QUIXXI: A COLLABORATIVE BLOCKCHAIN DEVELOPMENT PLATFORM

QUIXXI.COM
INFO@QUIXXI.COM

Abstract. Quixxi currently offers libraries and services that provide high-end security and analytics features for application developers. Rapid developments in distributed ledger technologies will force application developers to interact with the latest blockchain systems to ensure their products remain relevant in a rapidly evolving marketplace. In the present paper, Quixxi proposes a new collaborative developmental platform, called the Quixxi Platform, which aims to meet this emerging need by bridging the gap between community blockchain experts and traditional application developers. The platform will host service modules that utilise a range of blockchain-specific services and software libraries. Application developers can incorporate these service modules into their applications, enabling them to access the latest blockchain technologies without the burden of acquiring expertise in back-end blockchain processes. The Quixxi Platform therefore provides an abstraction layer to the developer, enabling rapid development of applications that can work across multiple blockchains. Some examples of modules within the Quixxi Platform include a Token Creator and an ERC20 Agent. These modules will enable developers to issue and manage custom cryptocurrencies from within their applications, providing new ways to engage customers and achieve efficiencies in accounting and transparency. Other modules may be used to incorporate state-of-the-art decentralised data-storage services into applications. Quixxi will develop some core modules in-house, however, the platform will also be open for contribution by external (i.e. community) developers. The latter are incentivised to contribute to the system by means of rewards denominated in Quixxi Tokens, a tradeable blockchain token that facilitates access to Quixxi Platform services. In this way, the platform serves both client developers and the larger blockchain ecosystem by rewarding community blockchain developers for their expertise and ingenuity. Quixxi believes this approach will inspire community developers to push the boundaries of what is possible and expand the frontier of blockchain services, providing a shoulder for other developers to stand upon.

1. Introduction

The Quixxi Integrated App Management System [1] currently offers libraries and services that provide high-end security and analytics features for application developers (i.e., "client developers"). Quixxi's flagship product, the Quixxi Security AppShield, is a multi-layered binary protection engine that uses military grade cryptography standards and security algorithms specifically designed to protect applications against malware, tampering, decompilation, and reverse engineering. Implementation of these essential features is a burden on developers that can be significantly lessened by Quixxi products. For a small fee, developers can easily incorporate state-of-the-art security and analytics features into their applications, ensuring customer safety and providing valuable usage data. This frees developers to focus on the core problem that their application is trying to solve, rather than peripheral functionality that is common to many applications.

A central objective of Quixxi's service is to lower the barrier of expertise for client developers, helping them to access state-of-the-art security implementations and cutting-edge technologies. Clients can rest assured that Quixxi products benefit from an expert understanding of the latest technological developments and remain up-to-date, thereby abstracting the necessary maintenance away from the client developer. With this in mind, the present paper proposes a new collaborative developmental platform, called the Quixxi Platform, which aims to benefit both client developers and community developers by providing a new way to interact with emerging technologies such as blockchains and cryptocurrencies.

Recent years have seen a rapid rise in technological advancements. In particular, blockchain technologies have revolutionised the way in which value is created, stored, and transferred, in addition to offering a myriad of new organisational possibilities. The introduction of decentralised autonomous agents represents a fundamental change in the way that information can be stored and delivered. Cryptocurrencies such as Bitcoin [2] and LiteCoin [3] revolutionised the world of financial transactions by introducing trust-free decentralised transactions, and are increasingly being adopted by mainstream financial institutions. For example, UBS and Santander are among a number of institutions that have adopted the Ripple Transaction Protocol [4], which allows them to trade cryptocurrencies and benefit from a shared ledger [5]. Organisations such as the Ethereum Enterprise Alliance [6] demonstrate the keen interest that large corporations display in these emerging technologies. SWARM [7] (built in conjunction with Ethereum [8]), Sia [9], and IPFS [10] promise to revolutionise data storage and retrieval,

while a host of other services, with similarly groundbreaking ideas, are in the pipeline. These technologies are poised to radically change the expectations and needs of modern application developers. In particular, we envision a need to utilise these technologies without acquiring a detailed technical understanding of the underlying service and associated security considerations. Although we can't predict the full extent of the revolution that will be unleashed by blockchain technologies, one thing is clear - customer needs are changing, and application developers will need rapid access to the latest developments to ensure their products remain relevant in a rapidly changing marketplace.

The Quixxi Platform will be specifically designed to help client developers stay abreast of the latest technologies by making it easier for them to incorporate cutting-edge services into their applications. The platform will provide a range of blockchain-specific services and software libraries that abstract-away the detailed operation of the underlying systems, allowing client developers to enhance their applications through interactions with blockchains such as Ethereum, Bitcoin, and Zcash [11]. It will also offer customised application-specific services to meet the growing demand for other blockchain-related applications.

