Software Heritage
Collecting, preserving and sharing the software source code of Mankind

Roberto Di Cosmo
Inria

roberto@dicosmo.org

March 8th 2017

Software Sustainability workshop
Software is everywhere

At the heart of our society

- communication, entertainment
- administration, finance
- health, energy, transportation
- education, research, politics
- …

Knowledge enabler

- *Key mediator* for accessing *all* information
- *Essential component* of modern scientific research

Software embodies our collective Knowledge and Cultural Heritage
1 Source code is essential…

2 … but we are not taking care of it!

3 The Software Heritage initiative

4 Where we are today

5 Technical insert

6 Status
“The source code for a work means the preferred form of the work for making modifications to it.”

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

“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
/*
 * SFU uses two B[i][j] : 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 rtxhash or
c * external classifier) into 8 subhashes of 4 bits.
 */
#define SFU_BUCKET_SHIFT 4
#define SFU_NUMBUCKETS (1 << SFU_BUCKET_SHIFT)  /* N bins per level */
#define SFU_BUCKET_MASK (SFU_NUMBUCKETS - 1)
#define SFU_LEVELS (32 / SFU_BUCKET_SHIFT) /* L */

/* SFU algo uses a virtual queue, named "qbn" */
struct sfu_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.”

Roberto Di Cosmo (www.dicosmo.org)
Distinguishing features

- executable and human readable knowledge (an all time new)
  - even hardware is… software! (VHDL, FPGA, …)
  - text files are forever
- naturally evolves over time
  - the development history is key to its understanding
- complex: large web of dependencies, millions of SLOCs

In a word

- software is not just another sequence of bits
- a software archive is not just another digital archive
Free and Open Source is eating the Software world

Definition (Free Software)

A program is free software if the program’s users have the four essential freedoms:

- Freedom #0, to run the program, for any purpose
- Freedom #1, to study how the program works, and change it
- Freedom #2, to redistribute copies
- Freedom #3, to improve the program, and release improvements

There are also varying obligations according to the license: BSD, GPL, Apache, AGPL, . . .

The good news

Free / Open Source is everywhere, so there is much more source code easily available!

The bad news

...
Source code is essential...

... but we are not taking care of it!

The Software Heritage initiative

Where we are today

Technical insert

Status
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 goes away?
Software is missing its own Research Infrastructure

A wealth of software research on crucial issues...

- safety, security; test, verification, proof;
- software engineering, software evolution;
- big data, machine learning, empirical studies;

If you study the stars, you go to Atacama...

... where is the very large telescope of source code?
Outline

1. Source code is essential...
2. ... but we are not taking care of it!
3. The Software Heritage initiative
4. Where we are today
5. Technical insert
6. Status
Our mission

Collect, preserve and share the source code of all the software that is publicly available.

Past, present and future

Preserving the past, enhancing the present, preparing the future.
We are working on the foundations one infrastructure to build them all

- Mankind’s memory
- Long term preservation
- Unique reference
- Software Wikipedia

- Reference repository
- Provenance
- Certification
- Security

- Reproducibility
- Traceability
- Open Access
- Software studies

- Universal SourceBook
- Reference examples
- Enriched source code
- Code documentation

Software Heritage

Roberto Di Cosmo (www.dicosmo.org)
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
1. Source code is essential...
2. ... but we are not taking care of it!
3. The Software Heritage initiative
4. Where we are today
5. Technical insert
6. Status
Sources

- GitHub
- Debian, GNU
- Gitorious
- Google Code, …
Dataflow

Forges

GitHub linter

GitLab linter

Debian linter

PyPi linter

Software Heritage Archive

Merkle DAG + blob storage

Listing (full/incremental)

Loading & deduplication

Scheduling

Package repos...

