Software Heritage

Can we handle all our source code?

Roberto Di Cosmo

roberto@dicosmo.org

January 23rd, 2019
EGC 2019

Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Outline

1. Introductions
2. Software is knowledge
3. Software Heritage
4. Under the hood: architecture and data structure
5. Under the hood: identifying billions of objects
6. Demo time
7. A revolutionary infrastructure
8. Building for the long term
9. Conclusion

Short Bio: Roberto Di Cosmo

Computer Science professor in Paris, now working at INRIA

- *30 years* of research (Theor. CS, Programming, Software Engineering, Erdos #: 3)
- *20 years* of Free and Open Source Software
- *10 years* building and directing structures for the common good

1999 *DemoLinux* – first live GNU/Linux distro

2007 *Free Software Thematic Group*
150 members 40 projects 200Me

2015 *Software Heritage* at INRIA

2018 *National Committee for Open Science*, France

Outline

1. Introductions
2. Software is knowledge
3. Software Heritage
4. Under the hood: architecture and data structure
5. Under the hood: identifying billions of objects
6. Demo time
7. A revolutionary infrastructure
8. Building for the long term
9. Conclusion

Software is everywhere

Software is everywhere

Bottomline

software embodies our scientific, technical and cultural heritage

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 III source code (excerpt)

```
float Q_rsqrt( float number )
{
    long i;
    float x2, y;
    const float threehalfs = 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 * ( threehalfs - ( x2 * y * y ) ); // 1st iteration
// y = y * ( threehalfs - ( x2 * y * y ) ); // 2nd iteration, this
can be removed

    return y;
}
```