Interacting with the aforementioned decentralised technologies requires payment of fees, usually in the native currency of the platform. To facilitate abstraction of these payments away from client developers, the Quixxi Platform will rely on a new internal token, named the Quixxi Token (QXE). Client developers will use QXE to pay ongoing usage fees or subscriptions for static libraries and/or dynamic services offered on the platform. Developers will pre-pay and maintain a balance of QXE in order to utilise features of the Quixxi Platform, with the option to "top-up" when necessary. In this way, the Quixxi Platform will abstract-away transaction costs, inter-blockchain complexities, and various operational fees into a consolidated, exchangeable token (i.e., QXE). This proposed model of payment will initially supplement the existing Quixxi subscription model with the potential to completely replace it in future.

In addition to helping client developers incorporate the latest blockchain technologies into their applications, the Quixxi Platform will also contribute to the broader blockchain ecosystem by rewarding community blockchain developers that contribute blockchain services to the platform. Community developers can submit software service modules to Quixxi, which will be vetted in detail by the Quixxi team, before being added to the Quixxi Platform as endorsed community services. These community-developed service modules will be available to all client developers using the platform. As community service modules are used by client developers, Quixxi will remunerate the author of the community service a portion of the fees collected from module users, in the form of QXE. This will incentivise community developers to contribute to both the Quixxi Platform and the blockchain ecosystem as a whole. In particular, experienced blockchain developers will be incentivised to help other less experienced developers benefit from their unique knowledge and skills. This approach allows Quixxi to support community developers with expertise in emerging technologies by providing a platform for them to publish their work and ensuring they are rewarded for services they develop.

> *The Quixxi Platform will have a positive impact on the blockchain ecosystem by subsidising community blockchain developers while simultaneously bridging the divide between client developers and the adverse technical challenges inherently present in emerging technologies such as blockchain.*

The layout of this paper is as follows. The Quixxi Platform is described in Section 2, along with a list of example blockchain services. Section 3 describes the community involvement inherent in the platform and outlines the fee structures and community rewards. The Quixxi Token is discussed in Section 4 and conclusions appear in Section 5.

2. The Quixxi Platform

In this paper we refer to the Quixxi Platform as the overarching system which enables client developers to easily interact with, and integrate, various blockchain services and libraries. An overview of this system is provided in Figure 1. It facilitates the use of complex features and emerging technologies by client developers without the overhead of developing the requisite expertise. The platform consists of three main layers, namely the application wrapper layer, the services layer and the service support layer, as described below. While some libraries and services will be offered for free, others will require payment. To facilitate this, an accounting sub-layer is also necessary.

2.1. Application Wrapper Layer

In order to provide value in the current application development ecosystem, the Quixxi Platform will provide an easy-to-use, well-documented API[1] layer that will "wrap" modern mobile and/or desktop applications. Applications developed within this wrapper will benefit from the inclusion of software libraries and resources such as those that are currently available in the Quixxi SDK. Developers will choose which libraries and services they would like to include in their

