What would you do with **billions** of source code files?

Challenges and opportunities in software archival

Roberto Di Cosmo

roberto@dicosmo.org

February 7th 2017

Inria - EPFL workshop

Software Heritage

THE GREAT LIBRARY OF SOURCE CODE
Ten years of research on open source software

Di Cosmo, Leroy, Treinen, Vouillon et al.
*Managing the complexity of large free and open source package-based software distributions*
*ASE 2006*

Abate, Boender, Di Cosmo, Zacchirolí
*Strong Dependencies between Software Components, ESEM 2009*

Di Cosmo and J. Vouillon.
On software component co-installability, *ESEC/FSE 2011*

Abate, Di Cosmo, Treinen, Zacchirolí
*Learning from the Future of Component Repositories, CBSE 2012*

Vouillon, Dogguy, Di Cosmo.
Easing software component repository evolution. *ICSE 2014*

Abate, Di Cosmo, Gesbert, Le Fessant, Treinen, and Zacchirolí.
Mining component repositories for installability issues, *MSR 2015*

Claes, Mens, Di Cosmo, and Vouillon.
A historical analysis of debian package incompatibilities, *MSR 2015*

### Tools

- Dose library: [http://gforge.inria.fr/projects/dose/](http://gforge.inria.fr/projects/dose/)
- Coinst suite: [http://coinst.irill.org](http://coinst.irill.org)
- Debian QA: [http://qa.debian.org/dose](http://qa.debian.org/dose)
Ten years of research on open source software

A recurring pattern

- identify a real world problem whose solution requires a research effort
- work hard to find a solution
- implement a tool, validate it on real world cases
- publish a research article
- foster adoption (the hardest part!)

In a picture

Under the hood

Question:

What were the technical prerequisites that made this work possible?
Technical and legal enablers

Availability
- all the (history of) Debian packages (since 2005)
- no *technical* restrictions
- no *legal* restrictions on content or metadata

Traceability
Debian packages have
- *unique identifier*
- *reference central repository*

Uniformity
Debian packages: a reference catalog
- *uniform metadata structure*
- *uniform naming and versioning schema*

These are all essential features for *reproducibility* and for *preservation*...  
... we need them for *all* software!
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
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;
- empirical and big data studies;

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

... where is the very large telescope of source code?
Our mission

Collect, preserve and share the source code of all the software that lies at the heart of our culture and our society.

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
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
The Knowledge Conservancy Magic Triangle

Legenda (links are important!)

- articles: ArXiv, HAL, ...
- data: Zenodo, ...
- software: *Software Heritage* to the rescue
Archive coverage

Our sources

- GitHub — full, up-to-date mirror
- Debian — daily snapshots of all suites since 2005–2015
- GNU — all releases as of August 2015
- Gitorious, Google Code — local copy (Archive Team & Google)

Some numbers

150 TB blobs, 6 TB database (as a graph: 5 B nodes + 50 B edges)

The richest source code archive already, … and growing daily!
The archive: a (giant) Merkle DAG
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
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?
So, software as Big Data

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
Web API (FOSDEM’17 release)

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/
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
}
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.../",
  "history_url": "/api/1/revision/6072557b6.../log/",
  "merge": false,
  "message": "0.10: The Oh f*ck it’s PyCon release",
  "parent_urls": [ "/api/1/revision/10149f66e.../" ],
  "parents": [ "10149f66e..." ]
}
GET https://archive.softwareheritage.org/api/1/content/\adc83b19e793491b1c6ea0fd8b46cd9f32e592fc/

{
    "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
The Software Heritage community

The core team

- Roberto Di Cosmo
- Stefano Zacchiroli
- Nicolas Dandrimont (Engineer)
- Antoine Dumont (Engineer)

Inria as initiator

Inria, the .fr CS research institution, strong FOSS culture, W3C founding partner

Early Partners and Supporters

Software Heritage is

- a reference archive of all FOSS ever written
- a fantastic new tool for research software
- a unique complement for development platforms
- an international, open, nonprofit, mutualized infrastructure
- at the service of our community, at the service of society

Learn more

www.softwareheritage.org – sponsoring, job openings
wiki.softwareheritage.org – internships, leads
forge.softwareheritage.org – our own code

Questions?