Software as Critical Infrastructure

open source, resilience, and the backbone digital twins need

Roberto Di Cosmo

Director, Software Heritage
Inria and Université Paris Cité

7 July 2026
EDT 1st Scientific Workshop
Inria, Rennes, France

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

Outline

Short Bio: Roberto Di Cosmo

Computer Science professor in Paris, now working at INRIA

- *35+ years* of research (Theor. CS, Programming, Software Engineering, Erdos #: 3)
- *25+ years* of Free and Open Source Software
- *15+ 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

2008 *Mancoosi project* www.mancoosi.org

2010 *IRILL* www.irill.org

2015 *Software Heritage* at INRIA

2018 *National Committee for Open Science*, France

2021 *EOSC Task Force on Infrastructures for Software*, European Union

Outline

Software is all around us

Invisible fabric of digital society

Knowledge is in the source

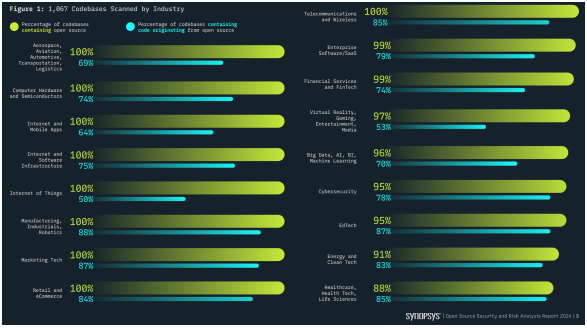
```
/**
 * @brief The basic unit of the simulation and is associated to a geographical location.
 *
 * Interventions (e.g., school closures) are tracked at this level. It contains a list of its
 * members (people), places (schools, universities, workplaces etc.), road networks, links to
 * airports etc.
 */
struct Microcell
{
        /* Note use of short int here limits max run time to USHRT_MAX*ModelTimeStep - e.g. 65536*0.25=16384 days=44 yrs.
           Global search and replace of 'unsigned short int' with 'int' would remove this limit, but use more memory.
        */

        int n; // Number of people in microcell
        int adunit; // admin unit microcell belongs to
        int* members; // array of members/hosts of microcell

        int* places[MAX_NUM_PLACE_TYPES]; // list of places (of various place types) within microcell
        unsigned short int NumPlacesByType[MAX_NUM_PLACE_TYPES]; // number of places (of various place types) within mircocell
        unsigned short int keyworkerproph, move_trig, place_trig, socdist_trig, keyworkerproph_trig;
        unsigned short int move_start_time, move_end_time;
        unsigned short int place_end_time, socdist_end_time, keyworkerproph_end_time;
        TreatStat moverest, treat, vacc, socdist, placeclose;
        unsigned short int treat_trig, vacc_trig;
        unsigned short int treat_start_time, treat_end_time;
        unsigned short int vacc_start_time;
        IndexList* AirportList;
};
```

Covid Sim (excerpt)

Len Shustek, Computer History Museum 2006

“Source code provides a view into the mind of the designer.”

Open source is the default — even in commercial software



- 96–97% of audited commercial codebases *include* open source
- ~77% of their code *is* open source (OSSRA 2024/2025)

A digital twin is no exception: models, solvers, couplers, data connectors, interfaces (e.g. Corese, OpenFlexo).

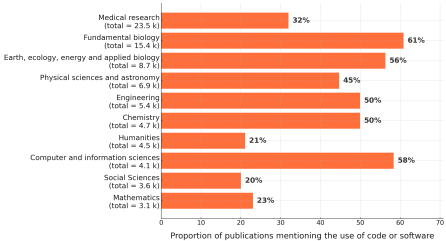
It must run, be trusted, and be reproduced — sometimes for decades.

how do we manage that source code? where does it live? what happens when it disappears?

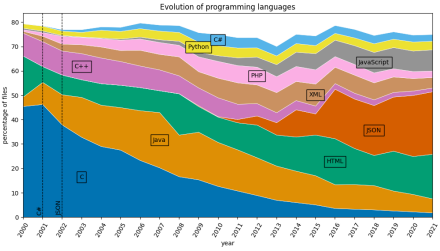
Outline

A global undertaking with great success …

Pillar of Science across all research areas



Many programming languages



From all continents

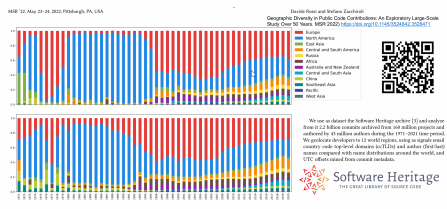
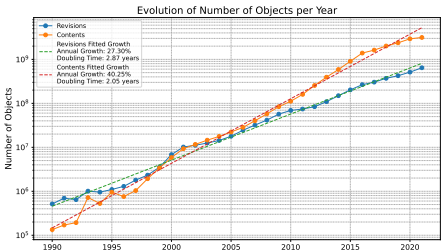


