Project INplusONE

Easy and privacy preserving time banking based on smart contract

BETH 2019, KISS challenge

ETH Zürich, 14.02.2019

Team INplusONE members:

Cedric Weibel (project leader)

Amr Tarek

Haoqian Zhang

Ali Tarek

Abdelaziz Elemary

Han Luo

Mostafa Garana

The software code to this project is open source and available on the URLs listed under *Source code and installation Instructions*.

Inhaltsverzeichnis

Outline of the paper and introduction to bitcoin

KISS, our partner company, proposed a challenge to renew their current work exchange system with a blockchain based trading system to manage and exchange time vouchers. Roughly said, the challenge was to design a system to record, send and receive tokens in a decentralized way. We approached the challenge in a team of seven students. In this paper I am going to talk in depth about our approach to this challenge, how we used blockchain technology, what we presented as solution and how we could further improve our solution.

The implementation of the whole system is rather basic and is meant to be an example on how to realise such a system. Moreover, the focus of this paper is to outline the potential of this technology in relation to this task. In recent years the blockchain technology became famous in relation to the hype of bitcoins, but the technology has way more potential. With so called smart contracts decentralised transactions can be executed. To state in this context more precisely, tokens received for physical work can be exchanged without having a third-party server controlling the whole process. To come straight to the point, we did not implement a fully blockchain powered software. We rather used a mix between blockchain technology and a server hosted by KISS for performance reasons. More to this is explained under the section: *Our approach to solve the task*.

Introduction: what is the problem to solve?

We are a multinational team of seven students. Two from China, Haoqian Zhang and Han Luo and four from Egypt, Amr Tarek, Ali Tarek, Abdelaziz Elemary and Mostafa Garana and last but no least Cedric Weibel from Switzerland. All students of computer science, except Cedric studying mechanical engineering. Obviously, such a diverse team with different cultural backgrounds can benefit from a lot of creative ideas of construction.

The main goal of our partner company KISS is to become the "4th money-free pillar" of the Swiss old age provision system. KISS helps to cut cost and strengthens the community spirit as people can live at home longer and suffer less from loneliness. KISS member, no matter old or young, provide each other with different sorts of assistance (but no medical care) and get time vouchers for their effort. Examples for such assistance are in the household,support in transportation or simply socializing.

Currently all the time banking is done manually and through the company KISS. This results in high cost and slow processing. On the long term, KISS funding is not ensured. Additionally, the whole system is rather new and the current political system is not familiar with time banking. Nevertheless there is already a similar system productive in another state of Switzerland[1]. Spitex, a non-profit organisation collaborates with the University of St.Gallen and provides a time-banking system for its students. The paper mentions another similar and productive system in Japan called "Fureai-Kippu-System". But both systems mentioned do not make use of the blockchain technology[2].

The requirements given to us by KISS are as follows:

- Exchange of time vouchers between two parties of which one is receiver of time credits and the other one is giver of time credits.
- Low transaction costs
- Easy to use and self-explaining system on any device for elder and very old aged folks
- Ensure privacy of each member
- Flexible blockchain platform
- Largely scalable

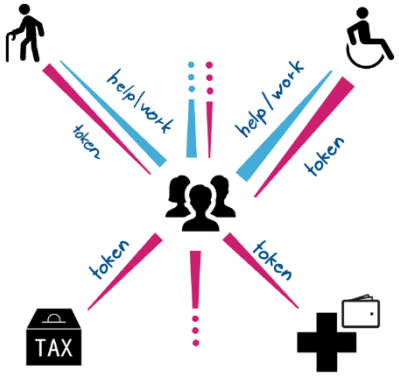

Figure 1: Basic flowchart on how the system should work.

To further evaluate the points mentioned: The two parties which exchange work with time-tokens are usually an elderly and a younger person. Low transaction cost is meant to be in relation to the processing of a smart contract. Transaction fees need to be paid to the miners and to the bitcoin network for confirming the transaction[3]. Elderly folks are often slightly handicapped in motor activities and sight. Additionally, many are not tech-savvy like the younger generation. Thus, the app needs to be easy to use and self-explaining. Flexible platform is an important requirement to the system. The whole blockchain technology is rather new and is evolving rapidly. In addition to that, the implementation will not be perfect at first instance. Many updates and features must be added in future. Last but not least, if the whole system will be implemented in Switzerland, the whole architecture needs to be largely scalable for several millions of users.

[1]4. Säule, http://www.zeitvorsorge.ch/kcfinderimg/files/doc/Die%204%20S%C3%A4ule.pdf, 02.03.2019
[2]Pflegewährungen in Japan, https://monneta.org/fureai-kippu/, 18.03.2019
[3] What are the transaction fees?, https://bitcoinfees.info/, 18.03.2019

Our approach to solve the task

Our approach was to use the blockchain technology to connect the members who are receiving and the one providing help. The privacy of the members is so ensured. Blockchain should guarantee full anonymity between two members exchanging work with tokens and no third-party server in between. The transaction cost between two parties will be handled by KISS, thus they will need an income to fund the transactions. One way to ensure the funding is to sell collected data (e.g. statistical data regarding top services needed by elder people) to third-parties. Needless to say that all collected data is to be handled with highest data protection.