[1]Application Programming Interface

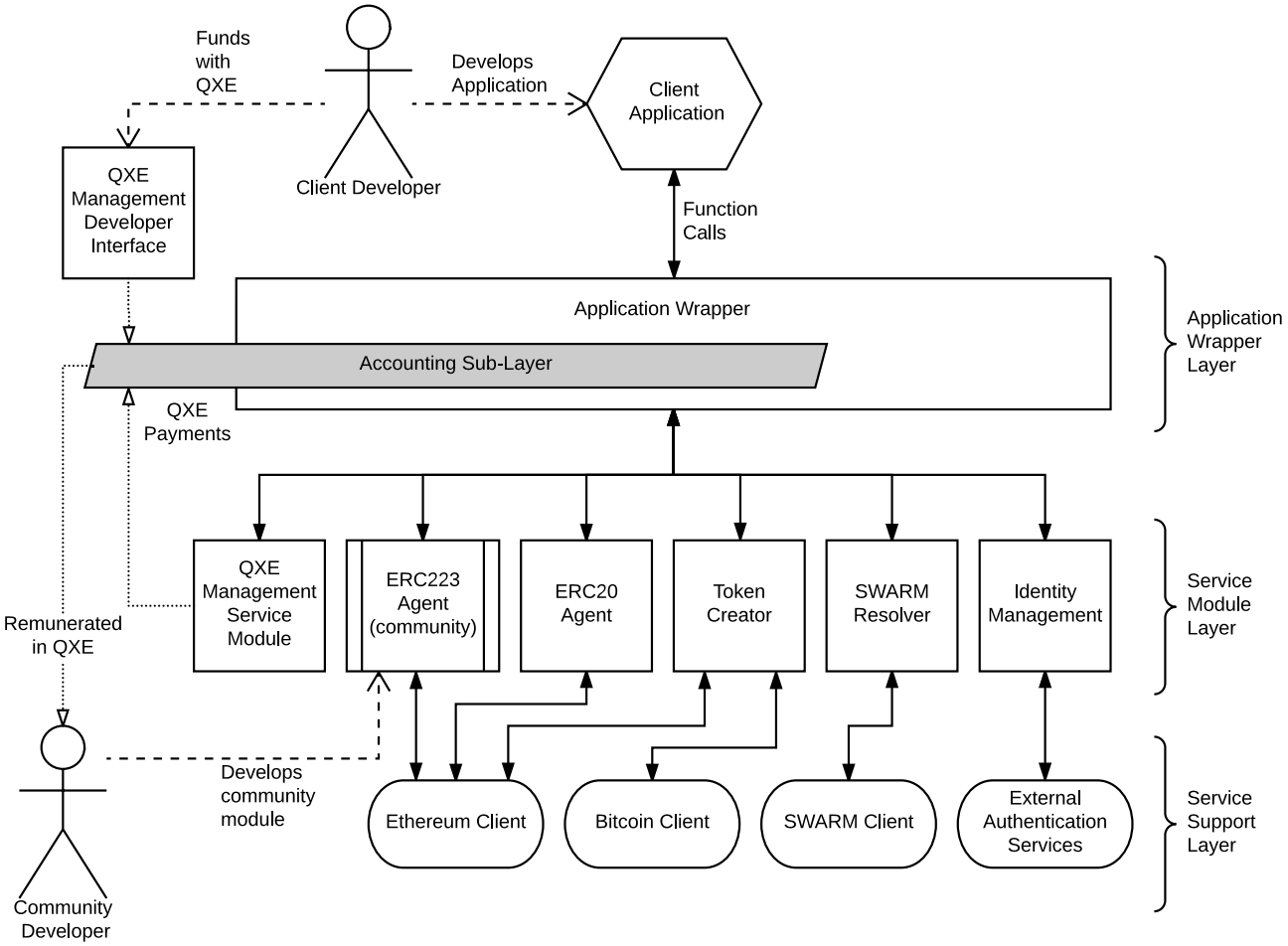

FIGURE 1. The Quixxi Platform is separated into three conceptual layers, the *Application Wrapper Layer*, the *Service Module Layer*, and the *Service Support Layer*. *Client Developers* can build applications within the *Application Wrapper* in order to access features provided by various *Service Modules* (some examples are shown). Community developers can contribute additional *Service Modules* which extend the feature set available in the *Application Wrapper*. Payments for services are made using a newly proposed Quixxi Token (QXE) by either the Client Developer or their end users. Community Developers are remunerated for their contributions in the form of QXE.

wrapper, and applications will interact directly with this layer to access the features of the Quixxi Platform.

2.1.1. ACCOUNTING SUB-LAYER

Within the application wrapper layer exists a sub-layer which provides accounting for the libraries and resources utilised by the application. This layer will ensure that either the developer or user of an application (configurable by the developer) has sufficient QXE to pay for any non-free service modules, such as those which require per-usage fees to be paid to a blockchain.

While some modules may require a one-time fee (for example, static software libraries), other modules will need to charge a fee every time the module is used. This fee may be payable by the application developer, or may be passed directly to the user. The fee charged by various modules must be constantly adjusted in order to account for changes in the operating environment. For example, as the Bitcoin network becomes congested, the fees charged for Bitcoin transactions may increase.

To ensure the development experience is as simple and consistent as possible, all fees and charges are denominated in QXE. Price changes for services will be managed in a consistent way to ensure developer expectations are managed properly.

2.2. SERVICE MODULE LAYER

This layer hosts an array of services and libraries to be consumed by the developer and/or end user. The types of services available on the platform will be tailored to the current and emerging needs of the development community. In the platform's initial state, it will focus primarily on blockchain-related technologies. Blockchains are an emerging and rapidly developing

technology, making it difficult to predict the range of applications that will be of interest in the future. Accordingly, the range of service modules offered by the Quixxi Platform will be subject to technological developments, particularly given the community involvement inherent in the platform.

Preliminary services offered on the platform will be based on existing blockchain applications. However, the platform is purposely designed to inspire the blockchain community to push the boundaries of what is possible and develop service modules that anticipate and meet emerging customer needs. Quixxi aims to serve application developers by helping them make great applications while simultaneously supporting the blockchain community by providing a platform for community developers to publish their work.

