OpenCollab: A Blockchain Based Protocol to Incentivize Open Source Software Development and Governance

Yondon Fu

April 26, 2017

1 Introduction

In today's software dependent society, open source software is everywhere. Parties ranging from large technology corporations like Google to individual hobbyist developers use open source software as the building blocks of their own projects. Tools that a developer once had to build from scratch are now widely available for anyone to use for free on websites like Github. Not only can anyone easily use these software packages, but anyone can also freely access, inspect and alter the source code, tailoring it for his or her own specialized needs.

The implications of democratized access to quality software is wide ranging. The open source web framework Ruby on Rails not only powers popular applications such as Twitter and Github that millions of people rely on everyday, but it also made web application development accessible to a broader audience by abstracting away the details of composing together components such as HTTP request handling, database querying and templating. Given the importance of open source software, the task at hand for technologists is to figure out how to make open source software projects sustainable such that organizations and individuals in the future can continue to rely on them in the future.

The sustainability of an open source software project is tied to project health and support. Project health is determined by how actively and adequately project developers communicate with users such that the project addresses the needs of the community. Project support is determined by the availability of financial and technical resources to develop a project[5]. A sustainable open source software projected needs to be both healthy and supported.

A healthy and supported project optimizes the use of developer time and attention, the scarcest resources in open source software projects. Communication between developers and users in a healthy project informs developers of community needs and issues to potentially focus their time and attention on. Availability of financial and technical resources in a supported project ensures developers are free to allocate their all of their time and attention on project issues. Consequently, in a healthy and supported project, developers can properly allocate their time and attention according to community needs.

If project developers fail to properly allocate their time and attention, users might leave a project in search of alternatives that better suit their needs. Canonical, the company behind the Linux operating system Ubuntu, created fragmentation in the Linux community when it shipped a new version of Ubuntu with the Unity interface rather than the standard GNOME interface[3]. Canonical's failure to properly poll for user opinion and allocate developer time and attention accordingly ultimately hurt the Ubuntu project.

1.1 Bounty Systems

A proposed solution to open source software sustainability is a bounty system for project issues such as the one operated by the website Bountysource[2]. In these systems, users can attach monetary bounties to project issues that are rewarded to contributors that successfully resolve issues. While these bounty systems may help projects attract more contributors, they do not take into account the incentives of maintainers. The work done by maintainers to review and merge in code is just as crucial as the work done by contributors. Consequently, bounty systems only partially help with project support.

Furthermore, bounty systems do not necessarily help with project health. Although in some cases, multiple users placing bounties on an issue might signal the importance the community places on that particular issue, it is also possible for malicious actors to place large bounties on issues that would negatively impact project quality. Such a possibility can place a burden on developers of filtering signal from noise and also introduces the possibility of collusion - a contributor might share a large bounty if a maintainer agrees to merge it into the codebase even if the contributed code is of poor quality. As a result, bounty systems can actualy hamper communication between developers and users leading to unmet community needs. Adding any form of financial compensation to open source software projects needs to take into align the incentives of all parties involved or else perverse incentives might arise leading to malicious behavior that harms the quality of the project.

Lastly, bounty systems operated by websites like Bountysource rely on a centralized entity to facilitate transactions. This reliance on a centralized entity not only results in a central point of failure, but can actually be more costly for users. For example, although users can freely transact within the bounty system, Bountysource charges a 10% withdrawal fee if a user wants to cash out. As a result, users choose between giving up a portion of their monetary rewards and giving up the numerous opportunies to use their monetary rewards for their own benefit outside the bounty system. This withdrawal free discourages users from leaving the system which benefits Bountysource, but harms users.

Cryptocurrencies, blockchains and smart contracts introduce the possibility of building protocols with embedded economic incentives. These protocols obviate the need for a centralized entity to facilitate transactions within a distributed network of users.

1.2 Contributions

The primary contributions of this thesis are the following:

- A command line tool that enables a decentralized Git workflow for developing open source software without relying on a a centralized service like Github (Section 3).
- A proof-of-concept blockchain based protocol to incentivize open source software development and governance (Section 4).

The motivation behind these contributions is to push the discussion on how to improve open source software sustainability. In particular, these contributions are attempts to answer the following questions relating to open source software sustainability.

- How can developers poll for user opinion on issue priorization for a project?
- How can a project attract regular contributors?
- How can maintainers be incentivized to carefully review and merge pull requests such that the quality of a project is upheld?
- The set of maintainers for a project should be able to change over time. However, maintainers might be reluctant to leave a project in fear of creating a leadership vacuum. How can incentives be structured such that people want to become maintainers?

A detailed description of these contributions can be found in Sections 3 and 4.

2 Background

Despite only recently receiving more mainstream attention among developers and business people, cryptocurrencies and blockchains have a long history. The technical foundations of cryptocurrencies and blockchains evolved from past work by computer science researchers. Two of the most important areas of research are in electronic cash systems and digital time-stamping.