Net. queue in Linux (excerpt)

```
/*
 * SFB uses two B[l][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

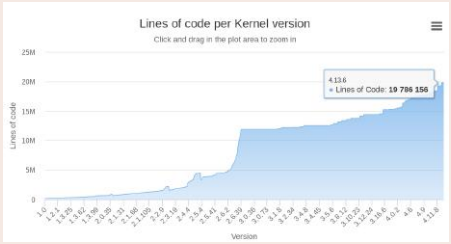
~ 50 years, a lightning fast growth

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!

Outline

1. Introductions
2. Software is knowledge
3. Software Heritage
4. Under the hood: architecture and data structure
5. Under the hood: identifying billions of objects
6. Demo time
7. A revolutionary infrastructure
8. Building for the long term
9. Conclusion

Software Heritage, in a nutshell

Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share the *source code* of *all the software*

Preserving our heritage, enabling better software and better science for all

Software Heritage, in a nutshell

Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share the *source code* of *all the software*

Preserving our heritage, enabling better software and better science for all

Reference catalog

find and reference all the source code

Software Heritage, in a nutshell

Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share the *source code* of *all the software*

Preserving our heritage, enabling better software and better science for all

Reference catalog

find and reference all the source code

Universal archive

preserve all the source code

Software Heritage, in a nutshell

Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share the *source code* of *all the software*

Preserving our heritage, enabling better software and better science for all

Reference catalog

find and reference all the source code

Universal archive

preserve all the source code

Research infrastructure

enable analysis of all the source code

A principled infrastructure

A principled infrastructure

Technology

- transparency and FOSS
- replicas all the way down

Content

- intrinsic identifiers
- facts and provenance

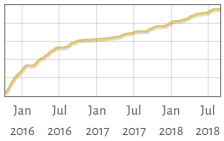
Organization

- non-profit
- multi-stakeholder

Coverage

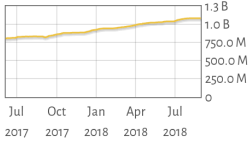
Source files

5,603,274,836



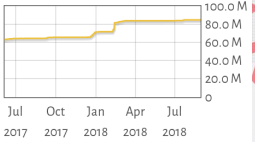
Commits

1,248,389,319



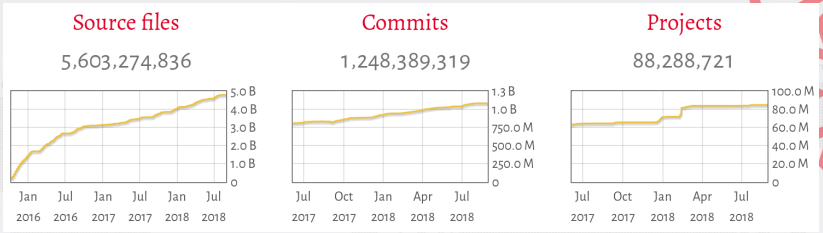
Projects

88,288,721



GitHub · debian · GitLab · Google code · {P} · python Package Index · GITORIOUS · GNU · HAL archives-ouvertes.fr · Inria inventeurs du monde numérique

Coverage



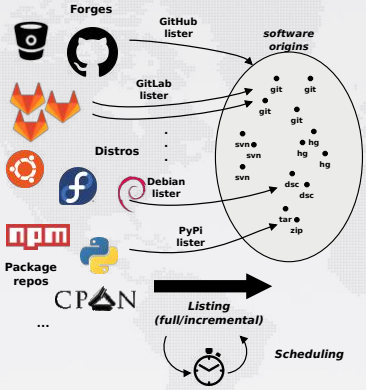
GitHub debian GitLab Google code {P} python Package Index GITORIOUS GNU HAL archives-ouvertes.fr Inria inventeurs du monde numérique

- 200 TB (compressed) blobs, 6 TB database (as a graph: 10 B nodes + 100 B edges)
- The *richest* public source code archive, … and growing daily!

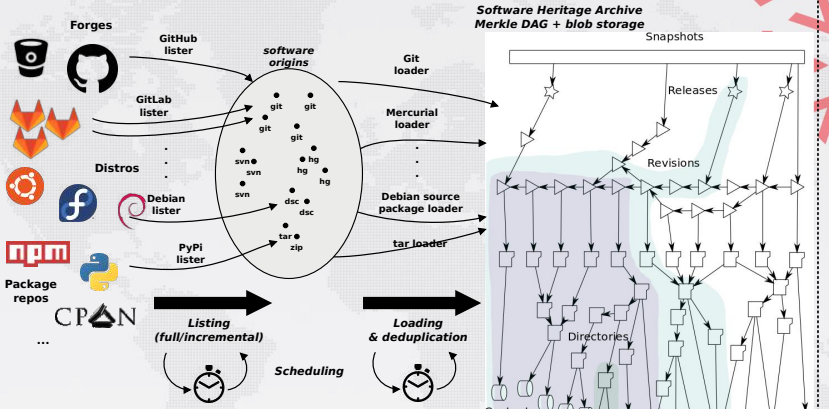
Outline

1. Introductions