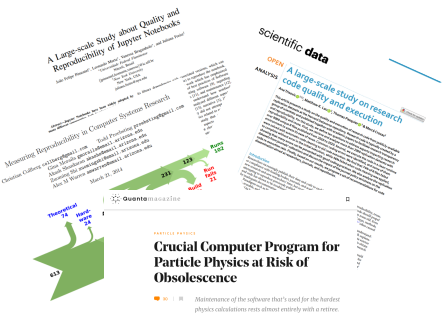
Figure 3: Ratio of commits (above) and active authors (below) by world zone over the 1971–2020 period.

An exponential growth



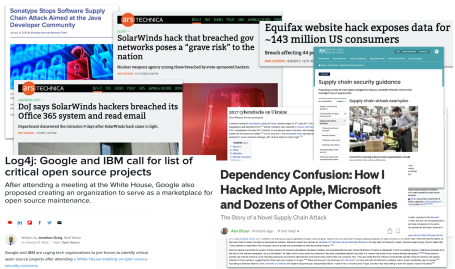
… but also, from different angles, same bad news

Reproducibility, maintenance in Academia



(articles: here, here, here and here)

Security, integrity, traceability in Industry



Can they track the software that they

- ship, use, acquire
- has that bug or vulnerability

… policy is (finally) coming …

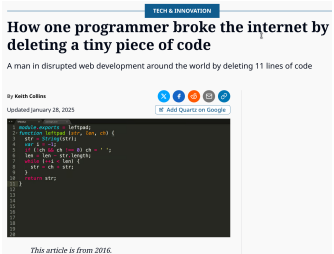
With great exposure comes great scrutiny…

… by both good and bad actors

- OSS is more and more targeted by attackers: *supply chain attacks*
  - event-stream (2018) · Log4Shell (2021) · node-ipc (2022) · XZ utils (2024)
- Increased policy attention to secure OSS, e.g.:
  - US: Biden's executive orders (2022, Jan 2025!)
  - EU: Cyber Resilience Act (CRA), progressively coming into effect
    - open source compliance work organised in the ORC WG

Breaking news: a global dependency network!

An early warning: left-pad (March 2016)



TECH & INNOVATION

How one programmer broke the internet by deleting a tiny piece of code

A man in disrupted web development around the world by deleting 11 lines of code

By Keith Collins

Updated January 28, 2025

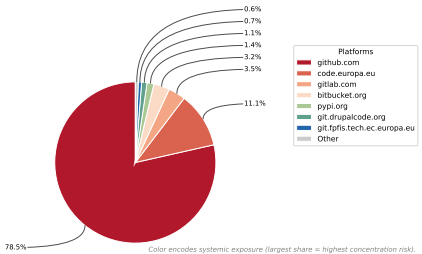
Add Quartz on Google

```
module.exports = leftpad;
function leftpad (str, len, ch) {
  str = String(str);
  var i = -1;
  if (!ch && ch !== 0) ch = ' ';
  len = len - str.length
  while (++i < len) {
    str = ch + str;
  }
  return str;
}
```

This article is from 2016.

Facebook, PayPal, Netflix, Spotify affected; web broken worldwide ~2.5 hours (Wikipedia)

A clear and present danger



Used by `europa.eu` devs *(Source: Software Heritage)*

"*If GitHub or PyPI disappeared tomorrow, our research and engineering would not just slow down — it would stop.*"

Core needs for research, cyber and resilience

1. Availability

Code must *stay accessible* — even when a registry fails, a forge shuts down, or a project vanishes.

2. Integrity

Code must be *verifiable* — unchanged across copies, forks, mirrors… and rewritten histories.

3. Traceability

Code must be *accounted for* — provenance, dependencies, SBOMs: know *what is inside* what we run.

Research, cybersecurity and resilience all rest on these *three guarantees*:
who can provide them, for all public code?

It is not enough to put code on a (sovereign) forge!

