Between working repositories and cold-storage long-term archives

An inquiry into Github's Arctic Code Vault program

Adrian Demleitner
Tanzmatten 4, 2502 Biel/Bienne
https://orcid.org/0000-0001-9918-7300

Essay for the seminar

Das Internet als Infrastruktur: aktuelle Dynamiken und globale Konflikte

University of Bern
Institute of Social Anthropology
Dr. Johanna Mugler

Abstract

Github is one of the largest platforms for hosting software projects, with roughly 56 million users. They enable programmers all over the world to publish and backup their code and to collaboratively work on their projects. As an archival platform, Github is also concerned with the long-term storage of the projects they host. Part of a larger preservation strategy, the 17’000 most popular public projects were copied on film to be stored in a former coal mine – on a demilitarized arctic island in the Svalbard archipelago. The expected lifespan of these films is a millennium. Given such a period, questions of maintenance and access arise. Who will care for these archives? How will access be decided upon now, in 100 years, or even in 1000? Next to this incredible technological feat of long-term preservation; how are questions of social practices and power dynamics woven into the design of these archival infrastructures?

Keywords

Github, Software Heritage Foundation, digital archive, source code, open-source software, long term preservation

Github and its Arctic Vault program

On February 2nd, 2020 I received a badge on Github. I have contributed to the Artic Code Vault. Some of the software I have written will be preserved on a film-based technology called Piql[1]. The durability of that film is said to be up to 1000 years. The films themselves are stored in the Arctic World Archive which is located in a decommissioned coal mine on the island Svalbard in Norway[2]. Admittedly, my work that found its way into the archive is not the best in terms of quality. Why would anybody want to preserve it and what will happen to my work in the next 1000 years?

In the following, I want to analyze how and why Github is spending considerable efforts to realize its archiving strategies[3]. I will do so by comparing their public documents and underlying infrastructure with another long-term preservation strategy by the Software Heritage Foundation, a non-profit initiative supported by UNESCO.

Github positions itself as a social platform, wrapped around an ingenious little piece of software called git. Git is a distributed version control system (VCS), which enables programmers to preserve the history of the things they're working on, back up their projects, as well as share and collaborate on the same code projects. Git, one of the most popular VCS, is essential and central to every programmer's workflow. Since its inception, it has found its way into all kinds of other areas, from the academic context to artistic projects (Davis, 2015), due to its enabling of digital collaboration.

To be able to enjoy git to its fullest it needs to be installed on a remote computer, which acts as a backup as well as mediating server between all project participants.

---

[1] https://archiveprogram.github.com/, accessed March 7th, 2022

[2] https://arcticworldarchive.org/, accessed March 7th, 2022

[3] I will not be able to go into the specific reasons why Github's archive is in a decommissioned coal mine in the arctic circle, which is a topic on its own (Dodds, 2021).

Github offers exactly this as a free service, amongst other things, taking the edge of the more technical aspects of VCS. By offering a great service at no cost, as well as the ability to manage profile pages, Github became the largest social network for programmers (Begel et al., 2013). Github profiles act as a kind of portfolio for programmers, on which they can show affiliation, their projects, the technologies that they know, and crucially their skills, expertise, and style of coding. If a programmer wants to have an online presence, they need to be on Github.

Git is used to backup work in progress or finished source code, often accompanied by documentation. This act of sharing and documenting are important practices of individuation and community-building in the open-source software world (Beel & Wallace, 2020; Coleman, 2013). It's part of the scene's social practice, but also enforces and keeps the open-source ideology present. There is a focus on immediacy. Preservation is thought of in terms of practicality.

Source code as epistemology

Recent advances in the digital humanities are shifting this understanding. The Software Heritage project claims that source code, the raw material that is hosted on Github, is "the only representation of software that contains human-readable knowledge" (Di Cosmo & Zacchiroli, 2017). In their paper, the authors identify problems that prevent the proper long-term preservation of source code into accessible archives. The central idea of the project is to preserve software, respectively source code, and make it accessible for referencing and research practices. Essentially, they shift the understanding of source code from a crafts-person material to a cultural good, worthy of being preserved in the long term.

The Software Heritage project wants to create a central repository, a generous archive of all available open-source source code. They want to preserve software as well as the software's development history. To achieve that, they rely on VCS systems, such as git, but not limited to it. They developed a strong set of principles to guide the project but also their technological developments. Through

“Transparency and Free Software” as well as “Multi-stakeholder and non-profit”, for example, they vow to only produce open-source themselves without being misled by financial incentives (Di Cosmo & Zacchiroli, 2017, p. 5).