2. Software is knowledge
3. Software Heritage
4. Under the hood: architecture and data structure
5. Under the hood: identifying billions of objects
6. Demo time
7. A revolutionary infrastructure
8. Building for the long term
9. Conclusion

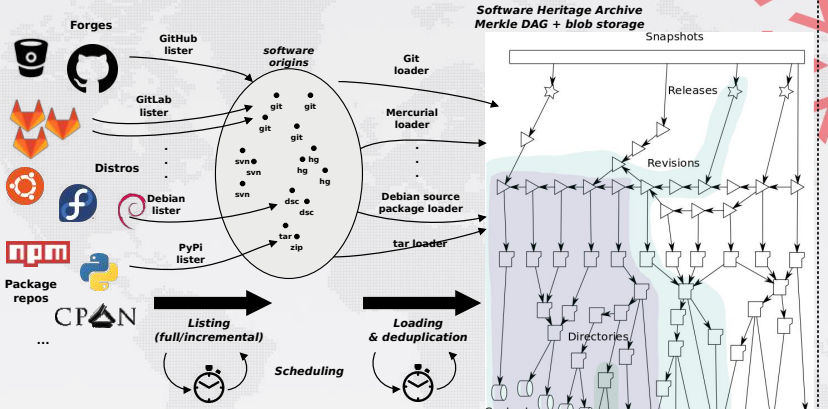
Automation, and storage



Automation, and storage



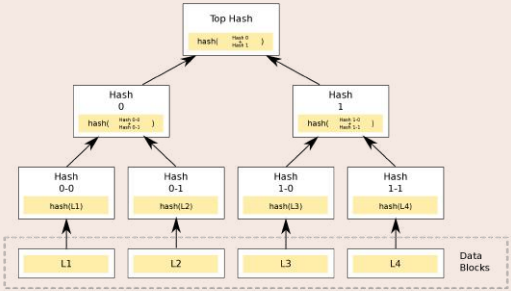
Automation, and storage



Full development history permanently archived in a uniform data model.

Much more than an archive!

Merkle tree (R. C. Merkle, Crypto 1979)



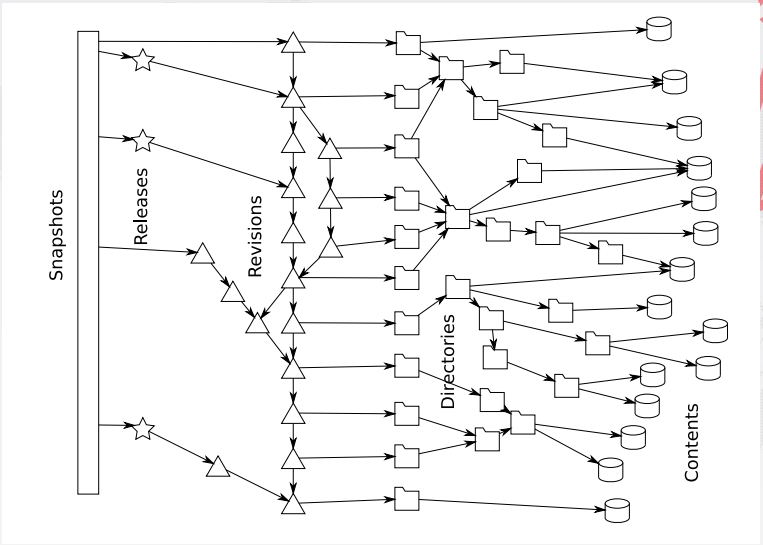
Combination of

- tree
- hash function

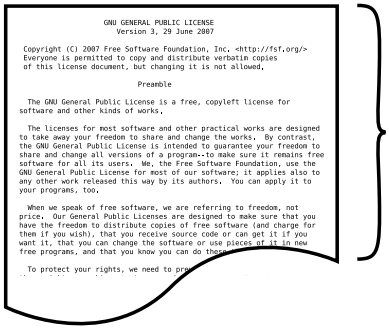
Classical cryptographic construction

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

The archive in pictures



Contents



GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

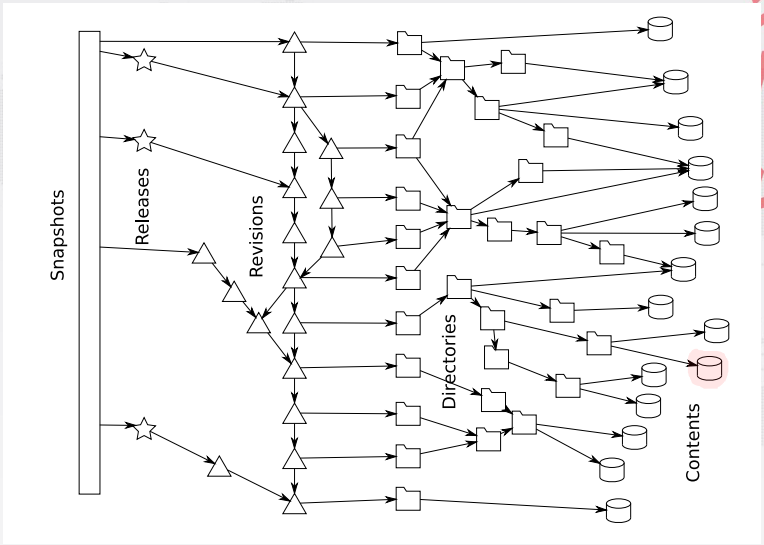
The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these

To protect your rights, we need to pre

sha1: 8624bcdae55baeef...
sha256: 8ceb4b9ee5aded...
sha1_git: 94a9ed024d385...
length: 35147

The archive in pictures

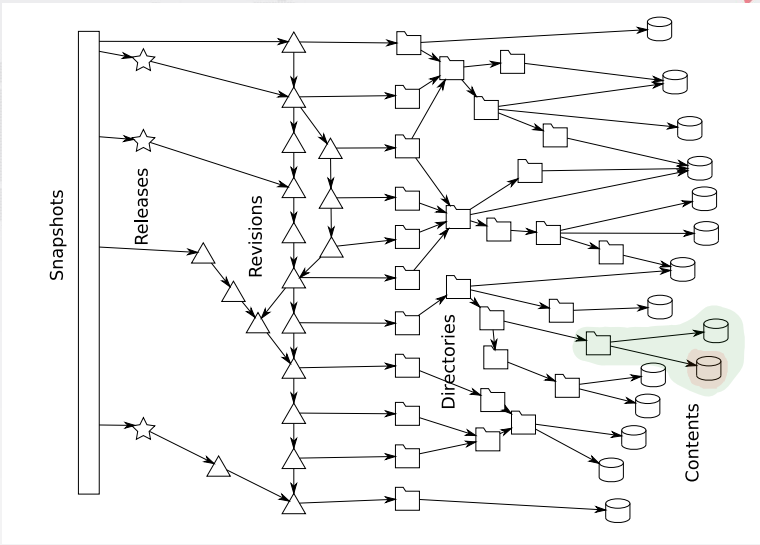


Directories

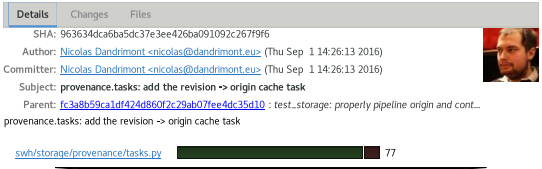
```
100644 blob c5baade4c44766042186ef858c0fd63d587ebf09 .gitignore
100644 blob 2d0a34af6f52cf3cf6b0c2f7bd0648fbd255e77f AUTHORS
100644 blob 94a9ed024d3859793618152ea559a168bbcbb5e2 LICENSE
100644 blob d9b2665a435a43f8a79a84e0867751dfb095c7bb MANIFEST.in
100644 blob 524175c2bad0b35b975f79284c2f5a6d5eaf2eb4 Makefile
100644 blob 5c7e3a5bbddb038682ba7793f440492ed9678bb3 Makefile.local
100644 blob 8617980629cd24e6080404f09aa749b085b3e07b README.db_testing
100644 blob 76b29f94cf815e0869c414d38d78d7ce08ec514e README.dev
040000 tree e1e10ecef948af0b93adb0372afc89f12e92618a bin
040000 tree 83e56d0beaf7793c77a45a345c80fcb8af503013 debian
040000 tree a34c9c4ba213f0cedc67f9816348d27955577af5 docs
100644 blob f2a6d32c6135aa7287bbd76167b01df2ae4f1539 requirements.txt
100755 blob eee147c36caf1bbc2d820da8dc026cb5b68180bc setup.py
040000 tree 224bb4c1f4c67fca1d160bffd2d06094e7e1abf3 sql
040000 tree 8631c9cd77bbe993168107ab5baf51f40c6300be swh
040000 tree 8fb905b56ba8ed692f1209b2773b474c6c1d66c1 utils
```

id: 515f00d44e92c65322aaa9bf3fa097c00ddb9c7d

The archive in pictures



Revisions

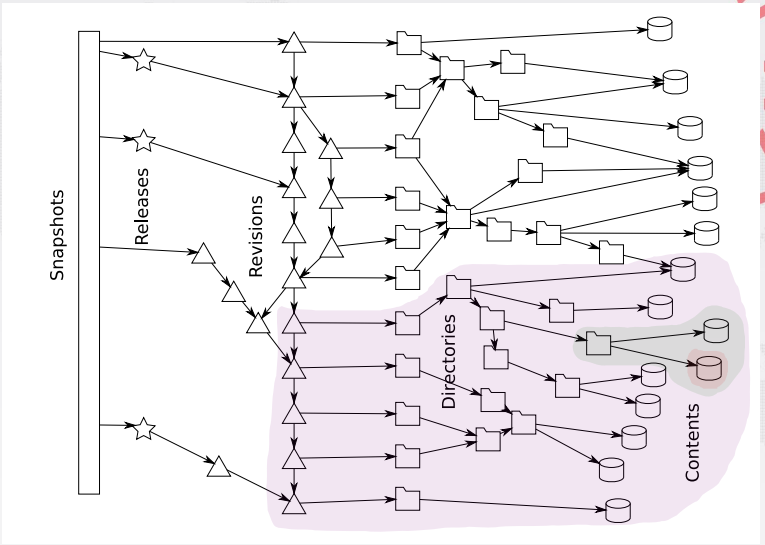


tree 515f00d44e92c65322aaa9bf3fa097c00ddb9c7d
parent fc3a8b59ca1df424d860f2c29ab07fee4dc35d10
author Nicolas Dandrimont <nicolas@dandrimont.eu> 1472732773 +0200
committer Nicolas Dandrimont <nicolas@dandrimont.eu> 1472732773 +0200

provenance.tasks: add the revision -> origin cache task

id: 963634dca6ba5dc37e3ee426ba091092c267f9f6

The archive in pictures



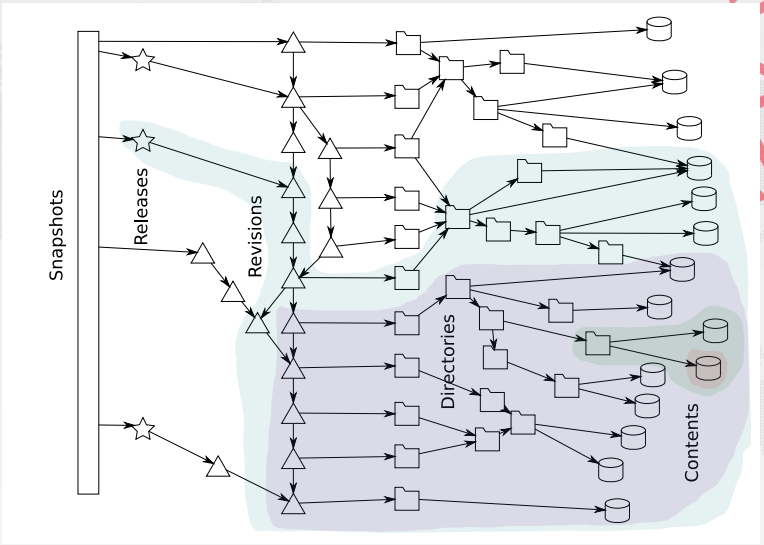
Releases

