Introduction Open-source development began as a revolution against the high cost and often sub-par products that closed-source software development companies created.These companies were known to monopolise certain areas in IT. They were shipping software products that were buggy, less interoperable, expensive and had slow-release cycles.. Developers across the world began to collaborate, working asynchronously at times suitable for them to create alternatives that were more user-friendly, highly performant, free to use and has the backing of a huge community This was a game changer.

Open Source, Today Today, open source software products run the world Everyone relies on open source software products directly or indirectly, and the movement has forced the once closed-source companies who were anti-open-source to begin to release open-source projects as well as contribute to existing open projects. However, companies are increasingly becoming irresponsible with the way they manage users data and experiences. We believe its time for open source to move towards a new phase where its communities provide competing and innovative services that are open, transparent and driven by a user-first philosophy. Right now, open source communities mostly rally to build static products that form part of a more significant product or service offered by closed-source, equity-driven companies who rarely contribute back to the community or have questionable ethics. These open communities have little power to effect changes on the misbehaving companies that use their product. Unethical practises by companies, regulators and government motivated the creation and sustainability of Bitcoin[1] A peer-to-peer electronic cash system that allows people to send, receive and control their money without any intermediary. Bitcoin is an example of a service designed to be open, transparent, pro-users and governed by a community of people spread around the planet.

Open Source, Tomorrow The shift towards community-driven services spearheaded by open source communities is already happening in the cryptocurrency industry which is worth over $176 billion as at the time of writing this whitepaper, and it is the most prominent pioneer in the creation, incentivization and governance of systems that are created in the open with publicly disclosed mission and ethical standards. Projects like Bitcoin and Ethereum have a large community of participants who regularly hold administrators and core developers accountable to the standards they have publicly championed. If we intend to derive benefits from open source services quickly, then the tools and ideologies that make these communities successful must be adapted for communities building centralised applications which are undoubtedly the most accessible, convenient and faster than their decentralised versions. By adopting the open and collaborative structure of open source to centralised systems, making their development, governance and executions more transparent and accountable, we make it easy for communities composed of mostly mutually distrustful collaborators to build all kinds of applications (desktop, mobile, VR) together and integrate any business model like centrally-run organisations. However, for us to realise a future where open source services compete with centrally-led services, we need to find and fix the challenges and obstacles that may produce failures.

Obstacles Of Open Source Services A host of infrastructural inadequacies challenge the dream of community-led open source software services directly solving end-user problems with the same infrastructure and support that centrally-governed organisations enjoy. In this section, we describe some of the significant issues we believe to be most worrying.

Collaboration Tools & Platforms Open source developers are distributed across the world. They speak different languages, have different beliefs and live in different timezones, but these differences have not slowed down the progress of the OSS community. Most developers contributing to open source projects today use a distributed version control system (DVCS). These tools provide a common development protocol that collaborators use to contribute and manage a codebase. Most developers make use of the Git[2] to share, track, package and version their contributions. Git is a distributed version control system developed in April 2005 by Linus Torvalds to aid the development of the Linux kernel. It has since grown to be one of the most important tools utilised by developers and enterprises across the world. Git allow developers to work in a decentralised approach where they manage replicated source code repositories, contribute and resync with other developers. Unfortunately, running and maintaining independent git servers is challenging for most users Requires understanding of some networking, good internet, high-availability, server cost if hosted on the cloud. For this reason, cloud-based code hosting services like Github[3] emerged to provide users with a service that host repositories, thereby eliminating the need for users to run and maintain personal servers. These services also provide additional collaboration utilities like issue tracking, reviews, pull request that enable users to communicate problems and manage new releases of their software. Millions of developers use Github and other code sharing platforms to build great software, but the central ownership, governance tools optimised for authoritarianism and execution of the service make it unfit to host community-owned open source services.

Shortcomings of Centralized Code Sharing Platforms In this section, we highlight some of the reasons why centralised code hosting platforms are unsuitable for building true community-led, shared enterprises.