2.1 Electronic Cash Systems

The advent of the Internet and the mainstream adoption of computing devices allowed people to freely transfer data between each other. A logical extension of online data transfer is online cash transfer. However, creating an electronic cash system comes with a number challenges. Similar to distributed systems, a useable electronic system also requires the ACID properties[4]:

- Atomicity: a transaction either occurs completely or does not occur at all
- Consistency: relevant parties for a transaction agree on the critical facts of exchange
- Isolation: transactions do not interfere with each other
- Durability: if one or more computers crash, the system should be able to recover to the last consistent state

David Chaum attempted to design a system with some of these properties with his DigiCash project. In DigiCash, a bank issues and verifies electronic tokens that network users can exchange with one another[4]. One of the flaws of DigitCash is that it can only offer durable transactions in exchange for a loss anonymity. Network users must present tokens to the bank for verification or be at risk of double spending attacks since electronic messages can be duplicated. Additionally, this reliance on the bank creates a bottleneck for system throughput and a central point of failure. In the scenario where an attacker gains access to the bank's private key, the attacker can create counterfeit tokens that are indistinguishable from valid tokens at will.

2.2 Digital Time-Stamping

The Internet and mainstream adoption of computing devices allowed for the widespread digitization of all types of documents. While the digitization of documents provided many benefits, it also came with the problem of how to certify the existence and time of creation or change of a digital document.

In 1991, Haber and Stornetta presented a time-stamping method for digital documents that consisted of certificates cryptographically signed by a time-stamping service. The certificates contain the hash of the document as well as linking information from a previous certificate which includes a hash of the previous certificate's linking information[6]. The result is a hash linked chain of certificates that prevents the faking of time-stamps.

Bayer, Haber and Stornetta extended this time-stamping method using merkle trees. In the original time-stamping method, verification of a document timestamp can require at most N steps by following the chain links to a time-stamp certificate that is trustworthy[1]. Instead of linking N hashes of documents, the hash values can be stored in a merkle tree. Participants can record the hashes of their own documents and the sibling hash values along the path from the document hash to the root of the merkle tree. Consequently, verification can be done in at most $\lg N$ steps by presenting the document hash and the $\lg N$ hashes on the path to the root. This modified time-stamping approach reduces storage requirements and verification time.

2.3 Blockchains

Blockchains combine learnings from previous electronic cash systems like Chaum's DigiCash and digital time-stamping methods. A blockchain is distributed ledger that is not controlled or managed by a central entity. The ledger is powered by a network of connected computers that use a consensus mechanism to reach agreement over shared data[7].

2.4 Cryptographic Primitives and Data Structures

2.5 Bitcoin

2.6 Ethereum

3 Decentralized Git Workflow

3.1 Mango

3.2 Extensions to Mango

4 OpenCollab Protocol

4.1 Protocol Roles

- **Voters**: vote for project issues by staking tokens. The weight of a vote is proportional to the number of tokens staked.
- **Contributors**: open pull requests to resolve issues by staking tokens. If a contributor's pull request is merged into the project, the contributor earns a portion of an issue's token reward.
- **Maintainers**: review and merge pull requests for issues by staking tokens. If a maintainer successfully merges in a pull request, the maintainer earns a portion of the issue's token reward.

4.2 OpenCollab Token

The OpenCollab token (OCT) powers the OpenCollab protocol. The value offered by the token is influence over an open source software project. Voters acquire tokens to signal the importance of various project issues. Contributors acquire tokens to open pull requests and earn additional tokens by making quality contributions to a project. Maintainers acquire tokens to merge pull requests and earn additional tokens for merging quality contributions into a project. Furthermore, the token is used for a number of purposes in the protocol:

- Used in a staking mechanism for issue voting. Voters stake tokens when voting for an issue. The amount of tokens staked signals the importance a voter places on an issue.
- Used in a staking mechanism for opening pull requests. Contributors stake a certain number of tokens when opening a pull request. If a contributor's pull request is closed without being merged in to the project, the contributor's staked tokens are destroyed. The possibility of losing staked tokens discourages contributors from opening pull requests unless they are confident about the quality of their contributions.
- Used in a staking mechanism for merging pull requests. Maintainers stake a certain number of tokens when they initiate a merge. Before a merge is finalized, a token holder can challenge a maintainer's merge to start a voting round. If token holders decide to veto a maintainer's merge, the maintainer's staked tokens are destroyed. The possibility of losing staked tokens discrouages maintainers from merging pull requests that do not benefit a project. The challenge and voting process for a merge is described in more detail in Section 4.5.

4.3 Voting on Issues

4.4 Opening Pull Requests

4.5 Merging Pull Requests

5 Conclusion

References

[1] Dave Bayer, Stuart Haber., and W. Scott Stornetta. "Improving the Efficiency and Reliability of Digital Time-Stamping". In: *Sequences II: Methods in Communication, Security and Computer Science*. 1993, pp. 329–334.

[2] *Bounty Source: Support for Open-Source Software*. https://www.bountysource.com/. Accessed: 2017-04-24.

[3] Jon Brodkin. *Ubuntu Unity is dead: Desktop will switch back to GNOME next year*. https://arstechnica.co.uk/information-technology/2017/04/ubuntu-unity-is-dead-back-to-gnome/. 2017.

[4] L. Jean Camp, Marvin Sirbu, and J.D. Tygar. "Token and Notational Money in Electronic Commerce". In: *Proceedings of the 1st USENIX Workshop on Electronic Commerce*. 1995, pp. 1–12.

[5] Nadia Eghbal. *What success really looks like in open source*. https://medium.com/@nayafia/what-success-really-looks-like-in-open-source-2dd1facaf91c. 2016.

[6] Stuart Haber and W. Scott Stornetta. "How to time-stamp a digital document". In: *Journal of Cryptology* 3.2 (1991), pp. 9–111.

[7] Peter Van Valkenburgh. *What is "Blockchain" anyway?* https://coincenter.org/entry/what-is-blockchain-anyway. 2017.