Projects *change* — rename, move, *rebase* — and platforms *disappear*:

2015: the first big bad news

Google Code and Gitorious.org shutdown: ~1M endangered repositories

- broken links in the web of knowledge (my papers too)

Big bad news keep coming in

- summer 2019: BitBucket announces Mercurial VCS sunset
- july 2020: BitBucket erases *250.000+* repositories (including research software)
- summer 2022: GitLab.com considers erasing all projects that are inactive for a year

In Academia too!

- 2021: Inria's old gforge is unplugged… breaks the Opam build chain for OCaml

We need a *universal archive* of software source code: now we have one!

Outline

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

Inria with unesco

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

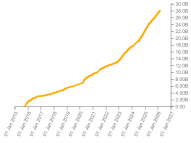
Inria with unesco

The largest open source code archive: one infrastructure, open, shared, non profit

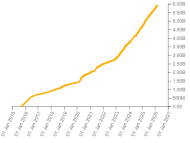
Unique digital common good *built in France since 2015*

Cultural Heritage | **Industry** | **Research** | **Public Administration**

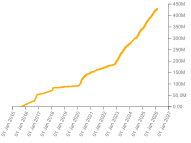
Source files
28,152,651,759



Commits
5,920,787,012



Projects
429,963,770



Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

Inria with unesco

The largest open source code archive: one infrastructure, open, shared, non profit

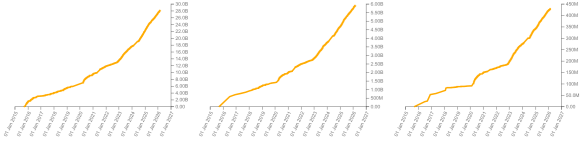
Unique digital common good *built in France since 2015*

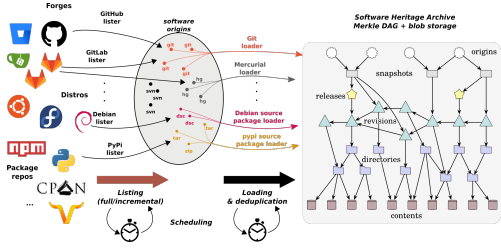
Cultural Heritage | **Industry** | **Research** | **Public Administration**

Source files
28,152,651,759

Commits
5,920,787,012

Projects
429,963,770





5000+ platforms

All versions, all history development in a single graph

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

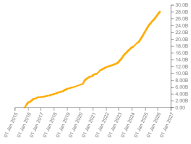
Inria with unesco

The largest open source code archive: one infrastructure, open, shared, non profit

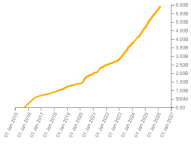
Unique digital common good *built in France since 2015*

Cultural Heritage | **Industry** | **Research** | **Public Administration**

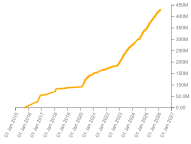
Source files
28,152,651,759

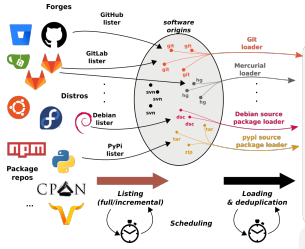


Commits
5,920,787,012



Projects
429,963,770





5000+ platforms

All versions, all history development in a single graph

- **50 × 10⁹** nodes
- **1000 × 10⁹** edges
- ~ **3 PB** of storage

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

Inria with unesco

The largest open source code archive: one infrastructure, open, shared, non profit

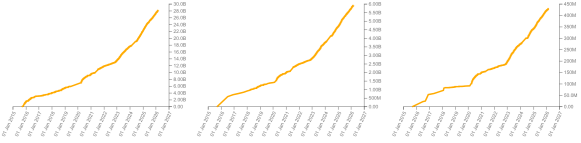
Unique digital common good *built in France since 2015*

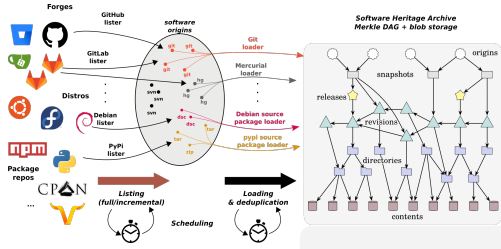
Cultural Heritage | **Industry** | **Research** | **Public Administration**

Source files
28,152,651,759

Commits
5,920,787,012

Projects
429,963,770





5000+ platforms

All versions, all history development in a single graph

- **50 × 10⁹** nodes
- **1000 × 10⁹** edges
- ~ **3 PB** of storage

A revolutionary **infrastructure** ensures **availability** guarantees **integrity** enables **traceability**

