Saving the Software Heritage: the process

Extended Abstract

Laura Bussi
Dept. of Computer Science
University of Pisa
mail istituzionale

Roberto Di Cosmo
Affiliation
school
email@edu

Carlo Montangero
Dept. of Computer Science
University of Pisa
carlo@montangero.eu

Guido Scatena
Dept. of Computer Science
University of Pisa
guido.scatena@unipi.it

May 9, 2019

1 - Software is everywhere, binding our personal and social lives, embodying a vast part of the technological knowledge that powers our industry, and fuels innovation. Software is an essential mediator to access digital information and a pillar of modern research in all fields. Actually, a rapidly increasing part of our collective knowledge is embodied in, or dependent on software artifacts.

But software does not come out of the blue: it is written by humans, in the form of software Source Code (SC), which is a precious, unique form of knowledge that, besides being readily translated into machine executable form, should also "'be written for humans to read"' [1], and "'provides a view into the mind of the designer"' [3]. It is essential to preserve this precious technical, scientific and cultural heritage over the long term.

Software Heritage is a non profit, multi-stakeholder initiative, launched by Inria in partnership with Unesco, that has taken over this challenge. Its stated mission is to collect, preserve, and make readily accessible all the software source code ever written, in what we call the Software Heritage *Vault*. Software Heritage designed specific strategies to collect software according to its nature [**CACM2018**].

For software that is easily accessible online, and that can be copied without specific legal authorizations, the approach is based on automation. This way, as of April 2019, Software Heritage has already archived more than 5 billion unique SC files from over 89 million different origins, focusing in priority on popular software development platforms like GitHub and GitLab and rescuing SC from Google Code and Gitorious that hosted more than 1.5 million projects, and now shutdown.

For source code that is not easily accessible online or requires curation, we take a different approach, based on a focused search, and with significant

human intervention. The DIPISASWHinitiative is the first major step in this direction: its goal is to design and evaluate a process to rescue, curate and illustrate landmark legacy SC. This process, the SoftWare Heritage Acquisition Process (SWHAP) needs to take into account the variety of physical media where the source code may be stored, the multiple copies and versions that may be available, the potential input of the authors that are still alive, and the existence of ancillary material like documentation, articles, books, technical reports, email exchanges.

The purpose of our contribution is to present the approach we are taking with the definition, pilot implementation and experimental usage of SWHAP, in order to get feedback from the HaPoC community on our goals and methods.

First of all, we identified some high-level requirements, for preservation, curation and illustration. They should accommodate all the furnishers, be them the scavengers searching around for relevant SC or the authors conferring their own work to SWH. Coverage of both offline and online SC is also required.

1. (preservation) The furnishers should have a standard way to save the original products, to allow further refined study by future stakeholders. Therefore, there should be a place where to store safely:
   (a) in case of offline SC, the furnished SC supports (Warehouse, for later reference);
   (b) in any case, the raw digital version of the furnished SC either as is, for online SC, or after a suitable extraction from its physical support, for offline SC (Depository, for later reference).
2. (preservation) The furnishers should have a standard support to SC curation, that is, a standard environment (Workbench, for later reference) to prepare what goes into the Vault, and to support the its transfer.
3. (illustration) The furnishers should have a standard way to save documents (pictures, etc.) of historical relevance related to the recovered SC. The illustrators should have easy access to it. This entails that there should be:
   (a) a standard place where to store this information;
   (b) a standard way to index/tag it, to facilitate its later use for illustration.
4. (non–functional) The activities related to offline (legacy) SC should converge as early as possible to those related to online SC.
5. (non–functional) Any supporting implementation should be based on open and free tools and standards.
6. (non–functional) Any supporting implementation should provide support for the cooperation and coordination of the many actors playing the many roles of the acquisition process.

The most relevant choices that are driving our design of the experimental environment supporting SWHAP follow. The backbone is GitHub, currently by

far the most popular place to host open source code on the web. Indeed, it is hosting the open source projects of many major technology companies and Open Source software foundations.

GitHub is, first of all, a Concurrent Versions System, where any user can develop, curate, integrate contributions to the project, and then submit them for acceptance or revision to the project's maintainers. Besides, GitHub also offers free space to open source projects: as such it seems well suited as the free and open backbone of the multi-users acquisition process we aim at.

Moreover, GitHub is already integrated with the crawler used so far in the automated SC collection: this will facilitate the convergence to the current online SC process, with its reliable preservation features.

Guido: Ho colto il senso?

To facilitate the access to the stored items, we are evaluating the proposals of the CodeMeta Project [2], whose mission is to raise the level of the infrastructure to support the preservation, discovery, reuse, and attribution of software to that of other research products such as journal articles and research data.
To archive the ancillary documents, Wikimedia [6] is the natural choice, given its openess and freeness.

[4] non ha a che fare con le ontologie del software, ma solo con i criteri di citazione del software: metterla nelle cose ancora da fare, alla voce "'Come aggiungere una sezione *cite as* al SC salvato

8 - [Experimental use: Run the process on a representative sample of SC developed in Pisa, in view of the 50th anniversary celebration, current candidates]

maestro del trionfo della morte

9 - [Conclusions: Confident that we can demonstrate the feasibility of a process which can be exported worldwide, with UNESCO support; also, it will support the objectives of Open Science. On a more philosophical side: is the goal sound or...?]

sarebbe da citare Spinellis e GNU, ad ora assenti

References

[1] Harold Abelson and Gerald J. Sussman. *Structure and Interpretation of Computer Programs*. MIT Press, 1985. ISBN: 0-262-51036-7.

[2] CodeMeta. *The CodeMeta Project*. `https://codemeta.github.io/index.html`. [Online; accessed 08-May-2019].

[GPL] *GNU General Public License*. Version 3. Free Software Foundation, June 29, 2007. URL: `http://www.gnu.org/licenses/gpl.html`.

[3] L. Shustek. "What Should We Collect to Preserve the History of Software?" In: *IEEE Annals of the History of Computing* 28.04 (Oct. 2006), pp. 112, 110–111. ISSN: 1058-6180. DOI: `10.1109/MAHC.2006.78`.

[4] Arfon M. Smith et al. "Software Citation Principles." In: *PeerJ Preprints* 4 (2016), e2169v3. DOI: `10.7287/peerj.preprints.2169v3`.

[5] Diomidis Spinellis. "A Repository of Unix History and Evolution." In: *Empirical Software Engineering* 22.3 (2017), pp. 1372–1404. ISSN: 1382-3256. DOI: `10.1007/s10664-016-9445-5`. URL: `http://www.spinellis.gr/pubs/jrnl/2016-EMPSE-unix-history/html/unix-history.html`.

[6] Wikipedia. *Wikimedia movement — Wikipedia, The Free Encyclopedia*. `http://en.wikipedia.org/w/index.php?title=Wikimedia%20movement&oldid=895312125`. [Online; accessed 08-May-2019]. 2019.

Notes and ToDo