Some example service modules are described in this section. However, we emphasise that the list is not exhaustive but rather representative of the type of applications that blockchain technologies can support. The platform is larger than the specific service modules described here - it provides a distribution framework that allows community developers to offer arbitrary state-of-the-art application services. A few example services are described below.

2.2.1. Token Creator

The units of value transferred in cryptocurrencies (such as, bitcoin and ether) are generated as a reward for work done by the operators of the individual platform (e.g. the miners for proof-of-work based systems). Holders of these tokens can transact value and/or pay for computational power to run programs on the blockchain (i.e. the tokens are spent as transaction fees that fuel computations).

Users wishing to access the service features available on a given blockchain must hold these tokens (in this case cryptocurrencies) to take advantage of the technology. These first-generation cryptocurrencies provide a simple template for a wider range of possible token services. The basic idea is clear - the creator of a new service may wish to issue tokens that define how users can access their service. When issued via a blockchain, such tokens automatically benefit from decentralised trust-free transactions. This approach can introduce business and accounting efficiencies for service providers, in addition to offering novel ways to access services and engage customers. Thus, even traditional services that do not directly rely on blockchain technologies, such as supermarkets or retailers, may seek to offer tokens in their applications to provide new ways to interact with customers, offer simplified transaction mechanisms, or introduce cost-savings via accounting efficiencies. Essentially anything of value can be "tokenised" on the blockchain and transacted digitally.

To illustrate with an example, one can imagine supermarkets issuing self-branded tokens that may be exchanged for discounted items, or retailers may offer tokens for early purchasers of soon-to-be-released products. Families could even offer tokens to reward children for doing their chores - the possible applications for this technology are abundant.

The Quixxi Platform's Token Creator service will allow client developers to create and issue blockchain tokens (on various blockchains), and mediate transactions, all from within their application. The Token Creator can be used as a stand-alone service by client developers whose blockchain requirements do not extend beyond token issuance and management, or by client developers seeking to include tokens alongside a host of other blockchain-related services. Quixxi's Token Creator service module thus brings the modern instruments of cryptocurrencies and blockchain tokens into the hands of application developers, without requiring them to understand the complexities of each token-issuing subsystem.

2.2.2. ERC20 Agent

Whereas the Token Creator service allows developers to issue their own tokens in accordance with individualised service needs, the ERC20 Agent service allows token-holders to manage and transfer their Ethereum-based tokens via the Ethereum blockchain.

Developers can incorporate easy-payment cryptocurrency transfers, either as stand alone applications or embedded within larger service structures, without the overhead of acquiring blockchain expertise. The ERC20 Agent allows blockchain users to transfer cryptocurrency/tokens via the ERC20 software interface [12], which is currently the standard interface for tokens issued on the Ethereum blockchain. Using the ERC20 Agent is simple and allows client developers to give users control of their cryptocurrency assets as part of their engagement with the developers brand, thereby extending the level of service and simplifying life for their customer. For example, one could envision banking applications that incorporate the ERC20 Agent to allow bank customers to manage their cryptocurrency holdings alongside traditional banking services. The ERC20 Agent would help organisations incorporate the new technology alongside existing infrastructure.

Similar service modules can be offered for other blockchain ecosystems. The Quixxi Platform is not tied to a particular blockchain system and will offer service modules that add value for developers, regardless of which blockchain ecosystem supports the service. Modules that simplify user-engagement with Bitcoin, colored coins (tokens issued on the Bitcoin blockchain), and Zcash will also be supported.

2.2.3. The SWARM Resolver

SWARM [7] is a distributed storage platform and content distribution service, designed to provide decentralised data storage and allow platform users to efficiently pool their bandwidth and storage. SWARM

showcases the power of blockchains to radically change the business model for existing technology service providers. Instead of paying centralised storage providers to house large amounts of content, or host popular content, as is common today, SWARM provides a decentralised service that allows users to take advantage of storage capacity on network nodes while simultaneously incentivising nodes to host popular content, as required to meet network demands. Furthermore, decentralised storage and distribution means content producers need not transfer content ownership to a centralised entity in order to meet their service needs.

The SWARM Resolver allows client developers to incorporate SWARM's decentralised storage services into their applications. Client developers can offer application users the ability to distribute popular content or satisfy their storage needs, safe in the knowledge that their data is not housed by a centralised entity and that they retain ownership of their content. The SWARM Resolver brings breakthrough blockchain technology to developers' fingertips and into the mainstream.

SWARM is not the only organisation developing decentralised blockchain data storage services. Organisations such as Sia and IPFS also aim to use blockchains to provide new ways of storing and retrieving data. The SWARM Resolver service provides a concrete example of a blockchain-specific service module that offers data storage and retrieval features, however, we emphasise that the Quixxi Platform will be compatible with service modules offering alternative data storage solutions.