```
tag v0.0.51
Tagger: Nicolas Dandrimont <nicolas@dandrimont.eu>
Date:   Wed Aug 24 14:36:03 2016 +0200

Release swh.storage v0.0.51

 - Add new metadata column to origin_visit
 - Update swh-add-directory script for updated API
[...]

commit c0c9f16b1e134f593e7567570a1761b156e6eb1d
```

```
object c0c9f16b1e134f593e7567570a1761b156e6eb1d
type commit
tag v0.0.51
tagger Nicolas Dandrimont <nicolas@dandrimont.eu> 1472042163 +0200

Release swh.storage v0.0.51

 - Add new metadata column to origin_visit
 - Update swh-add-directory script for updated API
-----BEGIN PGP SIGNATURE-----

iQIzBAABCAAdBQJXvZTNFhxuaWNvbGFzQGRhbmRyaW1vbnQuZXUACgkQ7AWLMo2+
neqorw//aq6SOb5DijzEa+kWN3rXgVS+1K1vEVh1wNKAwx8eKJ7aX2kEiLDtt7uf
ahpZ6pz3q8nqs6aC1+YrxBfcih3L2YtrdZeWXWqr8xWNMaEoYDb8qaphwh8AD5t2
ICBIlt2ujtXuCrDt93eKKPwvzZXq+hB0sMWy35Dr6jW7Z7K4Mu/PGglyIHPY55yo
IGEndWno7Vfh1Vm6t1n5qB7I5mXRaqA+becqddubTZ2xjj+jpIUqC8cyqN3hm/fL
qsj2mu8kyz3t8tG/H1/pV+I5OwBInPoS5TH0tujoJEVgPK/dHSP79QuHDHZFkCao
klj6kAWyU80Mxb+nKV/jeLbrR3+yWBFj3Qp5a1/V8oOTh6E1dALcNMpEaKCoKtMt
d/gMRax1IL/g0EDfnsW67G6sDwKPKPHngfVLQ3nV3GaQQTnu1RpMz006H9/hAwzC
Gg/K1PdHT4hzOll46wYPZyje0U2VXGFu6vVU9vFQ4ZR/WJn+0zMzdcRdRJlSUOMn
RpTTfUsbXUeXHGOpkgXhSYTnvp1gdPc76USTSk0aGe84AZm1Ik0mGrwXCVfPqlYo
nhhibBSHBNMoqyF6yTSOpUbYK70tpYRRUGKWDeRK0wKSxkWKUZGtKzy6JYqJjo29
gulwgZQif5qWQCB0OontAL2+HvPFaVyckMejUhg62cP/+EHIvUk=
=kOxP
-----END PGP SIGNATURE-----
```

id: 85083a5cc14a441c89dea73f5bdf67c3f9c6afdb

The archive in pictures



git show-refs

Snapshots