Censorship Code is free speech; similar to human languages, it is a form of expression used to communicate ideas; This has been proven in a U.S court [4] in the case between Bernstein v. the United States where Bernstein wanted to publish a paper and source code of his encryption system [5]. However, the government required him to submit his ideas for review and register to acquire a license as an arms dealer, and failure to do so would result in criminal penalties. Bernstein sued the government and won when it was ruled that the First Amendment protected software source code. However, not many countries have laws that recognise code as free speech. As a result, code sharing platforms are exposed to request and demands from governments to censor projects created by individuals of interests. It is not news that Github continues to face threats and attacks from governments[6] who want repositories removed. While code sharing platforms face a real threat from external and hostile actors, these services themselves can also exert censorship actions upon users and their projects according to predetermined or arbitrary terms which are mostly driven by their business models. Additionally, not only are users exposed to censorship behaviours incited by external agents and the code sharing platform, but they are also vulnerable to censorship by project admins or maintainers. Project maintainers are free to cut access to a repository they manage at any time and on their terms. The various levels of censorship threats make centralised code sharing platform unsuitable for hosting communities building innovative, competitive and possibly contentious applications and services. As long as contributions are hosted centrally, repositories continue to remain under potential threat.

Ownership Most tools and services on the internet support a single owner account architecture. An architecture that recognises and grants ownership and authority to one person identified by their email or phone number. Some of the motivation for this is the need to maintain security and to determine who pays the bills. Code sharing platforms are among these service providers built on this single-owner structure. All repositories must be owned and managed by one person who is granted full privileges to create, access, add members and destroy any resource associated with their account without notice. The consequence of this is that complicated, sovereign and community-led organisations cannot build software in a trust-less manner since these collaborators must trust the root account owner. One approach employed to remedy this problem involves the creation of committees and other secondary structures. However, we argue that these structures do not deliver real ownership and governance since there is a root password managed by an individual or organisation. In the crypto industry, they say If you do not own your private keys, you do not own the coins. Likewise for open source projects on a code sharing platform, If you do not own the password, you do not own the project. It is essential that contributors to community-led platforms can honestly and provably share ownership of a project without needing to trust a central authority such as the owner or maintainer of a repository or the platform operator. Real ownership of a community-owned software cannot be granted based on faith that the authority will always act in the best interest of collaborators.

Immutability The concept of immutability in computing refers to the unchanging, unalterable state of data. It refers to the inability of a piece of data to be altered. Immutable data is easy to reason about; We can make assumptions and build on top of them with certainty that their state will not change unexpectedly. Interestingly, the Git version control system used by millions of collaborators and supported by most code sharing platforms utilises an immutable data structure. Git includes concepts like branches and commits; Commits are backwards-linked collections of changes that exist inside a branch. Collaborators can begin working from any commit and never have to worry that future commits may alter the state of the commits they are extending. However, code sharing platforms do not guarantee immutability of an entire repository. Repositories as a whole are not immutable; Their owners or the platform operators can delete them. It is important to note that the ability to delete repository can serve as a tool for censorship. An account owner or platform operator can prevent people from collaborating by removing a repository from existence. Although repositories can be cloned easily, it is more challenging to re-organise a community disrupted by an act of censorship. The lack of guaranteed immutability on code sharing websites makes them unappealing for hosting community-led services. It will be devastating for a community to one day find their shared enterprise deleted by the account owner or platform operator.

Governance For open source communities to succeed in creating and managing decentralised organisations, they must first figure out governance, otherwise what would be obtainable is chaos, uncertainty, indecision, and unaccountability. These communities need to be able to formulate a governance system that is acceptable to all parties and enables them to make decisions quickly and fairly. On centralised code sharing platforms, there is only one kind of governance system One where the account owner dictates how and when collaborators interact with and contribute to the project. As a remedy, big projects delegate control to reputable, structured open-source organisations like Apache Software Foundation or the Linux Foundation. The ideal collaboration framework for decentralised organisations must take a vendor-neutral stance, only providing primitives and templates that can be used or adapted to create simple to complex governance models with enforcement carried out by the protocol where possible.

Economic Incentives People do not contribute to open source software irrationally. They do it because there is something to gain. Many people contribute because they want to learn or be members of a community of like-minded individuals. There are those who do it to improve their reputation so that prospective employers may be interested in their abilities. Companies whose business model depends on open source software are also incentivised to contribute to it.

A popular misconception about open source is the idea that open source contributions should attract no financial incentive and remain a free and voluntary activity. However, the truth is, most open source contributions impose time and financial commitments on contributors who need to develop, test, document, evangelise the product and fix bugs to ensure the product is suitable for both individual developers and enterprises. It is unrealistic to expect contributors to consistently offer their time and money to a project that gets used by others for commercial purposes and to expect no form of financial compensation as incentives to continue to remain committed to the long term improvement and sustenance of the project. If the quality of open source software is to be maintained or improved, the best way to guarantee it is by bringing financial incentives to the table.