2.2.4. Identity Management

Consistent and manageable identity on the blockchain is somewhat of a "holy-grail" technology. The Identity Management module doesn't aim to address all identity-related issues on the blockchain, but through the integrated collection of services is capable of mitigating the problem. This service can aid client developers in securely storing users' private keys (should the developer's application need to store private keys), and facilitate the re-issuance of keys should a password or key-file be lost or stolen. It can centrally store a database of known users based on public keys and their associated identities. This service may also include KYC services, a common issue in the blockchain application space.

Where appropriate, the Identity Management service (or an alternative service within the Quixxi Platform) will support established blockchain identify platforms such as uPort [13].

2.2.5. ERC223 Agent - Community Example

The blockchain space is rapidly evolving, and the specifications and protocols are currently being developed. The burden of keeping up with these specifications will lie with the Quixxi Platform and not the end user developers. One classic example is the Ethereum token application interface. A new standard is being formalised, known as ERC223 (an extension of the ERC20 interface), however, it is still some time away from becoming widely accepted. As this new interface is adopted, Quixxi will either develop a modernised ERC223 service or incentivise community developers to construct an ERC223 service for the Quixxi Platform. The ERC223 service provides an example of how the community could be involved to ensure the latest blockchain developments are available to application developers without a strong reliance on Quixxi's internal development cycle.

2.2.6. Services Summary

The above service modules exemplify the power of the Quixxi Platform to make blockchain technologies accessible to application developers. Service modules such as the Token Creator, the ERC20 Agent, and the SWARM Resolver draw on existing blockchain services and illustrate the potential for blockchain technologies to radically expand the possibilities available to application developers. Modules such as the ERC223 Agent rely on upcoming blockchain developments but serve to illustrate the power of the Quixxi Platform to incorporate emerging blockchain technologies and applications.

The Quixxi Platform is not localised to current technologies or even to current blockchains. As new chains are adopted, Quixxi will implement services and features to accommodate the needs of developers, ensuring they have access to the latest technologies. For example, as services of interest to developers appear on new blockchains, Quixxi will operate or support nodes for the new blockchain, along with an API that allows developers to interact with the chain. When individual blockchains add new features or complexities, Quixxi will add support for these and, if possible, abstract or distil them into an easy-to-use API for developers.

2.3. Service-Support Layer

The final layer of the Quixxi Platform manages the interaction between service modules and public decentralised systems, such as blockchains. As Quixxi is blockchain agnostic, the interaction with numerous blockchains is a necessity. The platform will operate multiple nodes with multiple clients (should the chain permit) for robustness and throughput. The services layer will interact internally with this blockchain layer in order to perform various functions on behalf of the applications utilising the Quixxi Platform. The number of nodes in this layer will scale with the usage of services, ensuring fast, real-time transactions with dynamic transaction costs, all handled internally by the Quixxi Platform.

3. Service Module Marketplace

The Quixxi Platform hosts service modules that provide client developers with simple tools that increase application functionality and improve the end user's experience. In order for the Quixxi Platform to meet the rapidly evolving needs of client developers, the development of service modules must be decentralised. Quixxi believes that many great innovations arise from community efforts. The Quixxi Platform not only accepts external contributions but actively encourages and incentivises their development. While Quixxi will develop a range of service modules in-house, external developers who contribute service modules to the Quixxi Platform will be rewarded with Quixxi tokens (QXE) (see Section 4).

Developers first submit their service modules to the Quixxi team, who scrutinise the code and ensure the proposed product is tested and deemed sufficiently secure to be included on the Quixxi Platform. Here, Quixxi will leverage its expertise in application security to vet products and ensure platform users are only offered high quality, secure service modules. Once a module has been approved it will be added to the platform as an endorsed community service. Quixxi clients can access the community service module in exchange for QXE as usual, however, a portion of the fees paid will be directed to the development team for that service module.

3.1. Service Module Fee Structures

Fees for the use of service modules are charged according to one of two methods, depending on the type of service:

Subscription: Some service modules will be offered via a subscription fee, whereby clients pay a regular fee to access the service for a designated period of time. Services offered under the subscription payment structure can be used an unlimited number of times within a designated period - QXE payments ensure unlimited access within the allocated time frame.

Per-use: A per-use payment structure is required if a service makes regular calls to blockchain applications that incur fees on the blockchain. Customers pay Quixxi in QXE for use of the service, and the acquired QXE are used, in part, to fund payment for the service on the blockchain.