```
commit 08ffeb25770109525eb3ce21691466c53a1d9158 refs/heads/atime
commit ba5443a24e3f9fe323a46c292cec4fcbe61c67eb refs/heads/directory-listing-arrays
commit d69e0dbf892383ff6589b27fbe1c05d27238d9c5 refs/heads/foo
commit cf7ff9eea0eb22f8946908f5a8019f67de468e08 refs/heads/master
commit 7eca197fc66d2024047e54b1ed9e8b44361a0fc2 refs/heads/tmp-directory-add
commit 642a205f37de85005a85d427b53ee4fb2252e82e refs/heads/tmp/generic-releases
tag 20f043b1379cf768d966597799fd4907c757f755 refs/tags/v0.0.1
tag 72a21991a384e539996dbb867bfb0bee72aee2cd refs/tags/v0.0.10
tag 3590e0ca0ebb070e5b376705fa230bbfa4ffa5cc refs/tags/v0.0.11
tag 33378427a403ba569a67777b8d58f6674fbc6556 refs/tags/v0.0.12
tag 06f74652755b327cf590311c2bfa036cf3b4b35d refs/tags/v0.0.13
tag 5a6325fe86ab854b581d7442667d92a11e32f3bd refs/tags/v0.0.14
tag 586fba4e580b4f5fab05f599367643cbcb1a9c7f refs/tags/v0.0.15
tag 8cd8b885f4090bf363177742bd289f660e5be51c refs/tags/v0.0.16
tag a542444ee3f0fbed35efb202fee035c809abc7d6 refs/tags/v0.0.17
tag 228a2f1650dd12222e556559462e1e06fc4993d9 refs/tags/v0.0.18
tag 606979a4ca05d497fc0d24aad00dce82636ef47c refs/tags/v0.0.19
tag 32bf5a59fc2a323baa6d5f15a6ad5382ec275a67 refs/tags/v0.0.2
tag 3147c3d31ec46cf6492f801e908b1237ebdff2c7 refs/tags/v0.0.20
tag 215ea50daba111e082e0b72e76eb4b6073a87908 refs/tags/v0.0.21
tag 3fb168c2072a5d6252124257a1e5dfc0f5ffa1df refs/tags/v0.0.22
tag 8cdbee8da4d73fc5d262789e460a16ac3c72aba4 refs/tags/v0.0.23
...
```

id: b464cad1b66fff266a37b46ea6e7a04b545e904b

Outline

1. Introductions
2. Software is knowledge
3. Software Heritage
4. Under the hood: architecture and data structure
5. Under the hood: identifying billions of objects
6. Demo time
7. A revolutionary infrastructure
8. Building for the long term
9. Conclusion

Our challenges in the PID landscape

Typical properties of systems of identifiers

uniqueness, non ambiguity, persistence, abstraction (opacity)

Our challenges in the PID landscape

Typical properties of systems of identifiers

uniqueness, non ambiguity, persistence, abstraction (opacity)

Key needed properties from our use cases

gratis	identifiers are free (billions of objects)
integrity	the associated object cannot be changed (sw dev, *reproducibility*)
no middle man	no central authority is needed (sw dev, *reproducibility*)

Our challenges in the PID landscape

Typical properties of systems of identifiers

uniqueness, non ambiguity, persistence, abstraction (opacity)

Key needed properties from our use cases

gratis	identifiers are free (billions of objects)
integrity	the associated object cannot be changed (sw dev, *reproducibility*)
no middle man	no central authority is needed (sw dev, *reproducibility*)

we could not find systems with both integrity and no middle man !

An important distinction: DIOs vs. IDOs

The term "Digital Object Identifier" is construed as "digital identifier of an object," rather than "identifier of a digital object" — *Norman Paskin. 2010*

An important distinction: DIOs vs. IDOs

The term "Digital Object Identifier" is construed as "digital identifier of an object," rather than "identifier of a digital object" — *Norman Paskin. 2010*

DIO (Digital Identifier of an Object) — identifiers for (potentially) non digital objects

- epistemic complexity (manifestations, versions, locations, etc.)
- need an authority to ensure persistence and uniqueness

An important distinction: DIOs vs. IDOs

The term "Digital Object Identifier" is construed as "digital identifier of an object," rather than "identifier of a digital object" — *Norman Paskin. 2010*

DIO (Digital Identifier of an Object) — identifiers for (potentially) non digital objects

- epistemic complexity (manifestations, versions, locations, etc.)
- need an authority to ensure persistence and uniqueness

IDO (Identifier of a Digital Object) — identifiers (only) for digital objects

- can provide both integrity and no middle man
- broadly used in modern software development (git, etc.)

An important distinction: DIOs vs. IDOs

The term "Digital Object Identifier" is construed as "digital identifier of an object," rather than "identifier of a digital object" — *Norman Paskin. 2010*

DIO (Digital Identifier of an Object) — identifiers for (potentially) non digital objects

