SUBS

Automatic payment for self-custodial wallet

Protocol Whitepaper

V1.0 October 2022

Ferdinand ATTIVI

ferdiattivi@gmx.com
pinonesteban@gmx.com

Abstract

With the emergence of SAAS, subscription and recurring payment systems have taken on unprecedented importance. However, several issues are surfacing such as lack of transparency from banks or service providers, high intermediary fees and other. A peer-to-peer version of the Subscription system allows online auto payment for self custodial wallets without a third party. It offers a more decentralized and efficient way for subscribers to access content or services, and can provide greater control and flexibility for both subscribers and providers. This can be achieved through the use of decentralized technologies, such as blockchain and smart contracts. We propose a solution to the problems of current subscription systems using blockchain technology. Subs is the first liquid system that allows peer-to-peer subscription and recurring payment for self-custodial wallets on blockchain. This document describes the definitions and theory behind the Subs Protocol explaining the different aspects of the implementation.

Contents

1 Introduction

The creation of the Subs protocol marks the advent of a decentralized subscription system that allows subscribers to access content or services directly from the provider without the need for a third party. App providers create subscription applications and their payment system in the subs contract. Simultaneously, in the same contract, the subscribers can subscribe to these applications while keeping their funds in their wallets. The subscribers simply give the right to the protocol through the regulators to trigger the payment according to the rules of the provider (every month, every week, every day...). The regulators are contracts or external actors who will come to regulate the payments of the subscribers to automate the payments and then earn rewards. A simplified scheme of the protocol presented in figure 1 below.

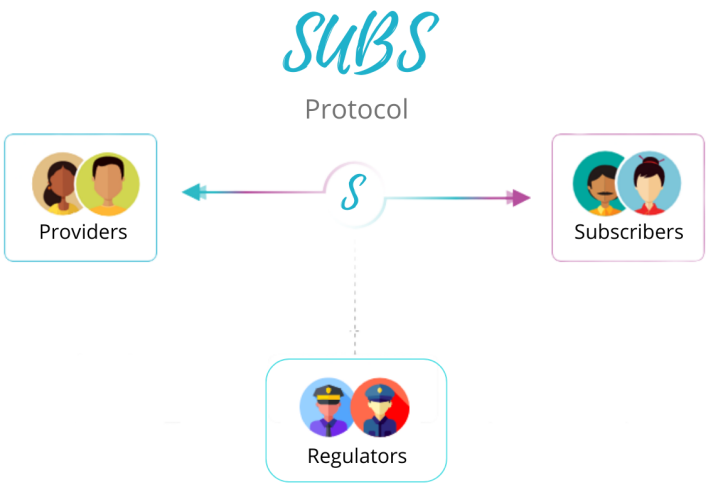

Figure 1 : The Subs Protocol

The automation of payments is decided by the protocol and all these actors ::

- For app providers, they can easily create a complete subscription and automatic payment system by providing their own rules.
- For subscribers, they can subscribe and unsubscribe to an application created by the app providers according to the rules provided by them
- For regulators, they regulate the protocol by automating or disabling subscriber payments according to the app providers' rules to earn a reward higher than their transaction costs.

1.1 Basic concepts

Subs protocol is a system that allows for the **creation** and **management** of subscription-based services on a blockchain called Subs Apps. App creators can create these services with automatic and recurrent payment systems. They can accept any number of payments with any number of **ERC-20** tokens in their app. In this type of system, subscribers can recurrently pay for access to services or content, and the **payments** are processed and recorded by the regulators.

Each application created has its own payments systems, its own **ownership** system to manage it and its **liquidity** that the creator can manage with the protocol.

Each payment system linked to an application has different attributes such as the type of **period** (monthly, yearly, daily, weekly, ...), the **tokens** it accepts as payment, the **minimum time** for which its user must approve its token in order for it to be taken every duration.
A user can subscribe to an application through these payments by paying the amount due with the tokens required by the payment. The user does not need to block these tokens in the protocol, he just needs to approve these tokens that will be taken according to each period defined by the application provider.

All these payments per period (monthly, daily, ...) defined by the app creator are made by the regulators when this period is **due**. The regulators automate the payment of the users to earn **rewards** in exchange. In case the user has no funds in his wallet, or any other reason for not paying his subscription, he will be given a time period calculated according to the period of his subscription to settle his payment. If this period is **over**, his subscription is deactivated by the regulators and it will be up to the user to renew his subscription.

After creating their application, creators have access to a large panel of components that they can just copy and paste onto their front-end application without coding. They can also let users subscribe to the decentralized subs application if they don't have web applications.

1.2 Related Work

This section aims to explain the different players involved in recurring subscription models and how they currently fail to provide a recurring subscription without trust and locking funds into a contract for everyone. It also discusses how Subs is addressing these issues and aims to provide a solution.

SuperFluid [1]

Superfluid is a smart contract framework on EVM networks, enabling everyone to move assets on-chain following predefined rules called agreements. Superfluid enables constant token flows on-chain. Money streams will continue perpetually until cancellation or the senders balance runs out. Money streams can also have their stream rate updated at any time. But with superfluid, the user cannot pay with any token, only a few tokens are pre-chosen and the user must wrap these tokens on the SuperFluid protocol before using them. This makes the user's tokens illiquid. **Subs** provide a system that anyone can use without a smart contract or web3 development knowledge base. All tokens can be accepted and defined by the application designer on a simple and flexible interface. No need to deposit the tokens in a contract or wrap it. The user keeps his token in his wallet and can use it as he wishes.