For example, the SWARM Resolver allows users to store data via the decentralised SWARM storage platform. Access to the SWARM platform incurs a small on-chain fee on the Ethereum blockchain, payable in ether. Quixxi will charge client developers a usage fee for access to the SWARM Resolver, payable in QXE, and will use the QXE to fund on-chain expenses incurred by the service on behalf of clients. Similarly, on-chain expenses incurred by service modules contributed by community developers will be covered by the usage fee paid by client developers for service access.

3.2. Rewarding Community Developers

Community developers charge client developers a set subscription or usage fee for access to their service module. This fee is payable in QXE. After the below expenses are subtracted by Quixxi, all remaining QXE received as payment for use of the community developer's module will be passed on to the developer as reward for their contribution.

Community developers must choose whether their service module is offered under a subscription fee structure or a per-usage fee structure, depending on whether the service involves static libraries or regular blockchain calls that incur on-chain fees.

Community developers should also take the following expenses into consideration when determining the fee structure for their modules:

- Quixxi will charge community developers a flat percentage fee of the total payment they receive for service modules offered on the Quixxi Platform. This fee is estimated at ten percent. The fee allows Quixxi to maintain and develop the Quixxi Platform in accordance with both community developers' needs and client developers' needs.
- Quixxi will vet service modules provided by community developers to ensure they are secure and of sufficiently high standard to appear on the Quixxi Platform. Expenses incurred by Quixxi during the vetting process ("vetting costs") will be recovered by charging community developers an additional fee.
- Community developers will have the option of paying the vetting cost upfront but may, in some instances, be offered to pay the "vetting fee" in an ongoing manner. The vetting fee would be set at a fixed percentage of the service module fee and would be subtracted from QXE payments received for the given service module. The vetting fee would only be charged for a fixed number of service uses, until the vetting costs were recovered.
- The precise dynamics involved in recovering the vetting cost will be a function of the complexity of the given service module and the strength of the business case for the service module. Quixxi will offer an upfront option or a vetting fee option in accordance with their assessment of the given service module. Community developers are assured, however, that if they are more optimistic than Quixxi, with regards to the strength of the business case for

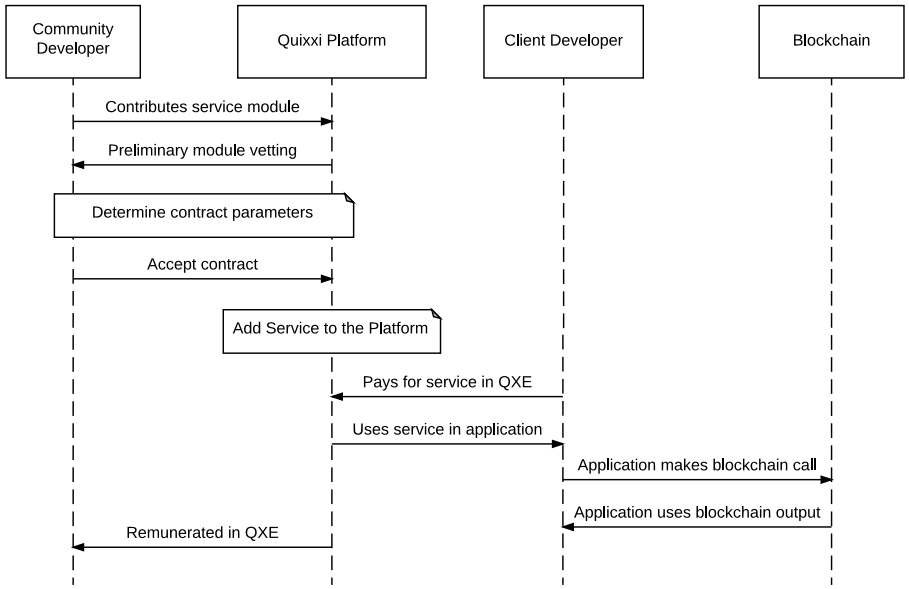

FIGURE 2. Procedural overview of the interaction between developers and the Quixxi Platform. Blockchain expenses incurred while making on-chain calls from Service Modules are covered by the QXE fees, as described in the text.

their service module, they will have the option to pay the vetting cost upfront and make their module available on the Quixxi Platform, subject to meeting Quixxi's security and quality standards.

A procedural overview of the interactions between community developers, client developers and the Quixxi Platform is shown in Figure 2.