The front end of the architecture, the web-app or mobile app, is easy to use. Big buttons and a simplistic design lead to an intuitive user experience. We decided, for demonstration purpose for a web-app. The KISS service catalogue is divided into subcategories like Household, Plants, Company/Entertainment, Cooking/Eating, Animals, Office and Family. On the one hand every service can be chosen and requested by the receiver. On the other hand, the giver choses what services he can provide. We used Ethereum as our blockchain platform because the developers have not enough experience with EOS, nor with Finance 4.0[4].

In our approach, the blockchain is only used for token exchange. Matching is too performance heavy, so KISS needs to do the matching locally on their server. We understand the process of matching in the following matter: it is to bring two persons (one searching for assistance and the other providing assistance) together who live as close as possible to each other. At first sight this violates our policy of full anonymity, but it ensures fast matching process.

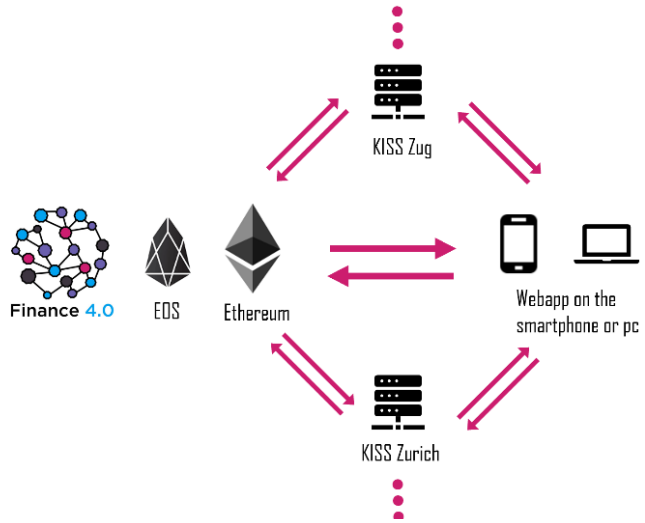

Figure 2: First concept of the architecture presented by us on our speech

4

Conceptual model

The Architecture

First of all, our architecture is based on three components: the frontend, the KISS servers and an Ethereum virtual machine.

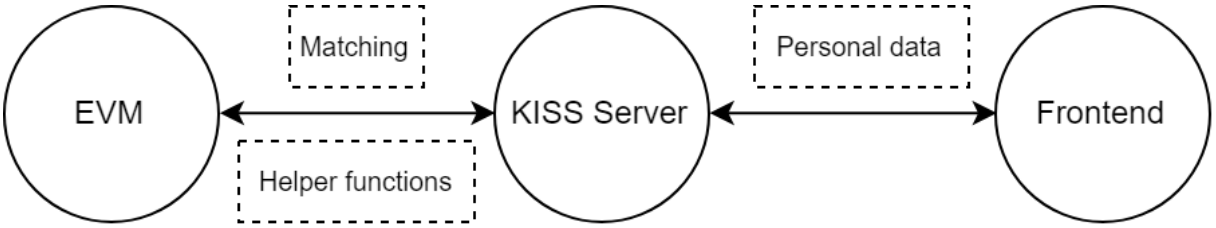

Figure 3 Interactions between the different parts of our architecture

1. Ethereum Virtual Machine (EVM):

   In this part we constructed the KISS token currency that will be rewarded to the user as an incentive to help. This was implemented using Solidity language, solidity is the programming language needed to develop on Ethereum network[5]. Our software acquires the user's information from the frontend and then execute one of the functions in the smart contract. If the user was a helper who donated his free time and offered to help for a specific number of hours, then the hours of help will be added to the helpers credit as tokens. Also, if a user wants to send tokens to another user, the send function will be executed from the smart contract. The EVM is not suitable for real life application as it is designed to be tested offline and only meant as a prototype.

2. KISS Server:

   Where all the user's sensitive data are kept, in addition to the pairing between the helper and the one who will offer help. This pairing is done by holding the requested needs of the user. Similar, when there is a helper logging into the system, the helper will enter how many hours and when exactly he will be free to provide the, from him chosen services. That information will then be directed to the KISS server matching system. If there is a match both users will be informed. Parameters like range between two users can be set manually into the software. The greater the network will be the more coverage there will be.
   Kiss servers will also hold the private information for each user of this application. Whether the one who offers help or need it, each user will have to enter some private information about himself as the telephone number, email, address, etc. to be able to distinguish between identities on the data bank.
   All this information till now are kept centralized in the KISS server to guarantee the total security for the user's private information.

[5] Solidity, https://solidity.readthedocs.io/en/v0.5.6/, 28.04.19

3. Frontend:

This part of the architecture is the most important part from the users perspective. In this part we focused on how to represent it as simple as possible, as this application will target mainly old people, so we made a very easy to use interface. After logging in with his personal login data, the user can manage his “online profile”. So to say, change personal data on the KISS server and choose the categories he needs help or he can provide. To mention it particularly again, all data is stored on the KISS server. So far not on the blockchain and especially not local on the users device. After entering all required data, the matching will be processed on the KISS server.