DESU [2]

Desu is the social platform where creators can monetize their content with cryptocurrency, social tokens, and NFTs. Desu is for only content creators. The create subscription function creates a smart contract between the creator and the subscriber. The smart contract transfers the payment/tokens into the Desu smart contract, tokens start getting transferred from the subscriber to the creator once every month. **Subs** are not only usable by content creators. It can be used by anyone who wants to create a subscription or recurring payment system with these tokens. The subs protocol is totally liquid, which means that the user does not need to deposit the funds in the contract.

Unlock [3]

Unlock is a protocol for memberships as time-bound Non Fungible Tokens. It is a protocol developers, creators, and platforms can use to create memberships. Unlock's goal is to ease implementation and increase conversion from "users" to "members," creating a much healthier monetization environment for the web. In short, Unlock is an open-source, collectively owned, community-governed, peer-to-peer system that creates time-based memberships. Unlock Labs created Unlock Protocol to provide an open, shared infrastructure for memberships that removes friction, increases conversion, enables scale, reduces costs, and evolves the web from a business model built on attention toward one based on membership. **Subs** can be used as a membership system where subscribers pay payments to follow their creators, but it is not limited to that. It all depends on the purpose of the application creator. Subs is automated by its regulators and can be used as a recurring payment system.

Coinbase Commerce [4]

Coinbase Commerce is a service offered by Coinbase that allows merchants to accept cryptocurrency payments on their websites. It is designed to be easy to use and allows merchants to accept payments in various cryptocurrencies, including Bitcoin, Bitcoin

Cash, Ethereum, and Litecoin. The service can be used to create subscriptions, which are recurring payments made by customers at regular intervals. To use Coinbase Commerce for subscriptions, merchants will need to integrate it into their website and configure it to create and manage subscriptions. This can typically be done using the Coinbase Commerce API or by using one of the available plugins or integrations offered by Coinbase. Once the integration is set up, merchants can create subscription plans and offer them to their customers. Customers can then sign up for a subscription and make payments using their preferred cryptocurrency. Coinbase Commerce handles the process of collecting and processing the payments, and the funds are deposited into the merchant's Coinbase Commerce account. With **subs** the application creator can accept all existing cryptos, he will not be obliged to accept only those proposed by a third party. With subs, users do not need an intermediary to store and manage their payments. The subs user does not need to know how to use an api to integrate subs in his application. Also the recurring payment is managed by the subs protocol itself, not by the user to implement it in his application.

All of these applications are superb and solve problems of their own, so this part is not intended to denigrate them as they also inspired the creation of Subs.

2 Protocol Architecture

The current implementation of the protocol is as follows :