- epistemic complexity (manifestations, versions, locations, etc.)
- need an authority to ensure persistence and uniqueness

IDO (Identifier of a Digital Object) — identifiers (only) for digital objects

- can provide both integrity and no middle man
- broadly used in modern software development (git, etc.)

IDOs and DIOs adress different needs

- for the core Software Heritage IDOs are enough
- we must not use DIOs for reproducibility

The Software Heritage IDO schema (see http://bit.ly/swhpids)

swh:1:**cnt**:94a9ed024d3859793618152ea559a168bbcbb5e2 full text of the GPL3 license

The Software Heritage IDO schema (see http://bit.ly/swhpids)

swh:1:**cnt**:94a9ed024d3859793618152ea559a168bbcbb5e2 full text of the GPL3 license

swh:1:**dir**:d198bc9d7a6bcf6db04f476d29314f157507d505 Darktable source code

The Software Heritage IDO schema (see http://bit.ly/swhpids)

swh:1:**cnt**:94a9ed024d3859793618152ea559a168bbcbb5e2 full text of the GPL3 license

swh:1:**dir**:d198bc9d7a6bcf6db04f476d29314f157507d505 Darktable source code

swh:1:**rev**:309cf2674ee7a0749978cf8265ab91a60aea0f7d

a **revision** in the development history of Darktable

The Software Heritage IDO schema (see http://bit.ly/swhpids)

swh:1:**cnt**:94a9ed024d3859793618152ea559a168bbcbb5e2 full text of the GPL3 license

swh:1:**dir**:d198bc9d7a6bcf6db04f476d29314f157507d505 Darktable source code

swh:1:**rev**:309cf2674ee7a0749978cf8265ab91a60aea0f7d

a **revision** in the development history of Darktable

swh:1:**rel**:22ece559cc7cc2364edc5e5593d63ae8bd229f9f

release 2.3.0 of Darktable, dated 24 December 2016

The Software Heritage IDO schema (see http://bit.ly/swhpids)

swh:1:**cnt**:94a9ed024d3859793618152ea559a168bbcbb5e2 full text of the GPL3 license

swh:1:**dir**:d198bc9d7a6bcf6db04f476d29314f157507d505 Darktable source code

swh:1:**rev**:309cf2674ee7a0749978cf8265ab91a60aea0f7d

a **revision** in the development history of Darktable

swh:1:**rel**:22ece559cc7cc2364edc5e5593d63ae8bd229f9f

release 2.3.0 of Darktable, dated 24 December 2016

swh:1:**snp**:c7c108084bc0bf3d81436bf980b46e98bd338453

a **snapshot** of the entire Darktable repository (4 May 2017, GitHub)

The Software Heritage IDO schema (see http://bit.ly/swhpids)

swh:1:**cnt**:94a9ed024d3859793618152ea559a168bbcbb5e2 full text of the GPL3 license

swh:1:**dir**:d198bc9d7a6bcf6db04f476d29314f157507d505 Darktable source code

swh:1:**rev**:309cf2674ee7a0749978cf8265ab91a60aea0f7d

a **revision** in the development history of Darktable

swh:1:**rel**:22ece559cc7cc2364edc5e5593d63ae8bd229f9f

release 2.3.0 of Darktable, dated 24 December 2016

swh:1:**snp**:c7c108084bc0bf3d81436bf980b46e98bd338453

a **snapshot** of the entire Darktable repository (4 May 2017, GitHub)

Current resolvers: archive.softwareheritage.org and n2t.org

Outline

1. Introductions
2. Software is knowledge
3. Software Heritage
4. Under the hood: architecture and data structure
5. Under the hood: identifying billions of objects
6. Demo time
7. A revolutionary infrastructure
8. Building for the long term
9. Conclusion

Demo time: let's highlight some features...

A "wayback machine" for software source code

- http://archive.softwareheritage.org/browse

Identification and sharing of billions of software artifacts

- http://bit.ly/swhpids for persistent identifiers

Depositing research software

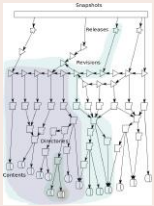
- http://bit.ly/swdepositblog

Outline

1. Introductions
2. Software is knowledge
3. Software Heritage
4. Under the hood: architecture and data structure
5. Under the hood: identifying billions of objects
6. Demo time
7. A revolutionary infrastructure
8. Building for the long term
9. Conclusion

A revolutionary infrastructure for industry

The *graph* of Software Development

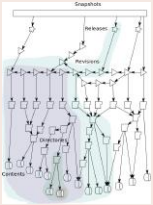


All of the software development in a single graph!

- lookup by content hash
- wayback machine for software development
  - `http://archive.softwareheritage.org/`
- … and much more

A revolutionary infrastructure for industry

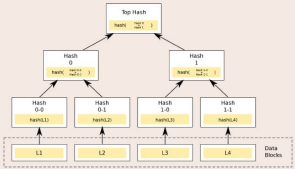
The *graph* of Software Development



All of the software development in a single graph!

- lookup by content hash
- wayback machine for software development