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

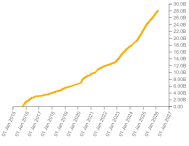
Inria with unesco

The largest open source code archive: one infrastructure, open, shared, non profit

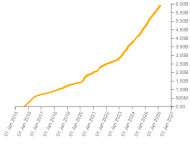
Unique digital common good *built in France since 2015*

Cultural Heritage | **Industry** | **Research** | **Public Administration**

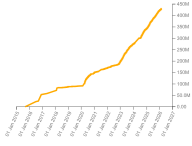
Source files
28,152,651,759

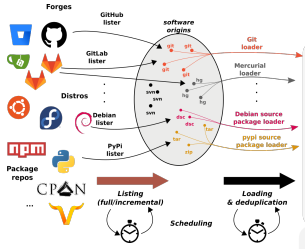


Commits
5,920,787,012



Projects
429,963,770





5000+ platforms

All versions, all history development in a single graph

- **50 × 10⁹** nodes
- **1000 × 10⁹** edges
- ~ **3 PB** of storage

A revolutionary **infrastructure** ensures **availability** guarantees **integrity** enables **traceability**

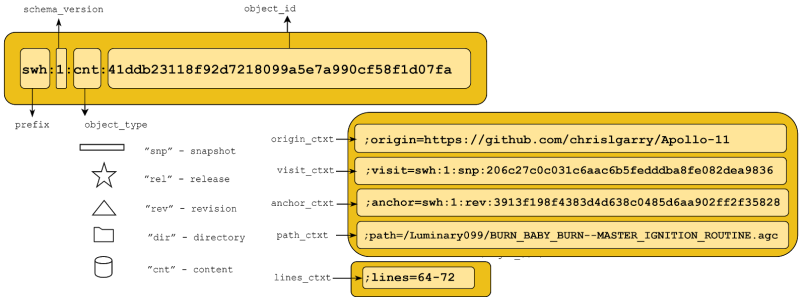
IBM Microsoft TII Technology Innovation Institute HUAWEI cnrs Infosys cea Ministère de l'Enseignement Supérieur et de la Recherche aws Google Université Paris Cité openinventionnetwork

AdaCore GitHub Université di Pisa Sorbonne Université Université de Lorraine Sant'Anna NWO université PARIS-SACLAY

Identification and integrity

Software Heritage Identifiers (SWHID)

see swhid.org



50+B
intrinsic,
decentralised,
cryptographic

Full fledged *source code references* for traceability, integrity and reproducibility

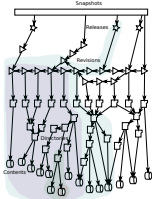
- Linux Foundation SPDX 2.2
- IANA-registered "swh:"
- WikiData property P6138

Examples: Apollo 11 AGC excerpt, Quake III rsqrt
Guidelines available, see the HOWTO

ISO/IEC 18670, see swhid.org

A revolutionary infrastructure: the graph

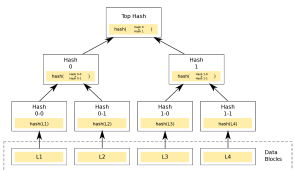
The *graph* of public software development



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 *global ledger* of public code



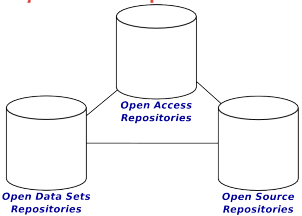
All of a software development… in a single Merkle graph!
Widely used crypto (e.g., Git, blockchains, IPFS, …)

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

the long-term memory a digital twin can anchor to

A revolutionary infrastructure: Open Science and Big Code

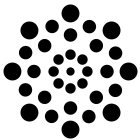
A *pillar* of Open Science



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

Browse, reference, cite — in practice

- Browse + Reference [ISO 18670] (Apollo 11 [excerpt], your work may be already there !)
- Trigger archival, use the updateswh browser extension, configure the webhooks
- Cite from the archive with biblatex-software (CTAN, ACMART)
- Describe with Codemeta (use codemeta generator)
- Curated deposit in SWH via HAL, see for example: LinBox, SLALOM, Givaro, NS2DDV, SumGra, Coq proof, ...
- Extracting all the software products for Inria, for CNRS, for CNES, for LIRMM or for Rémi Gribonval using HalTools
- Example with Parmap: devel on Github, archive in SWH, curated deposit in HAL
- Example research articles:
  - compare Fig. 1 and conclusions in the 2012 version and the updated version
  - SWHID in a replication experiment

Outline

swh-graph: bleeding edge technology

Q: Is it possible to efficiently perform software development history analyses at the scale of Software Heritage archive on a single, relatively cheap machine?