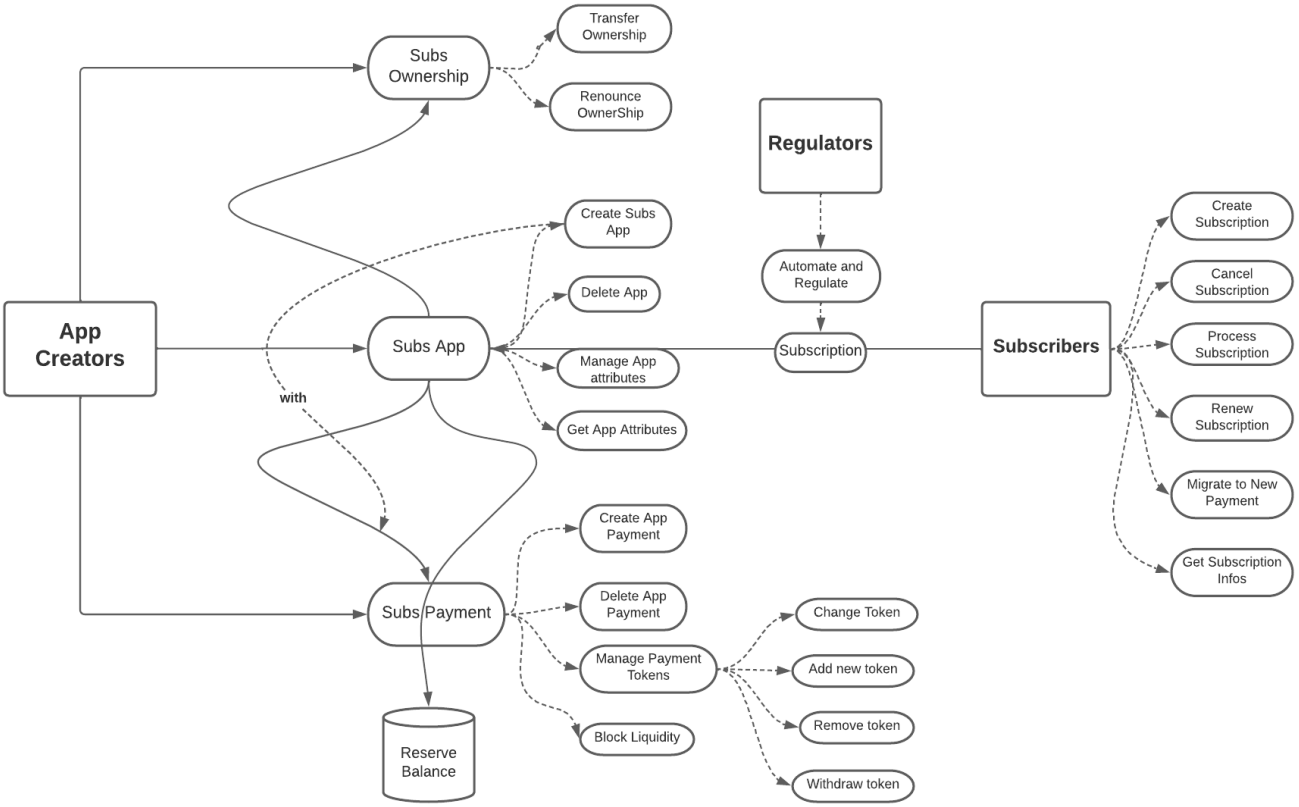

Figure 2 : Protocol Architecture

2.1 Subs Core

The Subs Core contract is the center of the protocol, it contains the two important parts of the protocol which are :

- The Subs App, which is the part that allows creators to create their subscription or recurring payment system applications in the protocol.
- Subscriptions, which is the part allowing users to interact with the created applications and regulators to regulate the protocol

These different parts include other parts that we will see later.

2.2 Subs Apps

In the subs protocol, anyone can create a recurring payment application for free, be it a subscription system or any other use case. To create an application the user provides different properties such as: the **name** of his application and his **payment system.**

The subs app contract is derived from the core contract and offers these different actions:

- Create App
- Modify App
- Delete App
- Get App properties

App creators can do these different actions to create and manage their application in the protocol.

2.3 Subs Payments

Each application has a payment system that contains different properties. When creating his application, the user fills in these properties. We can find property like : the **name** of the payment, the **type of payment** (Only ERC-20 tokens are accepted for now in the protocol, in the future, native coins will be directly accepted without going through a wrap), also the **tokens** the creator is willing to accept as a means of payment and their quantity, a **trial period** (optional, only if he wants), the **type of period** (second, minute, weekly, daily, monthly, yearly) and the **minimum period** for which the users of their application must approve their token. The application creator can therefore interact with his payment system through different actions :

- Add New Payment
- Delete Payment
- Change Payment
- Add New Token
- Remove Token
- Block liquidity
- Withdraw Token

The creators of applications can perform these different actions to manage their payment system without a third party.

2.4 Subs Ownership

Each application also has an ownership management system that allows creators to manage their application securely. The creator of the application can therefore **transfer** the ownership of his application, or **abandon** it completely if he wishes. This makes the

protocol more flexible and opens it to other use cases. Let's imagine a case where an application creator wants to sell his application for example. The application creator can therefore manage his ownership system through different actions :

- Transfer Ownership
- Renounce Ownership

2.5 Subscription

Subscribers can interact with the contract made by the creator of the application through this following actions:

- Create Subscription
- Process Subscription
- Automate Payment
- Cancel Subscription
- Renew Subscription
- Migrate To New Payment

2.6 Subs Liquidity

The Subs protocol has different liquidity reserves. Each application in the protocol has a balance. This is the cumulative total of all assets that this application may have received. In parallel to the balance of all the little applications, the protocol itself has its own global balance made up of fees paid by the users interacting with the applications of the protocol. :

The fees in the protocol are defined by default but can be changed by the governance mode. The fees paid by each user are different depending on the payment system of the application. [f]

The reserve of the protocol allows on one hand its development and on the other hand a collateral allowing to refund the creators of applications in case of security problems in the protocol.

3 The Subs App Contract

Users can perform certain actions within Subs App Contract that allow them to interact with the protocol to create and manage their application. Please note that not all of the checks in the contract are present on these drawings. Only the most important ones are shown.

3.1 Create App

The action of creating an application is very simple, the user needs to specify the information mentioned above[2.2]. The action sequence is as follows:

