)</td>
</tr>
<tr>
<td><a href="http://www.astroph.princeton.edu">www.astroph.princeton.edu</a></td>
<td>115 (31)</td>
<td>43 (14)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>53 (12)</td>
</tr>
<tr>
<td>astro.nasa.mil</td>
<td>110 (27)</td>
<td>98 (22)</td>
<td>3 (3)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 (1)</td>
</tr>
</tbody>
</table>

This table lists total number of links and broken links (HTTP status codes 3xx, 4xx, and 5xx) to top domains (domains with over 100 links) found within articles published in the four main astronomy journals between 1997 and 2008. The table also shows, for each domain, the portion of links to common filename extensions, as well as links that contain the tilde character.

doi:10.1371/journal.pone.0104798

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 around us
2. Software source code for Science!
3. The Software Heritage initiative
4. Status
5. Building for the long term
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
Outline

1. Software is everywhere around us
2. Software source code for Science!
3. The Software Heritage initiative
4. Status
5. Building for the long term
Current sources

- **live**: GitHub, Debian
- **one-off**: Gitorious, Google Code, GNU
- **WIP**: Bitbucket

150 TB blobs, 5 TB database (as a graph: 7 B nodes + 60 B edges)

The *richest* public source code archive, … and growing daily!
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
Demo time (breaking news!)

**Browsing the archive contents**
- archive.softwareheritage.org

**Archiving scientific software via HAL**
- open on the Inria instance, see the deposit guide at http://bit.ly/swhdeposithalen
1. Software is everywhere around us
2. Software source code for Science!
3. The Software Heritage initiative
4. Status
5. Building for the long term
Growing Support

Landmark Inria Unesco agreement, April 3rd, 2017

 Sharing the vision

 Contributing to the mission

Roberto Di Cosmo www.dicosmo.org

Microsoft www.softwareheritage.org May 14th, 2018

>= 100K€/year

>= 50K€/year

>= 25K€/year

>= 10K€/year

INRIA INVENTEURS DU MONDE NUMÉRIQUE

SOFTWARE

 >= 100K€/year

 >= 50K€/year

 >= 25K€/year

 >= 10K€/year

SOFTWARE

 >= 100K€/year

 >= 50K€/year

 >= 25K€/year

 >= 10K€/year

SOFTWARE

 >= 100K€/year

 >= 50K€/year

 >= 25K€/year

 >= 10K€/year

SOFTWARE

 >= 100K€/year

 >= 50K€/year

 >= 25K€/year

 >= 10K€/year

SOFTWARE

 >= 100K€/year

 >= 50K€/year

 >= 25K€/year

 >= 10K€/year

SOFTWARE

 >= 100K€/year

 >= 50K€/year

 >= 25K€/year

 >= 10K€/year

SOFTWARE
The next steps

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

The community
- academia: Open Access, research
- industry: better software
- cultural heritage: 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
An unique opportunity

Library of Alexandria of code

Take *urgent* action to
- recover the past
  - founding fathers still here
- structure the future
  - programming skyrockets

A CERN for Software

Build a *common infrastructure*
- supporting industry needs
- enabling software research
- fostering better science
- for society as a whole

Photo: ALMA(ESO/NAOJ/NRAO), R. Hills
Come in, we’re open, and you can help!

Software Heritage

www.softwareheritage.org  @swheritage

Support the effort, get involved!

- partnerships, mirrors
  mailto:roberto@dicosmo.org
- sponsoring
  sponsorship.softwareheritage.org
- donations
  www.softwareheritage.org/donate
- our own code
  forge.softwareheritage.org
Outline

6 Collection strategy

7 Selected research challenges: building the archive

8 Selected research challenges: using the archive
All the source code
All the source code, strategies
Online, open source code: automation overview

[Diagram showing software origins, Forges, Distros, Package repos, Scheduling, Listing, Loading & deduplication, Merkle DAG, blob storage, Debian source package loader, tar loader, Git loader, Mercurial loader, SVN, hg, git, dsc, zip, software origins.]

Roberto Di Cosmo
www.dicosmo.org

May 14th, 2018
Outline

6 Collection strategy

7 Selected research challenges: building the archive

8 Selected research challenges: using the archive
Data compression

Deduplication is performed at the file level *across all projects in the world*

**Pros**
- very efficient to cope with file clones
- quite resilient to technology changes

**Cons**
- a minor edit creates two different files

**Challenge: exploit file similarities**
- adapt / improve variable size checksums
- compression rates of up to 100 to 1 may arise
Metadata alignment

Many concepts related to source code
- project, archive, source, language, licence, bts, mailing list, ...
- developer, committer, author, architect, ...

Many existing ontologies
- DOAP, FOAF, Appstream, schema.org, ADMS.SW, ...

Many disparate catalogs
- Freecode (40.000+), Plume (400+), Debian (25.000+), OpenHub (670.000+), ...

Challenge: scale up metadata to millions of projects
- reconcile existing ontologies
- link and check existing catalogs with Software Heritage
- handle inconsistent data and provenance information
The Software Diaspora

- Code often migrates across projects: forks, copy-paste
- Code gets cloned: reuse, language limitations, code smells
- Projects migrate across forges: fashion, functionality
- Projects get cloned: mirrors, packages

Challenge: tracing software evolution across billions of files

- rebuild the history of software artefacts
- identify code origins
- spot code clones
- build project impact graphs
## Distributed infrastructure

### The software graph

- files
- directories
- commits
- projects

All de-duplicated in Software Heritage

### Challenge: design efficient architectures and algorithms

- replication and availability (CAP?)
- navigation
- query
- path analysis
Outline

6 Collection strategy

7 Selected research challenges: building the archive

8 Selected research challenges: using the archive
Software as Big Data

Remember the numbers

- 70+ million repositories ingested
- 900+ million unique commits
- 4+ billion unique source files / 200 TB of raw source code

and growing by the day!

Challenge: what can machines learn here?

- programming patterns / trends
- developer skills
- vulnerabilities
- bugs and fixes
Remember the numbers

- 70+ million repositories ingested
- 900+ million unique commits
- 4+ billion unique source files / 200 TB of raw source code

and growing by the day!

Challenge: can we make this fit in memory?

- efficient graph representation
- fast non-local queries
- mitigate the size/speed tradeoff