3.2.1. Payment Structure Example

As a concrete example, for the case of a service module charging client developers a per-usage fee, the above fee structure is succinctly summarised by the following equation:

$$R_n = P - f_Q \times P - B_c - F_V \times \theta(C_V - n \times F_V) \ , \quad (1)$$

where:

- R_n is the reward received by a community developer when their service module is used for the nth time.
- P is the per-usage payment made by an application developer to use the service module.
- f_Q is the fraction of the payment amount P that Quixxi charges community developers to host their service module on the platform. The Quixxi fee is estimated at 10%, giving $f_Q \approx 0.10$.
- B_c denotes any on-chain expenses incurred by the service module. If the service module does not incur on-chain expenses then $B_c = 0$.
- $F_V = f_V \times P$ is the vetting fee, charged as a fraction f_V of the per-usage payment P.
- $C_V = (V_c - V_u)$, where V_c is the total vetting cost and V_u is the amount of the vetting cost paid upfront by the community developer. Thus, C_V is the portion of the vetting cost outstanding when the service module is added to the platform.
- $\theta(x)$ is the Heaviside step function, defined as:

$$\theta(x) \quad = \quad \begin{cases} 0 & x < 0 \\ 1 & x > 0 \end{cases} \quad (2)$$

Thus, the vetting fee F_V is charged for each use of the service module, until the number of uses satisfies $n > n_V \equiv C_V/F_V$, at which point the vetting cost is recovered and the final term in Equation (1) vanishes (i.e., the vetting fee no longer applies). In cases where the full vetting cost is paid upfront, $V_u = V_c$, the final term in Equation (1) vanishes for any $n > 0$ (i.e., no vetting fee applies).

As evidenced by the above payment structure, the Quixxi Platform provides a bridge between client developers, who want to offer application services to their customers, and community developers, who seek to benefit from their expertise in the latest blockchain technologies. Quixxi believes the platform will inspire client developers to produce high-quality applications and simultaneously incentivise community developers to expand the forefront of blockchain services.

3.3. The QXE Management System

The QXE Management System will integrate closely with the Accounting Sub-Layer (details in section 2.1.1) and handle all QXE transfers, ensuring seamless payment for Quixxi Services. Client developers can interact with the system via the *QXE Management Developer Interface* to deliver subscription or usage service fees for access to service modules.

If an application regularly interacts with a blockchain and incurs on-chain token fees, the developer can include the *QXE Management Service Module* in their application. This allows application users to manage their tokens and exchange them for the use of services on the Quixxi Platform. Community developers can also create a wallet through the QXE Management System to manage their QXE tokens. The QXE Management System supports both on-chain and off-chain QXE payments as detailed in section 4.1.

4. The Quixxi Token(QXE)

Quixxi Platform services are accessed by the transfer of QXEs to Quixxi. There are three main reasons for introducing Quixxi Tokens:

- The Quixxi Platform will include both blockchain-related services and non-blockchain services. Blockchain services require interfacing with blockchain networks and may incur on-chain expenses payable in various tokens or cryptocurrencies. QXE is readily used for this purpose. Combining blockchain and non-blockchain service modules under a single payment structure simplifies payment dynamics and introduces accounting efficiencies.
- The Quixxi Platform is not bound to a particular blockchain ecosystem but can instead incorporate service modules that interface with arbitrary blockchains. Accordingly, it is appropriate to introduce a payment token that can be employed *across blockchain ecosystems*, rather than wed the platform to a particular pre-existing cryptocurrency/token. Although technologies such as Ethereum and Bitcoin offer the most-promising blockchains at present, it is unclear which technologies will come to dominate this space in the future. The Quixxi Platform will therefore be constructed on the basis of present-day blockchain technologies, while retaining a conceptually independent existence as an entity that is blockchain-agnostic, as far as the branding, marketing and user-experience is concerned. This is important, in terms of future-proofing the platform, as it allows the platform to readily incorporate new technologies in the future, while maintaining consistency across established branding and marketing strategies.
- Community developers will be rewarded for contributing service modules to the Quixxi Platform. Issuing rewards in QXE ensures a homogeneous accounting structure for the payment of service access fees and the incentivisation of external contributions to the platform. Homogeneity of accounting services permits additional accounting efficiencies.

To access blockchain-related service modules that incur on-chain expenses, client developers (or users of their applications) will be required to pay ongoing per-usage fees. The precise details of this fee structure will be module-dependent and, accordingly, will be determined as modules are developed by either community developers or Quixxi. On-chain expenses are determined by the blockchain protocols and are largely beyond the control of both community developers and Quixxi. A per-usage fee is preferable to ensure that client developers whose applications make infrequent on-chain calls can offer their services at cheaper rates than developers whose applications require frequent on-chain services.

A consequence of this fee structure is that it automatically creates a floor price for QXE. Platform users transfer QXE to Quixxi for access to service modules that use on-chain services. Quixxi must therefore hold a reserve of cryptocurrencies such as ether, bitcoin, or zcash to pay for on-chain services on the corresponding blockchains, such as Ethereum and Bitcoin.