Politics and powers of archives

With these two setups in mind, we now should inquire about the politics of the power of archives. If we look at archives as infrastructures, then we have to recognize that they depend on the distinguishing of an inside and an outside. Inside meaning human actants who are considered *experts* on archival practices as well as the content of the archive. Whereas the outside can be quite generic, but generally means an interested public, which may or may not be *experts*, but engages with the material of the archive.

Jacques Derrida made a similar observation when he noted that archives need certain exteriority.

> *“As “[t]here is no archive without a place of consignation, without a technique of repetition, and without a certain exteriority”, there is “[n]o archive without outside”* (Sieber, 2016, p. 27)

In his analysis he refers to the localization of the archive, pointing towards its materiality. Not explicitly mentioned but included is the social outside. Human actants are not part of the archive’s many processes but who help sustain it, for example through financing it, delivering material, or enriching the existing. Explicitly inquiring about the social dimension is a crucial aspect in infrastructural studies (Star, 1999).

> *“Already in the ancient residences of superior magistrates filing official documents, archives, according to Derrida, “needed at once a guardian and a localization”, and thus “could neither do without substrate nor without residence”. Archives, therefore “take place in a domiciliation”, and their ‘documents’ are “only kept and classified under the title of the archive by virtue of a privileged topology”* (Sieber, 2016, p. 26)

With that in mind, let us have a look at how Github is framing its endeavor.

Analyzing Github's public documents

There are three interesting documents regarding the Arctic Vault Program. First, we have a website that serves mainly promotional material, including a beautiful video, as well as frequently asked questions. Next, Github has written a guide (Evans, 2022) that accompanies the Vault Programme. The guide was authored by Jon Evans, a novelist as well as Code Vault director. In it, we find inspirational messages as well as a technical guide on how to read and decipher the content of the films. Last but not least, Github included a document titled tech tree. This document is a "selection of works intended to describe how the world makes and uses software today, as well as an overview of how computers work and the foundational technologies required to make and use computers" (Evans, 2022).

When reading the documents provided, I felt that three things were crucially missing. What is the intention or rationale behind the efforts, who is providing the means for the program, and who exactly is addressed in the guide.

I do understand that writing for an unspecified public that will read these documents a millennium later is a near-impossible task. What struck me was that this issue was not addressed. At least to the outside, the impression is given, that all the efforts are justified by saying that the "mission is to preserve open-source software for future generations" (Evans, 2022). Returning to Derrida, the archival division of an inside and outside is especially present in Github's attempt. Github is locking away the knowledge contained in the source code. One might imagine that they are afraid of a digital dark age and uplift themselves as the sole guardians of this aspect of cultural heritage. The code is essentially inaccessible, neither physically nor digitally.

Github was acquired by Microsoft in 2018 to gain market share and bring users into the Microsoft platform[4]. The revenue brought in by the Github platform is not meager, at around 200 million in 2018[5]. Compared to the overall revenue of Microsoft with 161 billion in 2021[6], it is not that much. The acquisition of Github is also part of changing Microsoft's public image. For most of its runtime, Microsoft was actively combatting open-source approaches and projects (Coleman, 2013). Through their efforts in the last ten years, they are becoming known and trusted for being a company invested in open-source, pitting them against privacy-violating actants in the tech sector[7]. Given the ownership and revenue, it becomes clear that the provision of means for the Vault Programme has non-financial incentives, aside from direct revenue.

This provision of means fundamentally alters the intentions of the program, especially if we compare it to the Software Heritage approach. The Software Heritage clearly states three beneficiaries of their undertaking: cultural heritage, science, and industry. With those use cases comes a good idea, for whom, and for what kind of people these archival undertakings are made and what their specific needs are, how they want and could access the archives as well as what they want to do with it.

Between working repositories and cold-storage long-term archives

If we take the three initial issues together, addressees, provision of means, and rationale, it becomes quite unclear who benefits in the end. This is by large the biggest difference in the approaches of the software heritage versus the one by Github. It is certainly not the authors of the code, which is now locked away in a

---

[4] https://news.microsoft.com/announcement/microsoft-acquires-github/, accessed March 7th, 2022

[5] https://www.cnbc.com/2021/10/17/gitlab-now-worth-twice-what-microsoft-paid-for-github.html, accessed March 7th, 2022

[6] https://www.microsoft.com/en-us/investor/earnings/fy-2021-q4/press-release-webcast, accessed March 7th, 2022

[7] https://www.techrepublic.com/article/whats-really-behind-microsofts-love-of-open-source/, accessed March 7th, 2022

decommissioned mine. After this comparison, I'm left with more questions than I started with. I tried to reach out to Github to know more about the program. But after a quick response to find out if my inquiry is of public relevance, I didn't hear anything anymore.