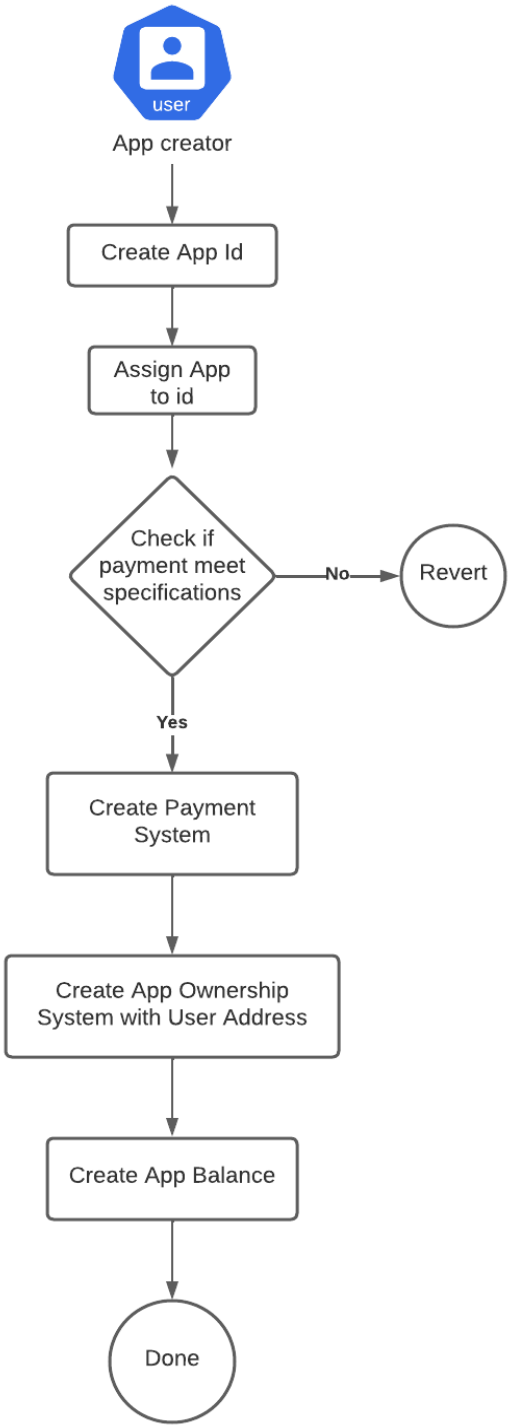

Figure 3 : Create App

3.2 Delete App

The action of deleting an application is very simple and does not have any particular state check. The action sequence is as follows:

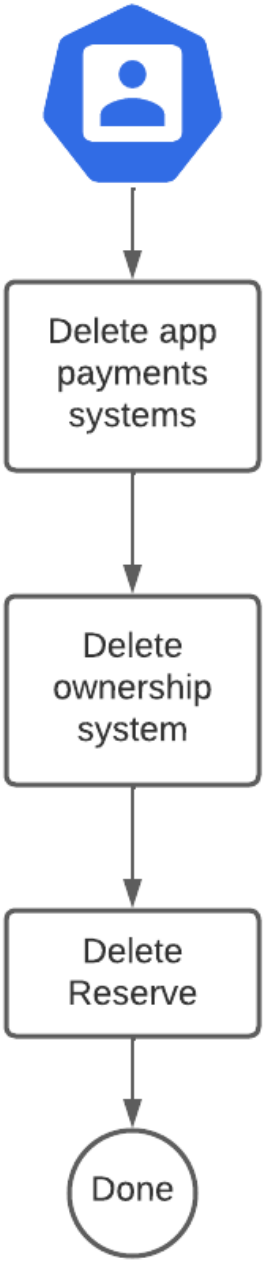

Figure 4 : Delete App

3.3 App payments

As mentioned above[2.3], each application has a payment system created automatically when it is created. Different actions can be done on a payment. The sequence of these actions is as follows:

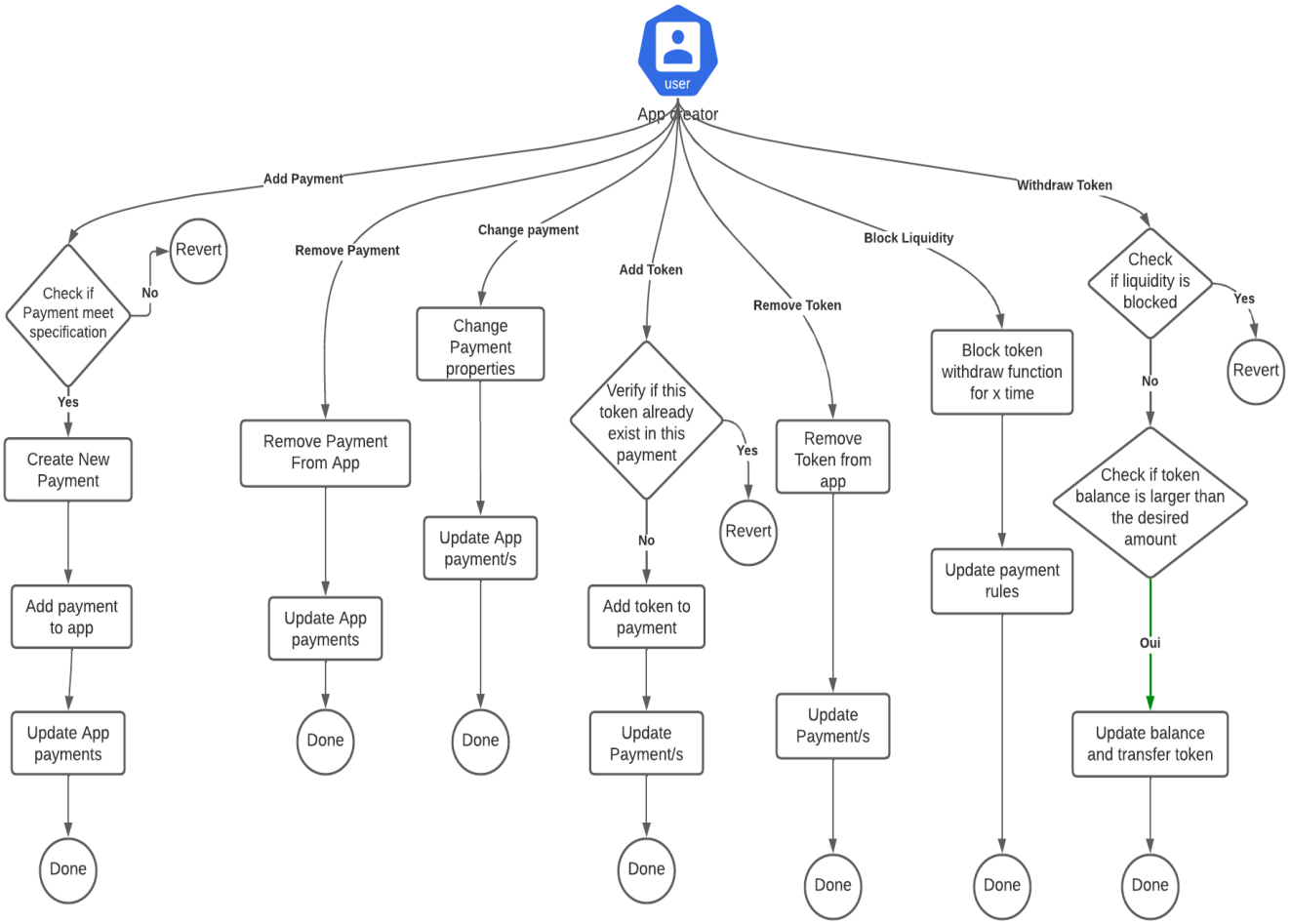

Figure 5 : App Payments actions

3.4 App Ownership

Ownership actions allow creators to transfer ownership of their application to a new valid address or null to abandon the application.

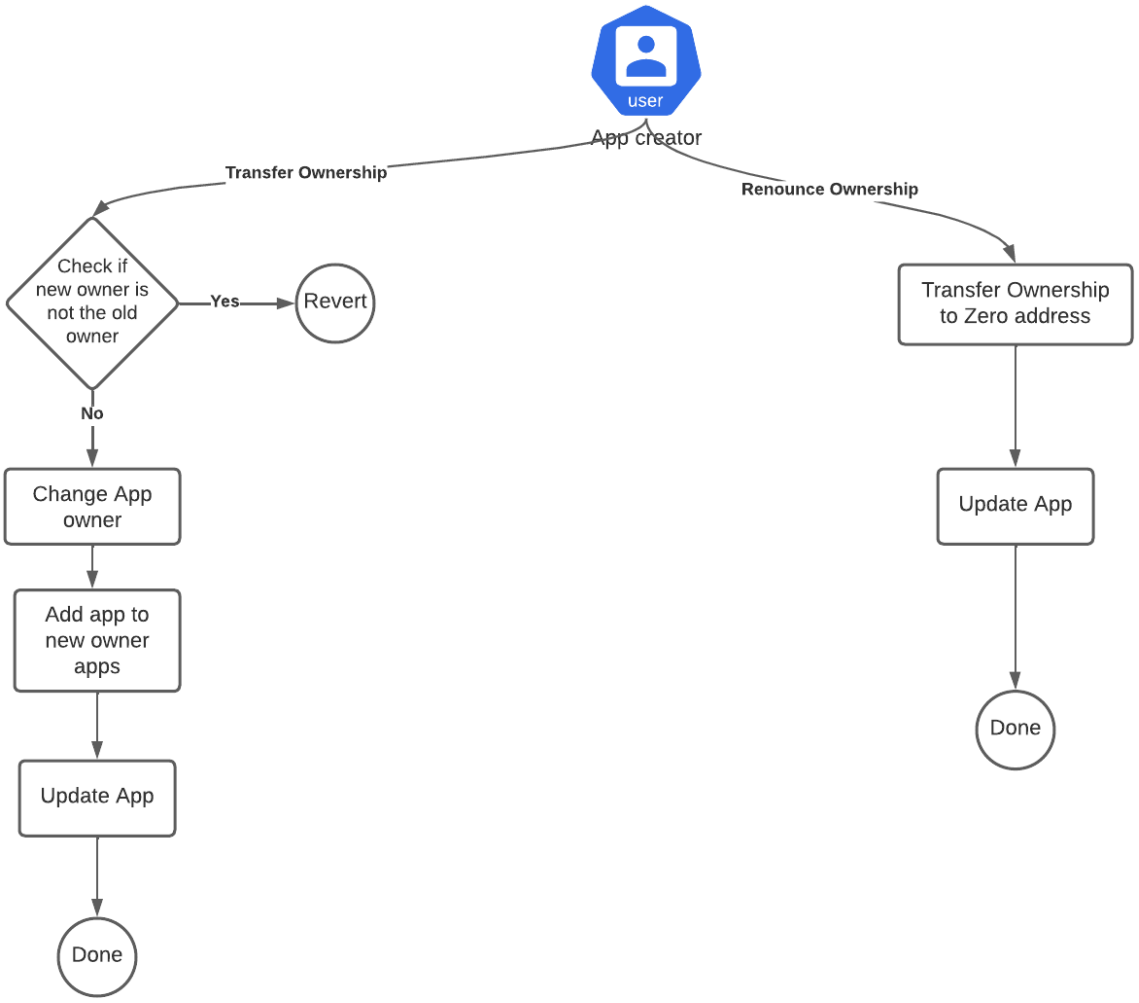

Figure 6 : App Ownership actions

4 The Subscription

Subscribers can perform certain actions within the subscription contract that allow them to interact with the protocol and creators' applications to subscribe, unsubscribe, and perform other operations on those applications.

Please note that not all of the controls in the contract (steps or requirement)are present in these drawings. Only the most important ones are shown.