  - `http://archive.softwareheritage.org/`
- … and much more

The *blockchain* of Software Development

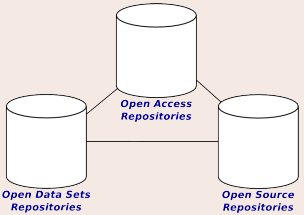


All of a software development… in a single Merkle graph!
Widely used crypto (e.g., Git, blockchains, IPFS, …)

- built-in deduplication
- intrinsic, unforgeable identifiers at all levels
- simplifies traceability (licensing, supply chain management)

A revolutionary infrastructure for research and innovation

A *pillar* of Open Science

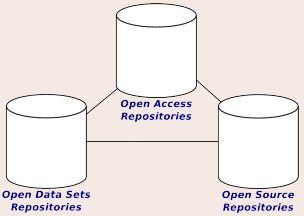


The *reference archive* of Research Software for Open Science

- curated deposit of research software
  - in collaboration with HAL, CCSD and Inria IES
  - now open *to all researchers*!
- intrinsic identifiers for reproducibility

A revolutionary infrastructure for research and innovation

A *pillar* of Open Science



The *reference archive* of Research Software for Open Science

- curated deposit of research software
  - in collaboration with HAL, CCSD and Inria IES
  - now open *to all researchers*!
- intrinsic identifiers for reproducibility

Reference platform for *Big Code*

- unique observatory of all software development
- big data, machine learning paradise: classification, trends, coding patterns, code completion…

Outline

1. Introductions
2. Software is knowledge
3. Software Heritage
4. Under the hood: architecture and data structure
5. Under the hood: identifying billions of objects
6. Demo time
7. A revolutionary infrastructure
8. Building for the long term
9. Conclusion

Raising Awareness

April 3rd 2017, Unesco Inria agreement

Inria
INVENTEURS DU MONDE NUMÉRIQUE

UNESCO

November 2018, Unesco Inria expert call

UNESCO
United Nations Educational, Scientific and Cultural Organization
"Building peace in the minds of
ABOUT US | THEMES | COUNTRIES | PARTNERSHIPS | JO

Home › All News › Experts call for greater recognition of software source code as heritage for sustainable development

Experts call for greater recognition of software source code as heritage for sustainable development

16 November 2018

Growing Support

Sharing the vision

Donors, members, sponsors

Research collaboration

Qwant source code search engine

Global network

FOSSID

- first independent mirror
- increased reliability

You can help!

Many scientific and technological challenges

machine learning, classification, efficient graph queries, metadata, …

You can help!

Many scientific and technological challenges

machine learning, classification, efficient graph queries, metadata, …

Reproducible Open Science

archive research software in SWH

reference it using *intrinsic identifiers*

build on SWH thematic portals for your discipline

You can help!

Many scientific and technological challenges

machine learning, classification, efficient graph queries, metadata, …

Reproducible Open Science

archive research software in SWH

reference it using *intrinsic identifiers*

build on SWH thematic portals for your discipline

Funding

- give *your own contribution* :
  `www.softwareheritage.org/donate`
- become a partner/sponsor/mirror :
  `sponsorship.softwareheritage.org`

You can help!

Many scientific and technological challenges

machine learning, classification, efficient graph queries, metadata, …

Reproducible Open Science

archive research software in SWH

reference it using *intrinsic identifiers*

build on SWH thematic portals for your discipline

Funding

- give *your own contribution* :
  `www.softwareheritage.org/donate`
- become a partner/sponsor/mirror :
  `sponsorship.softwareheritage.org`

Spread the word!

- *use* the archive and help others use it
- tell everybody about Software Heritage

Outline

1. Introductions
2. Software is knowledge
3. Software Heritage
4. Under the hood: architecture and data structure
5. Under the hood: identifying billions of objects
6. Demo time
7. A revolutionary infrastructure
8. Building for the long term
9. Conclusion

Join the revolution!

www.softwareheritage.org @swheritage

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

Jean-François Abramatic, Roberto Di Cosmo, Stefano Zacchiroli
Building the Universal Archive of Source Code
Communication of the ACM, October 2018

Roberto Di Cosmo, Morane Gruenpeter, Stefano Zacchiroli