Distros

Git loader

Mercurial loader

Debian source package loader

tar loader

...
Archiving goals

Targets: VCS repositories & source code releases (e.g., tarballs)

We DO archive

- file content (= blobs)
- revisions (= commits), with full metadata
- releases (= tags), ditto
- where (origin) & when (visit) we found any of the above

... in a VCS-/archive-agnostic canonical data model

We currently DON’T archive

- homepages, wikis
- BTS/issues/code reviews/etc. (collaboration with GHTorrent)
- mailing lists

Long term vision: play our part in a "semantic wikipedia of software"
Three properties are key for Software Heritage’s mission

**Availability**
- *all the history of all the software*
- no restrictions (technical, legal, … ) on *content or metadata*

**Traceability**
- *unique* identifiers: *one* name for each object
- *persistent* and *intrinsic* identifiers: no URLs, no DOIs, no middle man, no dangling pointers

**Uniformity**
- *one standard* metadata structure, *irrespective of the origins*
- *uniform* naming *schema*
What’s special

**Uniform data model**
- superset of *git*: ambition to *cover all VCS*
  - contents, directories, revisions, releases, origins, …

**Massive deduplication**
- the biggest git-like graph in the world right now
  - did you know? the original GPLv2 licence
    - appears with more than 500 different file names
    - including `aa.css.txt` and `FullSync.txt :-(`

**Provenance tracking**
- know *where* we found *what*, *when*
- essential for *traceability*
A complex, challenging scientific and technical task

Software Heritage

Collect
- discover
  - sources
- harvest
  - protocols
- ingest
  - VCS
  - data models

Organise and Preserve
- enrich
  - metadata
- analyze
- replicate
  - locations
  - technologies
  - stakeholders

Share
- download
- browse
- search
  - code
  - history
- watch
  - trends

Roberto Di Cosmo (www.dicosmo.org)
Our principles

**Open approach**
open source, transparency

**In for the long haul**
non profit, replication

Thomas Jefferson, February 18, 1791

…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.
Outline

1. Source code is essential...
2. ... but we are not taking care of it!
3. The Software Heritage initiative
4. Where we are today
5. Technical insert
6. Status
Merkle trees

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, Bitcoin, IPFS, …)
- built-in deduplication
The archive in a few pictures

A giant (extended) Merkle DAG
Fresh from the oven: first public version of our Web API
https://archive.softwareheritage.org/api/

Features

- pointwise **browsing** of the Software Heritage archive
  - … releases → revisions → directories → contents …
- full access to the **metadata** of archived objects
- **crawling** information
  - *when have you last visited this Git repository I care about?*
  - *where were its branches/tags pointing to at the time?*

Complete endpoint index

https://archive.softwareheritage.org/api/1/
A tour of the Web API — origins & visits

```
GET https://archive.softwareheritage.org/api/1/origin/ \
    git/url/https://github.com/hylang/hy
{
    "id": 1,
    "origin_visits_url": "/api/1/origin/1/visits/",
    "type": "git",
    "url": "https://github.com/hylang/hy"
}

GET https://archive.softwareheritage.org/api/1/origin/ \
    1/visits/
[ ...,
  { "date": 1473851066.769266,
    "origin": 1,
    "origin_visit_url": "/api/1/origin/1/visit/13/",
    "status": "full",
    "visit": 13
  }, ...
]
```
GET https://archive.softwareheritage.org/api/1/origin/ \
1/visit/13/
{
  ...
  "occurrences": {
    "refs/heads/master": {
      "target": "b94211251...",
      "target_type": "revision",
      "target_url": "/api/1/revision/b94211251.../
    },
    "refs/tags/0.10.0": {
      "target": "7045404f3...",
      "target_type": "release",
      "target_url": "/api/1/release/7045404f3.../
    },...
  },
  "origin": 1,
  "origin_url": "/api/1/origin/1/",
  "status": "full",
  "visit": 13
}
A tour of the Web API — revisions

GET https://archive.softwareheritage.org/api/1/revision/6072557b6c10cd9a21145781e26ad1f978ed14b9/