Idea

Apply graph compression techniques from the field of network analysis.

Results

The entire archive graph (~60 B nodes, *over 1 trillion edges*, June 2026) is served from ~510 GiB of RAM, traversed at the cost of tens of ns per edge (= hours for a full single-thread visit).

Paolo Boldi, Antoine Pietri, Sebastiano Vigna, Stefano Zacchiroli
Ultra-Large-Scale Repository Analysis via Graph Compression
SANER 2020, 27th Intl. Conf. on Software Analysis, Evolution and Reengineering. IEEE

Tommaso Fontana, Sebastiano Vigna, Stefano Zacchiroli
WebGraph: The Next Generation (Is in Rust)
WWW'24, the ACM Web Conference 2024

Rust and gRPC APIs available: docs.softwareheritage.org/devel/swh-graph/

From the **Software Sky** to **Institutional Insight:** Anatomy of an Observational Pipeline

Observation parameters (config file)

Stage 0: The Software Sky

The process begins with a complete map of the software universe, built from the Software Heritage archive, containing all observable software 'stars'. This stage is configured by observation parameters and a spectral filter definition (email domain selection).

Phase 1: Target Selection

Identify the stars associated with a specific institution by selecting contributors using institutional identifiers, which isolates the relevant region of the software sky for focused observation.

Quality Gate: Signal Strength Check

Before proceeding, the ploeline verifies that the selected region contains enough observable signals (software contributions) to produce statistically meaningful results.

Egicat to Novs Kiiba

Phase 2: Gravitational Cleaning (Deforking)

The process renvents gravitational artifects such as mitrers and fúres from the data to retain only the physically meaningful celestial objects (original software projects).

Phase 4: Spectral Enrichment

Each observed object is euriched with spectral data – such as activits, domains, and licenses – collected from multiple observation instruments to provide a deeper understanding.

Activity

Domains

Licenses

Adeptive Observation Scheduling

This fecture ensures the automatic relation of observation tokens to bypass issinment (PPO limits, ensuring continuous data collection.

Observation Cache

A cervivzen observation log is maintained, preventing retandant measurements and speeding up subsequent analyzes.

Phase 5: Astrophysical Analysis

The entiched observations are analysed to compute metrics, classily objects into scientific domains, and generate a synthetic, comprehensive view of the observed system.

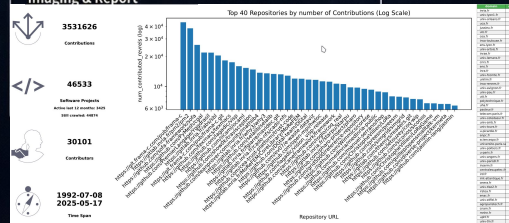
Phase 6: High-Resolution Imaging & Report

3531626
Contributions

46533
Software Projects
Active last 12 months: 3425
SWH covered: 46874

30101
Contributors

1992-07-08
2025-05-17
Time Span



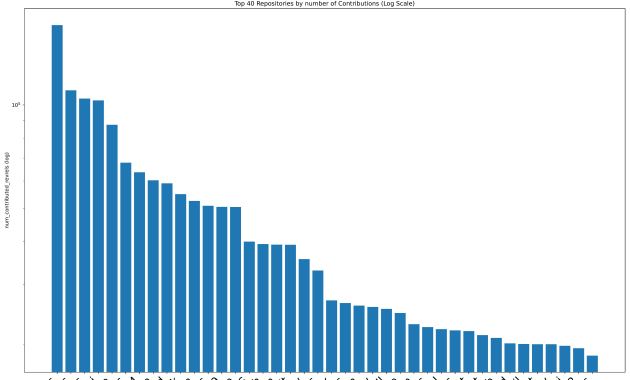
First images from the SWH "very large telescope"