Identifiers for Digital Objects: the Case of Software Source Code Preservation
iPRES 2018: Intl. Conf. on Digital Preservation

Outline

10 Strategy

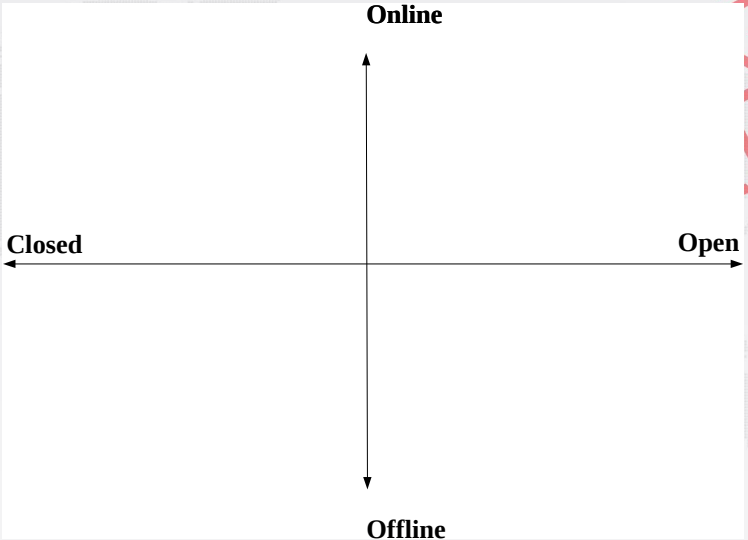
11 Software source code: a pillar of Open Science

12 An inconvenient truth

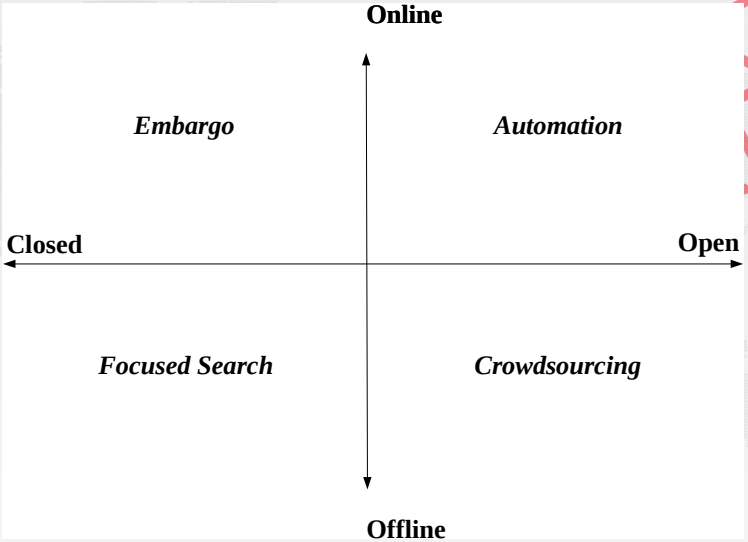
13 Data structure

14 Identifiers

All the source code



All the source code: strategy



Outline

10 Strategy

11 Software source code: a pillar of Open Science

12 An inconvenient truth

13 Data structure

14 Identifiers

The scientific method...

The experimental method

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

And then we reproduce and verify.

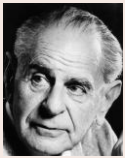
The scientific method...

The experimental method

- make an *observation*
- formulate an *hypothesis*
- set up an experiment
- elaborate 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

… evolves 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

… evolves 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*?

An example from my research field, Computer Science

Analysis of 613 papers

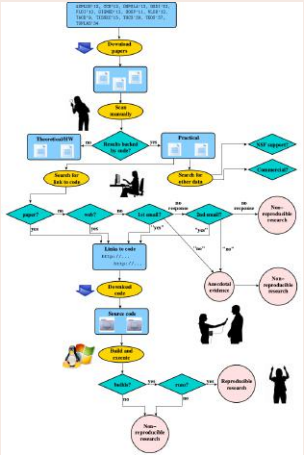
- 8 ACM conferences: ASPLOS'12, CCS'12, OOPSLA'12, OSDI'12, PLDI'12, SIGMOD'12, SOSP'11, VLDB'12
- 5 journals: TACO'9, TISSEC'15, TOCS'30, TODS'37, TOPLAS'34

all very practical oriented

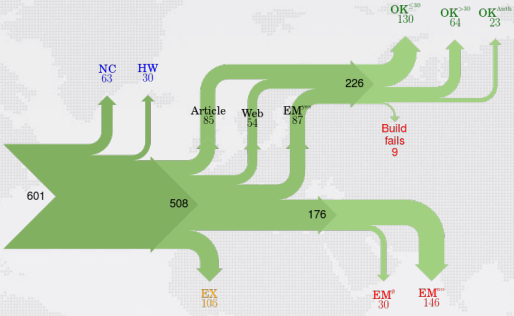
The basic question

can we get the code to build and run?

The workflow



… cont'd



…

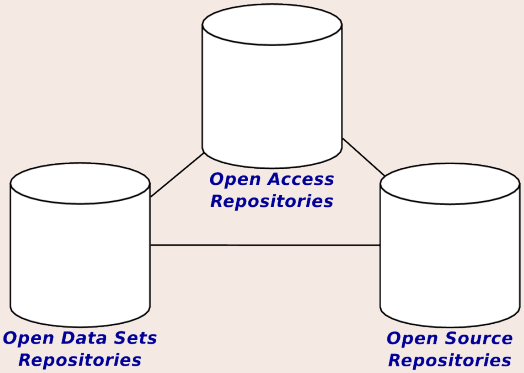
… that's a whopping 40% of non reproducible works!

The main reasons

source code (*or the right version of it*) cannot be found

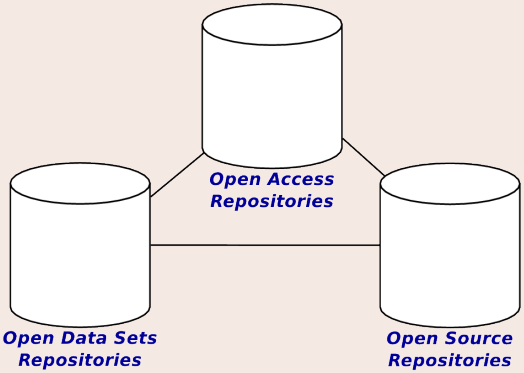
Software Source code is an important pillar

The Magic Triangle of Scientific Knowledge



Software Source code is an important pillar

The Magic Triangle of Scientific Knowledge



Nota bene

The links in the picture are essential

Outline

10 Strategy

11 Software source code: a pillar of Open Science

12 An inconvenient truth

13 Data structure

14 Identifiers

A *forgotten* pillar of Open Science

No reference catalog

to find and reference all the source code

A *forgotten* pillar of Open Science

No reference catalog

to find and reference all the source code

No universal archive

to preserve all the source code

A *forgotten* pillar of Open Science

No reference catalog

to find and reference all the source code

No universal archive

to preserve all the source code

No research infrastructure

to enable analysis of all the source code

A *forgotten* pillar of Open Science

No reference catalog

to find and reference all the source code

No universal archive

to preserve all the source code

No research infrastructure

to enable analysis of all the source code

Lack of recognition

not (yet) a first class citizen

- in the EOSC plan
- in the EU copyright reform
- in the scholarly works

A *forgotten* pillar of Open Science

No reference catalog

to find and reference all the source code

No universal archive

to preserve all the source code

No research infrastructure

to enable analysis of all the source code

Lack of recognition

not (yet) a first class citizen

- in the EOSC plan
- in the EU copyright reform
- in the scholarly works

Lack of guidance on how to

- choose a license
- cite a software project
- make source code available
- relate to industry best practices

No catalog, no archive, no references: 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.

No catalog, no archive, no references: 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.

No catalog, no archive, no references: 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!

Outline

10 Strategy

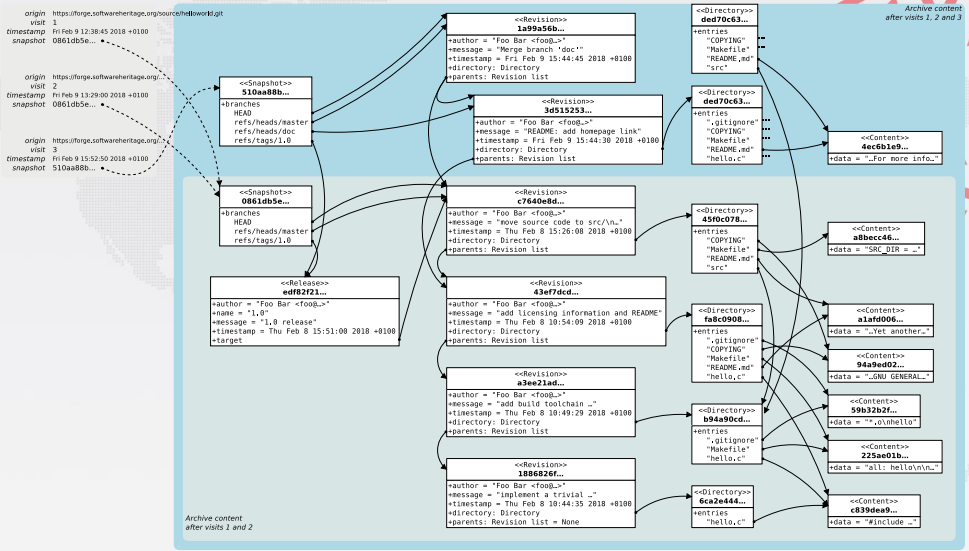
11 Software source code: a pillar of Open Science

12 An inconvenient truth

13 Data structure

14 Identifiers

A bird's eye view



Outline

10 Strategy

11 Software source code: a pillar of Open Science

12 An inconvenient truth

13 Data structure

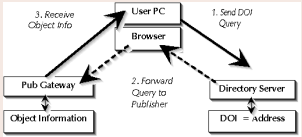
14 Identifiers

Limitations of DIOs

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

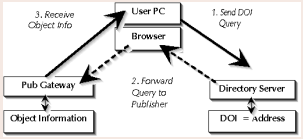


Limitations of DIOs

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*