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
email@edu

May 4, 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.

2 - In short, 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, which is a precious, unique form of knowledge that can be readily translated into a form executable by a machine: already in the 1980's Harold Abelson wrote Programs must be written for humans to read [**?**], and more recently Len Shustek stressed that Source code provides a view into the mind of the designer [**?**]. It is essential to preserve this precious technical, scientific and cultural heritage over the long term.

3 - 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 Vault. As detailed in the CACM 2018 viewpoint article, Software Heritage has designed specific strategies to collect software according to its nature. For software that is easily accessible online, and that can be copied without specific legal authorisations, the approach is based on automation. This way, as of April 2019, Software Heritage has already archived more than 5 billion unique source code files collected from over 89 million different origins, focusing in priority on popular software development platforms like GitHub and GitLab and rescuing software source code from Google Code and Gitorious that were shutdown during the last years with more

than 1.5 million projects hosted on them.

4 - For source code that is not easily accessible online and/or that requires curation, we need 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 source code. This process 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.

5 - The purpose of our contribution is to present the approach we are taking with the definition, pilot implementation and experimental usage of this process, in order to get feedback from the HaPoC community.

First of all, we identified some high-level requirements, for both preservation and illustration. They should accommodate all the furnishers, be them the scavengers searching 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 This is to allow further refined study by future stakeholders, and entails that 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 to prepare what goes into the Vault (Workbench, for later reference).
3. (illustration) The authors/scavengers 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 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 - Gone...
7 - To realize the expressed design we choose GitHub. Nowadays, major technology companies and many Open Source software foundations host their open

source projects to GitHub, which currently is by far the most popular place to host open source code on the web.

A CVS, where anyone can develop, curate, integrate and then submit changes to the project's maintainers, is well suited for the multi-users acquisition process we describe.

State as requir.

Moreover, GitHub offers extensive free disk space for open source projects, it is already integrated with the SoftwareHeritage crawler, realizing reliable preservation and convergence to on-line SC process.

Something on Wikimedia

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

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...?]

10 - References

To be moved to Bibtex

1 Preface to Abelson, Sussman, and Sussman, The Structure and Interpretation of Computer Programs, MIT Press, 1985

2 Free Software Foundation, Inc., The GNU General Public License, Version 3, 1, 2007

3 Shustek, L. J. What Should We Collect to Preserve the History of Software?, IEEE Annals of the History of Computing, 2006