Programming Language / Frameworks:

For the programming language in the backend we used Solidity to develop on Ethereum network, and for the frontend part we developed using JS on React Framework. Furthermore, there is a part in the backend developed in Python (flask) that handles the KISS Database. The flow of the application starts from the frontend page where user can select to transfer money, provide help or the type of the service he wants.

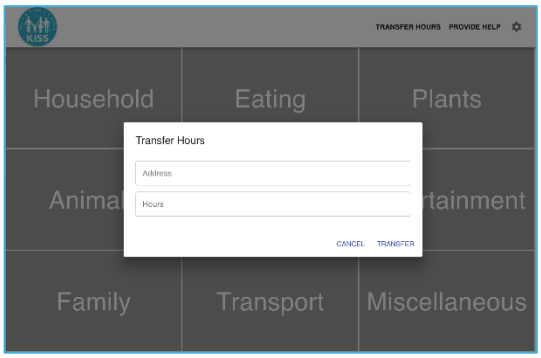

Figure 4 The basic design of the frontend with all featured categories. The right picture shows the pop-up windows to enter personal data or transfer hours.

First, if he chooses the type of service this action will be redirected to the backend where he can match with someone free to accept his request.

Second, if the user needs to transfer some hours to anyone, this will be redirected to the blockchain part where it handles the exchange of tokens between accounts with the “Send()” function built into the smart contract that takes the sender account and the receiver account as two parameters.

Finally, if the user selects “provide help” his information will be added to the backend database as someone who offers help in the hours he had been written in the form and then when another one needs help in the same time he will be matched with him on the KISS server.

Implementation of the blockchain technology

The Blockchain technology was chosen to be implemented in our project as it fits our requirements, to make a decentralized network between users that provides them with tokens to incentivize users to use our product. As a next benefit, blockchain makes it easy for users to exchange tokens between each other, or to exchange them as an example with currency value used for healthcare from the government. Even more, these tokens can be used in future to get help back from users. The young generation have so a possibility to boost their old age provision. Basically, the whole point of today's modern retirement annuity.

There are many blockchain networks, but we chose Ethereum network as it was one of the first pioneers in the Blockchain world. In addition, Ethereum has a very strong community that helps in any future bugs in development, with many existing and fully developed Dapps (decentralized applications) and live applications on the network. Another reason and the most important one for going with Ethereum is that it has a clear roadmap that can be understood and trusted. In ten to twenty years Ethereum will most likely still be around.

Many difficulties faced us at first when developing with Ethereum. One was testing the smart contracts. This was solved using Truffle framework and Ganache with Metamask[6] extension that makes it easy to compile, migrate and test the smart contract. Another difficulty we faced is the connection between the smart contract and react, but again we eventually solved this problem with the help of the community of Ethereum and the truffle framework.

There are many alternatives for Ethereum or for the blockchain in general to be used for this application. Decentralized ledger technologies (DLT) as Holochain, Cardano and P-Chain. But the problem with those are that right now none of them are stable enough to handle the project on the network. But as a future plan once the stability issue is solved one can substitute Ethereum with one of the many DLTs. As they provide more scalable, real-time aspect and better storage handling than Ethereum.

[6] Metamask, https://metamask.io/, 23.04.19

Conclusion, outlook and evaluation

As a conclusion, the whole concept of KISS and its digitalization has a vast potential. A similar system in St. Gallen already works fine. So, it could work as well in Zurich or in other states of Switzerland. The prototype software proved to be capable and extendable. There are many tasks open, as already mentioned previously, but nevertheless it is possible with a developer team to extend it to a robust and reliable software. The next step after a finished prototype of the software one could test it on a small scale. As an example, at a university or school.

Moreover, one of the major ideas in our future improvement is to have a more scalable and reliable application. Our current prototype application requires mainly Metamask, an extension for one's internet browser. Obviously, this is not the best way to do it as it will require from the user to use a web interface to control the Ethereum account, which makes it not feasible to be used from a mobile application (IOS / Android). As a next feature one could substitute the Metamask extension in the browser with another option to make the project applicable on all devices. As an example, we can use Trust Wallet[7] (Ethereum Wallet) which is supported on many more platforms.

The whole project gave us developers valuable experience in working with blockchain technology. Unfortunately, we could not implement code with the Finance4.0 platform, due to lack of time and experience. Nonetheless we think the platform has great potential and will be used more often in future time.

Source code and installation Instructions

In order to run the backend and the frontend of our application, Python3, pip3, Nodejs,React, Metamask, Truffle and Ganache are required. The source code and further installation instructions can be found on the following Github website:

https://github.com/betherworld/INplusONE

All illustrations are done by us with the help of https://www.draw.io/. Some of them can be found in our Powerpoint presentation. For any questions concerning the report please write to the project leader email address: ceweibel@student.ethz.ch

[7] Trust Wallet, https://trustwallet.com/, 23.04.19