13919714
Contributions

286614
Software Projects
Active last 12 months: 8238
Still crawled: 259191

92085
Contributors

1970-01-31
2026-03-06
Time Span



domains	count
gov.br	17716
gov.sk	9251
gov.au	4765
nasa.gov	3836
nih.gov	2556
cern.ch	1804
ucl.ac.uk	1827
fcl.gov	1781
nasa.gov	1758
gob.mx	1535
gouv.fr	1429
fnal.gov	1426
ucl.gov	1414
lanl.gov	1226
usgs.gov	1187
nist.gov	1163
sandia.gov	1167
gob.ar	1110
gov.cn	922
gov.fr	912
ga.ja	899
pnnl.gov	874
gsa.gov	866
cs.gov	864
gov.sg	850
nasl.gov	855
gob.pe	833
go.id	813
canada.ca	810
bnl.gov	777
nersc.gov	748
va.gov	692
cbs.gov	553
nga.gov	595
usda.gov	510
rpc.gov	507
cic.gov	490
gov.pl	483
gov.se	404
gov.ph	400
gob.ve	373
gob.hr	360
gov.my	358
gob.ec	352
gob.bo	347
admin.ch	344
gov.in	338
wa.gov	287
gob.do	282
hks.gov	273

Note: Showing top 50 of 1459 total domains.

UN member states contributions to public code, UN Open Source Week, June 2026.

Software Heritage and GNU Guix join forces to enable long term reproducibility

Connecting reproducible deployment to a long-term source code archive

Ludovic Courtès — March 29, 2019

GNU Guix can be used as a "package manager" to install and upgrade software packages as is familiar to GNU/Linux users, or as an environment manager, but it can also provision containers or virtual machines, and manage the operating system running on your machine.

One foundation that sets it apart from other tools in these areas is *reproducibility*. From a high-level view, Guix allows users to *declare* complete software environments and instantiate them. They can share those environments with others, who can replicate them or adapt them to their needs. This aspect is key to reproducible computational experiments: scientists need to reproduce software environments before they can reproduce experimental results, and this is one of the things we are focusing on in the context of the Guix-HPC effort. At a lower level, the project, along with others in the Reproducible Builds community, is working to ensure that software build outputs are reproducible, bit for bit.

Work on reproducibility at all levels has been making great progress. Guix, for instance, allows you to travel back in time. That Guix can travel back in time *and* build software reproducibly is a great step forward. But there's still an important piece that's missing to make this viable: a stable source code archive. This is where Software Heritage (SWH for short) comes in.

When source code vanishes

Guix demonstrates automated recovery using the Software Heritage archive

When a platform or registry fails… Guix falls back to Software Heritage and continues to build!

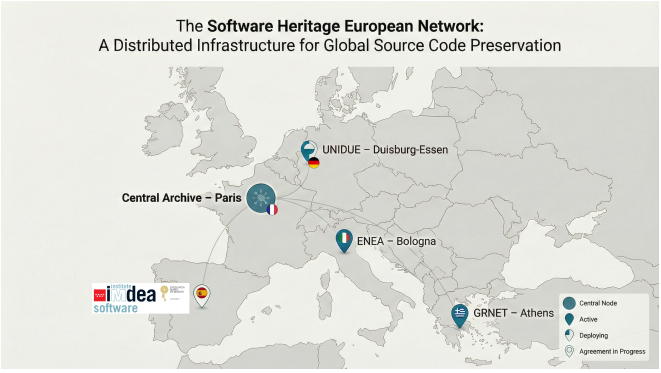
Since 2019 — *exactly what a digital twin that must rebuild in 2040 needs*

Sovereign fallback for resilient build chains

Extend to key package managers

npm, PyPI, Maven Central, Cargo, CPAN, RubyGems, Packagist, …

Realize ecumenical resilience



"When a registry fails — the build continues."

Outline

What this means for EDT

A digital twin is a *long-lived assembly of open building blocks*

For EDT to stand the test of time, these core blocks must be

- open — so they can be read, audited, fixed, and composed
- preserved — so they survive the forge they were born on
- findable — so the programme can map, reuse, and build on its own software

Here is how to make it *happen*.

The three pillars

1. Open license

The legal commons that makes reuse, audit, and composition *lawful*.

2. A shared platform

A collaborative development platform (forge): where the commons is built *together, in the open.*

3. A universal memory

An archive that preserves, references, describe and cites: these commons — *Software Heritage.*

A digital-twin engineering platform needs *all three* to be sovereign, reproducible, and durable.

- License the core building blocks under an OSI-approved open source license
- Develop in the open, on a shared collaborative platform
- Archive, describe and reference every release in Software Heritage from day one — for resilience, reproducibility, findability, citation, and sovereignty
- Treat the result as a digital commons, foundation for open innovation

A unique opportunity

Software Heritage is vendor neutral, open source, worldwide, long term — *get involved:* softwareheritage.org

Open by design is not a constraint.
It is what makes the whole greater than its parts.