{
    "author": {
        "email": "tag@pault.ag",
        "fullname": "Paul Tagliamonte <tag@pault.ag>",
        "id": 96,
        "name": "Paul Tagliamonte"
    },
    "committer": {
        "date": "2014-04-10T23:01:11-04:00",
        "committer_date": "2014-04-10T23:01:11-04:00",
        "directory": "2df4cd84e...",
        "directory_url": "/api/1/directory/2df4cd84e.../",
        "message": "0.10: The Oh f*ck it’s PyCon release",
        "parent_urls": [ "/api/1/revision/10149f66e.../" ],
        "parents": [ "10149f66e..." ]
    }
}
A tour of the Web API — contents

GET https://archive.softwareheritage.org/api/1/content/adc83b19e793491b1c6ea0fd8b46cd9f32e592fc/

```json
{
    "data_url": "/api/1/content/sha1:adc83b19e.../raw/",
    "filetype_url": "/api/1/content/sha1:.../filetype/",
    "language_url": "/api/1/content/sha1:.../language/",
    "length": 1,
    "license_url": "/api/1/content/sha1:.../license/",
    "sha1": "adc83b19e...",
    "sha1_git": "8b1378917...",
    "sha256": "01ba4719c...",
    "status": "visible"
}
```

- rate limits apply throughout the API
- blob download not available yet
Outline

1. Source code is essential...
2. ... but we are not taking care of it!
3. The Software Heritage initiative
4. Where we are today
5. Technical insert
6. Status
### Status (team)

#### The core team
- Roberto Di Cosmo
- Stefano Zacchiroli
- Nicolas Dandrimont (Engineer)
- Antoine Dumont (Engineer)

#### Scientific advisors
- Serge Abiteboul (French Science Academy)
- Jean-François Abramatic (former W3C director)
- Gerard Berry (CNRS Gold Medal, French Science Academy)
- Julia Lawall (Coccinelle, Linux Kernel, Outreachy)
Sponsors and supporters

Société Générale, Huawei, Microsoft, Nokia Bell Labs, DANS, ACM, Creative Commons, Eclipse, FSF, OSI, GitHub, GitLab, IEEE, OIN, OW2, SFC, SFLC, The Document Foundation, The Linux Foundation, see all 20+ more http://www.softwareheritage.org/support/

Software Heritage for society as a whole

- preservation of knowledge embedded in software
- access to the knowledge embedded in software

Growing mindshare

see http://annex.softwareheritage.org/
Software Heritage

www.softwareheritage.org  @swheritage

Everybody can participate

- co-founders
- sponsoring / partnership
- working groups, leads
- our own code

contact roberto@dicosmo.org
sponsorship.softwareheritage.org
wiki.softwareheritage.org
forge.softwareheritage.org

Roberto Di Cosmo (www.dicosmo.org)
7 Selected research challenges: building the archive
8 Selected research challenges: using the archive
9 Challenges and opportunities
10 Conceptualisation
11 Our web of reference is fragile
12 Analysis of the Digital Object Identifier
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
Selected research challenges: building the archive

Selected research challenges: using the archive

Challenges and opportunities

Conceptualisation

Our web of reference is fragile

Analysis of the Digital Object Identifier
Code search: an old problem

A natural need
- Find the definition of a function/class/procedure/type/structure
- Search examples of code usage in an archive of source code
- you name it…

A natural approach
- Regular expressions

We have all used *grep* since the 1970’s!

where is the challenge?
Finding a needle in a haystack: size matters!

How do we search in millions of source code files?

Google code search (open 2006, closed 2011)
see https://swtch.com/~rsc/regexp/regexp4.html reborn in 2013 for Debian http://sources.debian.net/

how

- build an inverted index of trigrams from all source files
- map regexps to trigrams
- filter files that may match
- run grep on each file (using the cloud)

performance

scaled reasonably well up to 1 billion lines of codes
Challenge: scaling up code search

What about *all the source code* in the world?

Software Heritage is *two orders of magnitude* bigger already
- over *two billion* unique source files
- *hundreds* of billions of LOCs

We need new insight for handling this.

Beyond regular expressions?