Two of the main channels from which open source collaboration may receive financial support are code-sharing platforms or open source foundations. Code sharing platforms can create mechanisms that could potentially allow open source developers to receive financial assistance in the form of donations or subscription to premium branches, but they do not do this. These platforms do a lot to enable collaboration but not enough to allow contributors to receive financial benefits.

Open source foundations such as Apache Software Foundation can receive donations and generate income through sponsorship, merchandise sales, support and hosting conferences but those funds are for the day-to-day running of the foundation of which none of it goes down to non-employee contributors. There needs to be a trusted, transparent and automated mechanism for creating, verifying and transferring value between users and collaborators of open source software. The introduction of such a mechanism will create many types of open source economies (such as bounty, bug hunting, support services).

In the era of community-led services, collaborators need to be able to generate or receive financial rewards to support continuous development and execution of their services. While centralised code sharing platforms are unsuitable for providing the needed infrastructure, blockchain technology can allow collaborators create shared software products, formulate protocol enforceable governance structure, vote and authorise actions, receive and distribute financial incentives without intermediaries.

Execution Environment To go beyond projects like libraries, frameworks, CLI tools to community-led services, we need to begin to consider program execution environments that are suitable for running shared applications in the form of dApp, SaaS, API and other types of software. This execution model must lend itself to be deployed, operated and governed according to the governance structure instituted by the community.

Collaborators will need an environment to run tests, stage and deploy their applications. Most code sharing platforms only provide continuous integration environments with the configuration of this environment still subjected to the authorisation of the single-user account owner. There is a need for a mechanism that allows a community to collectively decide what kind of execution environment and configuration they want for their projects.

A deployment environment includes a tool that runs an application and makes it accessible to target users. The ideal deployment environment must be unalterable, autonomous and can only allow itself to be upgraded according to the governance rules of the community.

Historically, consumer applications have been executed in a centralised, managed environment but with the emergence of blockchain platforms like Ethereum, applications can now be executed decentrally on thousands of nodes in a trust-less and autonomous manner. Decentralised applications are immutable; they cannot be altered or destroyed once deployed.

While many believe blockchain applications should replace all centralised applications, we believe community-led services can leverage both types of execution environments depending on their goals; This way they are better positioned to compete with centralised services providers on all fronts.

Collaborators may decide to build completely decentralized application that cannot be altered once deployed if the nature of their value proposition demands it. Otherwise, they may simply create centrally executed applications like SaaS, API, mobile application and more. These communities building centralised applications can setup governance system that is strong enough to provide some of the qualities of decentralized applications (immutability, openness, transparent and autonomy) through enforcement of an open and transparent constitution.

The ability to build applications that can leverage the best of decentralisation and centralisation will help collaborators build useful applications that can easily gain adoption and deliver great value and experience to users.

The Ideal Platform for Open Source Service

In section 2, we discussed the reasons why centralised code sharing platforms are unsuitable to lead the change towards a future of community-led software products and businesses. These changes range from censorship, ownership, and governance to execution models. We can see that the major obstacle is the inability to enforce ownership, transparency and accountability.

We take the following approaches to solve the issues:

- The use of public key cryptography as the primary mechanism to create identity and to authorise actions, will allow projects to have multiple owners or signatories who can partake in the day-to-day governance of the project.
- By the very nature of blockchains, repositories will remain immutable: They will always be accessible to everyone. Although, owners of the repositories may enable the ability to apply access level rules to prevent unauthorised write operations.
- The introduction of autonomous procedures (a.k.a smart contracts) will allow communities to create immutable applications to enforce ownership, governance, incentive structure and disbursement and business rules. Many nodes on a blockchain network execute autonomous procedures in a way that ensures correctness and resilience to attacks that would normally cripple a centralised equivalent.
- The use of Git makes it easy for millions of developers who are already familiar with the tool to easily create repositories, contribute, import their existing projects and integrate with other git compatible tools and services they already use.
- The combination of public key cryptography and the ability to create complex governance structures, external service providers can deliver value based on the governance history or events (e.g. proposals) of a project. For instance, a community may create and vote for a proposal to run ads on Google Adwords, upon the approval, a designated service creates a campaign on Google Adwords using the information provided in the proposal.