As just mentioned, service modules that incur on-chain fees will be offered under a per-usage fee structure on the Quixxi Platform. In general, on-chain fees are payable when a transaction is included in a block by a miner. For example, computational services provided on the Ethereum blockchain incur an expense in "gas", an on-chain concept that provides the "fuel" for computations, similar to the use of fuel by a motor vehicle. The amount of gas required to fuel a given computational process is fixed by the design specifications of the Ethereum blockchain [8]. However, blockchain users can incentivise miners to include their transactions in the blockchain by offering a larger ether/gas price, in accordance with market demands.

Service modules that incur on-chain costs will similarly involve a QXE/use price that accounts for the QXE/gas price. Modules may include service features that allow users to select the price they are willing to pay for gas, reflecting the urgency/importance of their transaction. For example, during a period of high network activity, when supply and demand forces tend to elevate the gas price, users may be willing to pay an elevated gas price to incentivise prompt processing of an urgent transaction. For a less urgent transaction, on the other hand, users may accept a delay in processing time, in exchange for the lower gas price that suffices to incentivise transaction processing after peak network-activity subsides. Services modules can be designed to meet these needs by offering discrete choices of QXE/gas rates, continuous choices of QXE/gas rates, or simply offering a fixed QXE/gas rate for all usage of the service. Depending on the sophistication of the service user, the service developer may provide detailed service features for the user or account for these matters behind the scenes,

perhaps merely requesting basic user-input to indicate the user's preferences and adjusting the usage fee accordingly. Service users hold a wallet of pre-paid QXE tokens which are used to fuel on-chain computations or fund subscriptions.

4.1. QXE implementation

While QXE tokens are designed to be blockchain agnostic, it is a reality that these tokens must exist on a particular blockchain in order to exist as a decentralised cryptocurrency. QXE tokens will be deployed on the most appropriate blockchain as the requirements of the platform evolve.

At the time of writing, the Ethereum blockchain appears to offer the greatest flexibility and security for issuance of tokens, therefore the first iteration of QXE will be implemented as an ERC20 [12] smart contract on the Ethereum blockchain. As the ecosystem evolves, it may be possible that Ethereum no longer remains the most suitable platform for QXE, at which time Quixxi tokens will be migrated to an alternative platform. For example, if the cost of QXE transfers on the Ethereum blockchain becomes too great (measured in ether or time-to-confirmation), then a transition from ERC20 tokens to a native blockchain may become appropriate. Such a transition would not result in the creation of additional tokens – tokens would be removed from one system and re-issued in the alternative.

Since transfers of QXE on the Ethereum blockchain cost "gas", denominated in ether, this could present an unreasonable cost overhead for service payments on the Quixxi Platform. To avoid excess gas expenditure on Ethereum while still paying incrementally for use of service modules in the Quixxi Platform, off-chain payments mechanisms will be implemented so that fees can be paid without the overhead of blockchain confirmation. In this way, fee payments can occur rapidly and without costing ether, followed by regular on-chain settlement.

5. Conclusion

The Quixxi Platform provides client developers with access to the latest blockchain technologies and services. The platform offers service modules that interface with blockchains, allowing developers to leverage the latest technologies without requiring expertise in complicated back-end blockchain processes. This extends the range of services offered by Quixxi and frees developers to keep doing what they love, namely making great applications that improve people's lives. The platform also supports community developers and the larger blockchain ecosystem by incentivising community blockchain developers to stay abreast of the latest technologies and develop service modules that can be distributed via the Quixxi Platform. This is made possible by the introduction of QXE, a blockchain token required to access Quixxi Platform services and used by Quixxi to reward community blockchain developers for their contribution.

References

[1] Quixxi. https://quixxi.com.
[2] Satoshi Nakamoto. Bitcoin A Peer-to-Peer Electronic Cash System. 2009.
[3] LiteCoin. https://litecoin.info/.
[4] Ripple. https://ripple.com/.
[5] Nathaniel Popper. The rush to coin virtual money with real value. *The New York Times*, 2014.
[6] Ethereum Enterprise Alliance. https://entethalliance.org/.
[7] Viktor Tron, Aron Fischer, Daniel Nagy, Zsolt Felfoldi, and Nick Johnson. Swarm: swap, swear and swindle. *SWARM*, 2016.
[8] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. 2015.
[9] Sia. https://sia.tech/.
[10] IPFS. https://ipfs.io/.
[11] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Mathew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous paymets from bitcoin. 2015.
[12] ERC-20 Token Standard. https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20-token-standard.md.
[13] Christian Lundkvist, Rouven Heck, Joel Torstensson, Zac Mitton, and Michael Sena. uport: A platform for self-sovereign identity. 2017.