4.1 Create Subscription

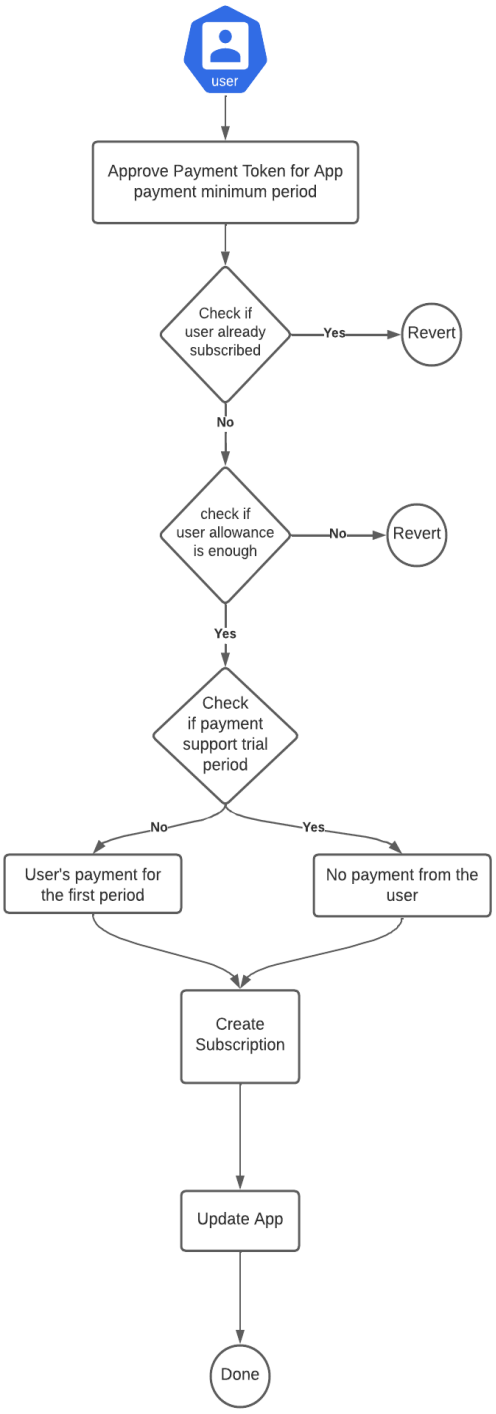

Figure 8 : Create Subscription

This action allows the user to subscribe to an application created by the creators for a recurring payment without blocking funds. To do this, the user just needs to approve his ERC-20 tokens and pay for the first period. The payment is then automatic and recurring every period defined by the application he subscribed to.

4.2 Cancel Subscription

This action allows the user to unsubscribe to an application created by the creators and delete his subscription to this application. If the user just wants to unsubscribe without deleting his subscription he can just call the revoke function (equivalent of the approve function with the amount 0) without calling the **cancel** function and thus be able to resubscribe later.

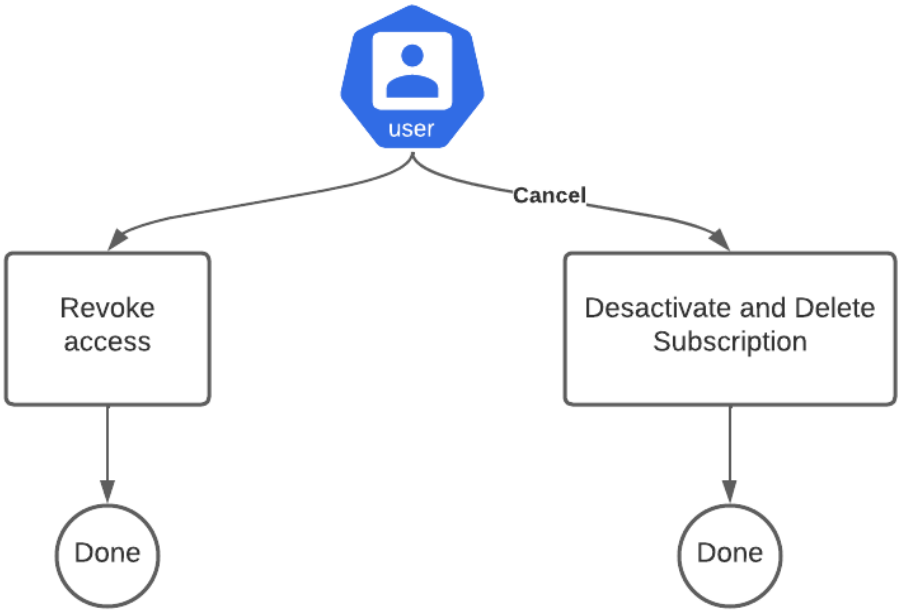

Figure 9 : Cancel Subscription

4.3 Process & Renew Subscription

The **process** action allows the user to re-subscribe to an application when his payment has failed several times and his subscription has been deactivated or when he has unsubscribed himself by revoking the access to his token through the protocol.

Also the **renew** action just allows to renew his subscription when it has expired. The user does again the actions he did when creating his subscription (approve and pay for the first time) but this action will not create a new subscription but only renew the existing one.

