The Licensing and Compliance Lab interviews Stefano Zacchiroli of Software Heritage

This is the latest installment of our Licensing and Compliance Lab's series on free software developers who choose GNU licenses for their work. In this edition, we conducted an email-based interview with Stefano Zacchiroli of Software Heritage

Software Heritage is a recently announced non-profit initiative to archive, organize, and share all publicly available software source code. Stefano Zacchiroli is a co-founder and current CTO of the Software Heritage project. He is a Board Director of the Open Source Initiative, member of FSF's High Priority Projects committee, and former 3-times Debian Project Leader.

Can you tell us a little about Software Heritage and what inspired you to create it?

Software Heritage's ambition is to be the memory of humanity when it comes to (Free) software. Our goal is to collect, preserve in the very long term, organize, and share *all* software that is publicly available in source code form. The vast majority of the corpus we target is the result of decades of work by the free software community. The rest of it will eventually (after copyright expiration) become free software too.

As for our inspiration to start the project, we were discussing seemingly unrelated topics: the cultural value of free software and the risk of losing some of it, the closed nature of state-of-the-art databases used in the IT industry to track the provenance of free software code, and the sad state of scientific reproducibility when software is used as part of scientific experiments. Software Heritage is the result of realizing that a comprehensive, curated archive of free software source code can help on all those fronts.

Can you tell us a little about the free software that powers software heritage?

The Software Heritage software stack is entirely free software. We use Debian GNU/Linux (main) on all our machines, be them bare metal servers or virtual machines. On top of them run our software foundations: PostgreSQL (knowledge base), Wordpress (main website), Phabricator (development forge), and Puppet (configuration management). Our business logic---source code crawling, ingestion, indexing, publishing on the Web, etc.---is implemented in Python 3 using several free software libraries and framework; some of those are: Psycopg2, Dulwich, Subvertpy, Celery, and Flask. All the code we develop ourselves is released under copyleft licenses (GNU AGPLv3 for public facing services, GNU GPLv3 for back-end code), with the exception of code meant to be reused in context where lax, permissively licenses are dominant (e.g., our Puppet modules, that we release under the Apache 2.0 license).

What features do you find unique about software heritage and what sets it apart from similar projects?

The field of digital preservation is luckily vast and full of initiatives that are all trying to prevent the risk of an upcoming digital dark age. The Internet Archive and the Archive Team are known to many free software hackers, but there are dozens of invaluable digital preservation projects out there. The peculiarity of Software Heritage is that we focus on source code, and that we aim at being exhaustive in that "little" niche. We love working with others though, and we are already doing so to make sure that the source code we archive can be cross-referenced with other important artifacts of software development (e.g., home pages, documentation, mailing lists, binaries, etc.) that others are already doing a great job at archiving.

As for our "style", we pride ourselves on our commitments to being a charitable initiative, to run the project openly and collaboratively, and to release all our own software as free software.

Why did you choose the GNU AGPLv3 and the GNU GPLv3 licenses for your software?

Copyleft licenses are the most natural choice for all sorts of critical software. They guarantee via legal means that not only the current version of some software is free, but also that all future adaptations of it will remain so.

Preserving the source code memory of humanity is a critical endeavor. It should be done transparently, as that allows everyone to review how the archival is being done, what safety measures are being put in place, etc. Additionally, the archived source code should be mirrored massively, as each additional independent copy reduces the chances of losing something. To that end archival software will be reused by diverse stakeholders and sometimes it will need to be adapted to fit new scenarios.

Releasing our software under copyleft licenses is first of all a contribution to operational transparency on what we are doing. Second, it creates a level playing field for everyone who will want to reuse our software, for archival or as yet unseen purposes.

How can individuals and organizations contribute?

I'm glad you asked! We're a very dedicated but also very small team, and we welcome help from all interested parties.

- Developers can participate as they usually do in free software projects: join our development mailing list or IRC channel, and dive into our code to submit bug reports or patches.
- Users can contribute by curating content on our wiki, that most notably hosts a suggestion box of endangered source code that we should archive. Raising awareness of the project by promoting it with peers and on social media is very welcome too.
- Organizations can help by becoming testimonials for what we do and by joining our sponsorship program to support our work.

What's the next big thing for Software Heritage?

Up to now we've focused on creating a solid initial corpus for the Software Heritage archive. We're quite pleased with the result: the archive already contains more than 2.8 billion unique source code files, 600 million commits, and covers more than 22 million projects. And it is growing steadily as we keep up with changes pushed to tracked version control systems (e.g., GitHub repositories).

The next big thing is content retrieval. Users can already check if we have archived source code they care about, but they cannot yet browse or download it. This is our top priority. Delivering on it requires engineering work, computing resources (e.g., bandwidth), and administrative scaffolding (e.g., processes to deal with takedown notices). In parallel with this we keep on expanding the coverage of the archive and growing a distributed network of mirrors to avoid single points of failure of any kind. It's a lot of work, but it's necessary and also a lot of fun!

Enjoy this interview? Check out our previous entry in this series, featuring Brett Smith of dtrx.