Given that, and with little material available to analyze I need to state that many statements in this text regarding Github have to be declared as educated guesses.

It becomes obvious that Github is regarding its program through a specific lense, a way of reducing socio-technological issues to mere technical ones (Seaver, 2021), leaving out the social dimension, or actively reducing it to relevant data-points. There is little space for the general public or even the millions of authors who host their code on the platform. The question arises then if our imaginaries of copyright and authorship are ready for such a long-term endeavor?

Github made sure to only preserve open-source licensed software and is, legally speaking, on the safe side. That said, laws are also built upon our imaginaries on how societies should operate. Here I'd like to come full circle to the starting point of this text. The majority of publicly accessible and open-sourced licensed code projects have non-financial incentives. The long and intense battle for open-source software (Coleman, 2017, 2013) has shown, that there are a substantial amount of people willing to back up such an approach, although having diverse backgrounds and motivations. Within that frame, code repositories are generally understood as practical tools by those that use them, to backup projects and to enable collaboratively working on them.

This means that neither imaginaries nor practice are prepared for open-sourced code projects to be locked away for a thousand years. The important difference between Github and the Software Heritage Foundation is that the former is cold-storage preservation, meaning that the code is frozen in time and place, whereas the latter imagines long-term preservation as a working tool. Similar to Jon Evans

not being able to write for a general reader in a thousand years, programmers are equally not able to write code that would simply work a millennium later.

Github almost forcibly entangled me with its Arctic Vault program, triggering a reflexion on what long term source code preservation entails, making me wonder about how the world will deal with software a millennium from now. Code and its accompanying licenses, we might state at the end, is a type of human knowledge akin to formalized protocols that need to be actively maintained to become meaningful and not decay into empty signifiers.

Bibliography

Beel, D., & Wallace, C. (2020). Gathering together: Social capital, cultural capital and the value of cultural heritage in a digital age. *Social & Cultural Geography*, *21*(5), 697–717. https://doi.org/10.1080/14649365.2018.1500632

Begel, A., Bosch, J., & Storey, M.-A. (2013). Social Networking Meets Software Development: Perspectives from GitHub, MSDN, Stack Exchange, and TopCoder. *IEEE Software*, *30*(1), 52–66. https://doi.org/10.1109/MS.2013.13

Coleman, G. (2017). From Internet Farming to Weapons of the Geek. *Current Anthropology*, *58*(S15), S91–S102. https://doi.org/10.1086/688697

Coleman, G. (2013). *Coding freedom: The ethics and aesthetics of hacking*. Princeton University Press.

Davis, R. C. (2015). Git and GitHub for Librarians. *Behavioral & Social Sciences Librarian*, *34*(3), 158–164. https://doi.org/10.1080/01639269.2015.1062586

Di Cosmo, R., & Zacchiroli, S. (2017). Software Heritage: Why and How to Preserve Software Source Code. *iPRES 2017 - 14th International Conference on Digital Preservation*, 1–10.

Dodds, K. (2021). Geopolitics and Ice Humanities: Elemental, Metaphorical and Volumetric Reverberations. *Geopolitics*, *26*(4), 1121–1149. https://doi.org/10.1080/14650045.2019.1697240

Evans, J. (2022). *A Guide To the GitHub Code Vault*. GitHub.

Seaver, N. (2021). Seeing like an infrastructure: Avidity and difference in algorithmic recommendation. *Cultural Studies*, *35*(4-5), 771–791. https://doi.org/10.1080/09502386.2021.1895248

Sieber, S. (2016). The Politics of Archives.: Media, Power, and Identity. In S. Sieber, K. Imesch, & S. Schade (Eds.), *Constructions of Cultural Identities in Newsreel Cinema and Television after 1945* (pp. 21–38). Transcript Verlag.

Star, S. L. (1999). The Ethnography of Infrastructure. *American Behavioral Scientist*, *43*(3), 377–391. https://doi.org/10.1177/00027649921955326