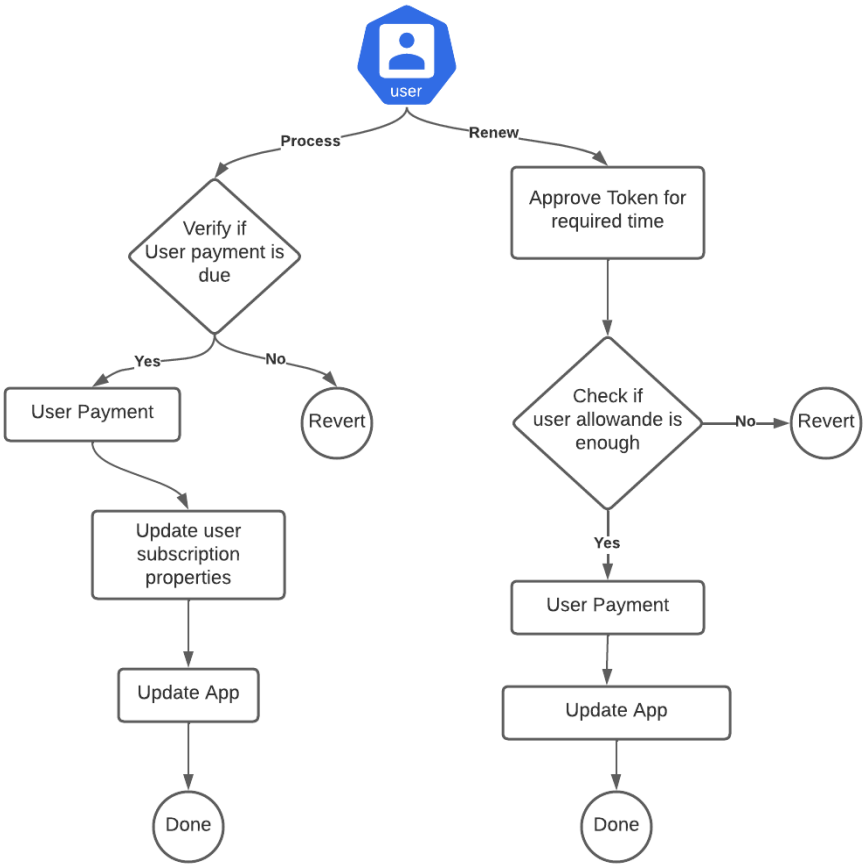

Figure 10 : Process, Renew Subscription

4.4 Migrate Payment

This action just allows the user to migrate his payment to another payment of the same application. The user will be able to change the type of payment whenever he wants.

4.5 Payment Due & Is My User

These two actions can be called by anyone and allow to see the status of the subscription of a given user. The method **paymentDue** allows to see if the subscription is due or over and the method **isMySubscriber** allows to check if the subscription of the user is active and to return the information related to his subscription. It is essential for the creator to know quickly if a user is a subscriber of his application or not and have the properties related to this user.

4.6 Automate Subscription

This method is called by regulators and allows them to either automate users' payments or disable their subscription if their payment has failed and subsequently earn rewards in **token subs** based on the number of tokens they hold in their wallet [5.3]