Advanced code search requires
- language specific *patterns*
- working on *abstract syntax trees*

Regular expressions are a nice *swiss-army knife* approximation, can we build a specific tool that scales?
Remember the numbers

- 50+ million repositories ingested
- 700+ million commits
- 3+ 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
Outline

7 Selected research challenges: building the archive
8 Selected research challenges: using the archive
9 Challenges and opportunities
10 Conceptualisation
11 Our web of reference is fragile
12 Analysis of the Digital Object Identifier
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)
So many challenges and opportunities: let’s name a few

Scientific (non) reproducibility
- inconsistencies
- data corruption
- fraud
- non reproducible findings…
(picture from Nature, Sep. 2015)

(Lack of) Security and Quality of Software
- vulnerabilities and bugs in systems…
- … become critical with IoT

(Potential) Citizen empowerment
we need to have access to the software that governs our life
Reproducibility (Wikipedia)
the ability of an entire experiment or study to be reproduced
either by the researcher or by someone else working independently.

Why we want it
- foundation of the scientific method
- accelerator of research: allows to build upon previous work
- transparency of results eases acceptance
- reproducibility is the essence of industry!

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
Separate objectives

These are all conceptually different notions

- collaborative development
  - GitHub, Bitbucket, GitLab(.com), FusionForge installations…
- distribution
  - Debian, CTAN, CRAN, NPM, …
- classification
  - Freecode, Libraries.io, FLOSSMole, …
- archival
  - Software Heritage, Zenodo, …
- big data
  - Software Heritage
Selected research challenges: building the archive
Selected research challenges: using the archive
Challenges and opportunities
Conceptualisation
Our web of reference is fragile
Analysis of the Digital Object Identifier
Disruption of 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.


Scholar roster of broken links

An example from Astronomy

How Do Astronomers Share Data?
Pepe, Goodman, Muench, Crosas, Erdmann
dx.doi.org/10.1371/journal.pone.0104798

PLOS August 28, 2014
Cool URLs (should not) change

What makes a cool URI?
A cool URI is one which does not change.
What sorts of URI change?
URIs don’t change: *people change them.*

Tim Berners Lee, 1998
https://www.w3.org/Provider/Style/URI

Yes, *people* change them…
sometimes behind your back!
Disruption of the web of reference: Inria’s own Gforge

Fixed, adding a redirection, by the Gforge team

in 1 day this one was fixed!

Not always that lucky, though …
Selected research challenges: building the archive
Selected research challenges: using the archive
Challenges and opportunities
Conceptualisation
Our web of reference is fragile
Analysis of the Digital Object Identifier
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
Real world examples...

An image from figshare

let’s click on the DOI…

Roberto Di Cosmo (www.dicosmo.org)

Software Heritage  March 8th 2017  17 / 19
… may indeed not work
Good citizen

Yes, we report broken links/does

Thank you for reporting this error.

<table>
<thead>
<tr>
<th>Problem DOI:</th>
<th>10.1021/jacs.orglett.6b02915.s002</th>
</tr>
</thead>
<tbody>
<tr>
<td>Publisher who owns the prefix 10.1021:</td>
<td>American Chemical Society</td>
</tr>
<tr>
<td>User's email address:</td>
<td><a href="mailto:doi@dicosmo.org">doi@dicosmo.org</a></td>
</tr>
<tr>
<td>Referring Page:</td>
<td><a href="https://flgsbire.com">https://flgsbire.com</a></td>
</tr>
<tr>
<td>Comments:</td>
<td></td>
</tr>
</tbody>
</table>

The DOI and comments (if provided) have been logged by CrossRef and forwarded to the publisher to correct the problem. Possible reasons for the error are:

- the DOI has been created but has not been registered by the publisher (this could be an error or it could be a timing issue and the DOI will be registered in the next few days)
- the DOI is cited incorrectly in the source
- the DOI does not resolve due to a system problem

Maintaining the integrity of DOIs is very important to CrossRef and we appreciate your help.