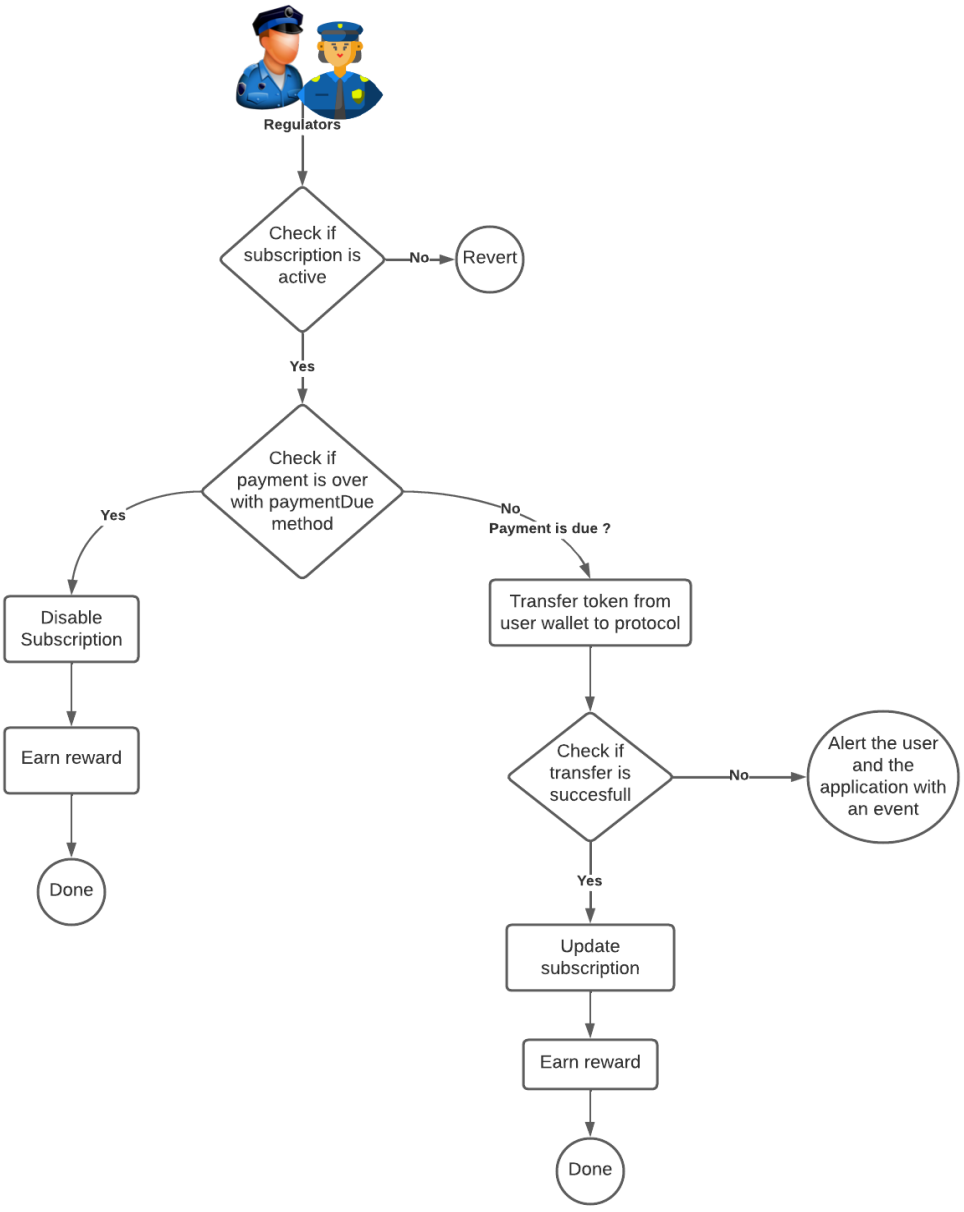

Figure 11 : Automate Subscription

5 Subs Economics

The subs protocol will be present on all EVM and Tezos chains to be used by any decentralized application that wants to implement a subscription or recurring on-chain payment system. The goal is to reach as many applications as possible, regardless of the blockchain used.

5.1 Subs Token

Initially, the first issue of the Subs utility token will be in the form of a classic ERC-20 token and will be deployed on all the chains mentioned above.

Subs token has 3 main uses cases:

- It serves to pay the regulators for any successful debit or cancelation [5.3]
- It also serves as a more advantageous payment method and give better return for the creator [5.2]
- It serves as governance for the protocol [5.2]

Information about the tokenomics of the protocol will be revealed in a second version of this document.

5.2 Subs Rules and Governance

The subs protocol has several rules implemented natively but which can be changed thanks to its governance mode. These rules can be summarized as follows :

★ Each application that accepts the subs token as a means of payment in its payment system, pays only 1% of fees to the protocol for each user payment.

★ Conversely, an application that does not accept the token subs as a means of payment pays a 2% fee to the protocol for each user payment

★ An actor must have at least 1000 subs tokens to become a regulator and automate transactions

★ Each reward must be greater than the transaction fee used by the regulator

★ If a user has not paid (does not have enough funds on his wallet, has revoked the app contract or any other reason) his subscription at the end of his period, he has the **k (small bonus time)** to pay, where **k** is a constant defined by this period

length. Ex: 1 month period, k is equal to 1 day and 1 day period k is equal to 12 hours.

The rights to the protocol are managed by the Subs token. The Subs protocol will begin with a decentralized governance system based on the Aragon framework, which will eventually become a completely self-governing protocol. The Protocol's Governance voting is weighted by SUBS token for decisions related to protocol parameters and upgrades of the smart contract. It can be compared to AAVE or UNISWAP governance where stakeholders vote on current and future parameters of the protocol.

5.3 Regulate to Earn

The regulate to earn can be just defined as the action of automating transactions in the subs protocol and earning rewards.

The subs protocol is automated thanks to these regulators which are external actors to the protocol. Anybody can be a regulator by having at least 1000 SUBS tokens in his wallet. The protocol subs have a simple interface allowing a beginner to call the automation method to earn rewards. But the more the protocol evolves, the harder it will be for a beginner to just click a button to automate a transaction in the protocol. The regulation of the protocol will also tend to more technical players with more advanced indexing applications.

The more tokens the regulator holds, the higher its reward. The reward is calculated to be always higher than the transaction fees, so that it is profitable for the regulator. Let's consider **R** as the reward and **F** as the transaction costs during the automation, so we have $\forall F, R > F$.

6 Conclusion

Subs protocol is based on an on-chain recurring payment model to allow anyone in the world to create a simple subscription or recurring payment system with their own currency without third parties. Users subscribe to these systems and give permission to the protocol to take the amount defined by the application through the regulators without locking their funds into a contract.

Subs is revolutionizing recurring payments on blockchain by bringing two key innovations to this ecosystem

- **Periodic payments without blocking funds**
- **Automatic payment for self-custodial wallet**

After the launch of the main network, the Subs protocol will continue to prioritize decentralization by implementing new features. Decision making will be conducted on-chain through the use of the Subs token for updating smart contracts.

References

Recurring blockchain subscription model issue. https://github.com/ethereum/EIPs/issues/948
SuperFluid . https://www.superfluid.finance/
DeSu. https://desu.tv/
Unlock . https://unlock-protocol.com/
Coinbase-Commerce https://www.coinbase.com/commerce
ERC-948.https://consensys.net/blog/blockchain-explained/subscription-services-on-the-blockchain-erc-948/
Current implementation of ERC-948. https://github.com/johngriffin/ERC948