Topic

The aim of the thesis will be a construction of a decentralized loan system using Ethereum technology. The final product should be a proof of concept of the possibilities of blockchain and smart contracts technologies.

Problem

Loan business is a vital part of the modern financial world. Being one of the oldest financial mechanisms to exist it provides the consumers with required capital. Throughout the whole history it has always been an example of a centralized system. That is one central entity hoarding the money from different sources gets inquired about financing possibilities. Such entity is in power to decide upon who shall receive the funding and with what interest. It is also the burden of the entity to deal with any cases of loan repayment disobedience.

The are a few problems with this concept all of which stem from the design itself. Firstly, if one wants to capitalize on his spare money through this type of scheme he has to create an entire new company with all the legal requirements and an administrative burden. Secondly, all the processes such as storage of money, allocation of loans, interest serving, acceptance of clients etc. have to be taken care of by the workers of the loan company. Thus, costs are generated and sources of human errors and introduced. Lastly, due to the way financial services are diversified around the world it is often very troublesome or even impossible to service loans internationally.

Solution

All of the aforementioned disadvantages can be circumnavigated using the fruits of modern computer science. That is blockchain and tightly related smart contracts. A Decentralized Autonomous Organization(DAO) is introduced which serves as a medium between those who have the capital and those who need the capital. The DAO which acts as a loan system allows anyone with any amount of Ethereum to bid a loan with an arbitrary duration and interest on a public exchange. At the same time a counterparty publishes an ask offer stating how big of a loan is required under specific duration and interest. The whole process closely resembles stock exchange. In result market forces lead to an equilibrium allowing both parties to reach their goal. The exchange itself is based on blockchain and no central server is required.

This design solves all of the issues pointed out so far. Anyone in the whole world with a connection to an internet is able to put his money to good use with few mouse clicks while maintaining anonymity. All of the operational activities are also immediately eradicated by market forces backed by smart contracts. Of course the problem of loan repayment still remains but the possible solutions of this specific mischief will be explained in the section describing a loan taker.

Loans

One loan offer on an exchange will consist of following parameters:

- **Basis**: Size of the loan denominated in Ethereum.

- **Duration**: The time after which the whole loan should be paid back together with interest. Depicted per predefined amount of days.

- **Interest**: Percentage of the loan basis which has to be additionally paid by loan taker. Calculated per predefined amount of days.

- **Collateral**: Information whether the loan is backed by a third party and has a low probability of going default. For example loan taker with no evidence to back his repayment probability will have only access to loans with collateral parameter being equal to None. At the same time loan taker with his loan being backed by a special bank agreement whereas there is a deposit with amount equal to the loan basis and interest incurred which is connected to a smart contract will have access to loans with collateral being equal to Yes.

The process of buying and selling loans will have the same characteristics as the one seen on the stock exchange.

That is loan ask whose size exceeds size of one respective loan bid will automatically cover the second identical loan bid. For example loan taker A asks for a loan of 1 ETH. There are two identical offers by loan provider B and C both equal to 0.5 ETH. The matching will automatically buy for A both loans from B and C.

Moreover, if loan taker asks for a loan of interest higher than the lowest present on the market he will be sold the loan with the lowest interest. For example loan taker asks for a loan of interest 2%. There are loans on the market being bid with both interest of 2% and 1%. The loan of 1% will be sold to the loan taker.

In terms of repayments the loan will function as an amortized loans. That is, the repayments will be constant with principal amount being paid back increasing and the interest amount decreasing. The formula for computing the installments looks as follows:

$$C = \frac{rP}{1 - \frac{1}{(1+r)^n}}$$

Where C is monthly installment, r is interest rate per month, P is principal amount/basis and n is a duration of a loan in periods equal to predefined amount of days.

Types of users

The project provides for three types of possible actors. They are as follows:

- **Loan providers**: Any user with spare Ethereum and access to the application. Such person bids his loan offer and once the loan is sold automatic processes backed by smart contracts care for repayments.

- **Loan takers**: User who has fulfilled required information inquires and can be assured of that at least one of the methods of repayment can be exercised on him. The methods are:

1. Regular individual refilling of his repayment account.
2. Usage of a collateral deposit from a third party.
3. Following legal procedures by bailiff of a country where the loan taker is located.

- **Authorities**: Third parties which for example serve the role of a collateral deposit or bailiff chasing the loan taker. This user has the possibility to refill the account of the loan takers repayment account.

Processes - WORK IN PROGRESS
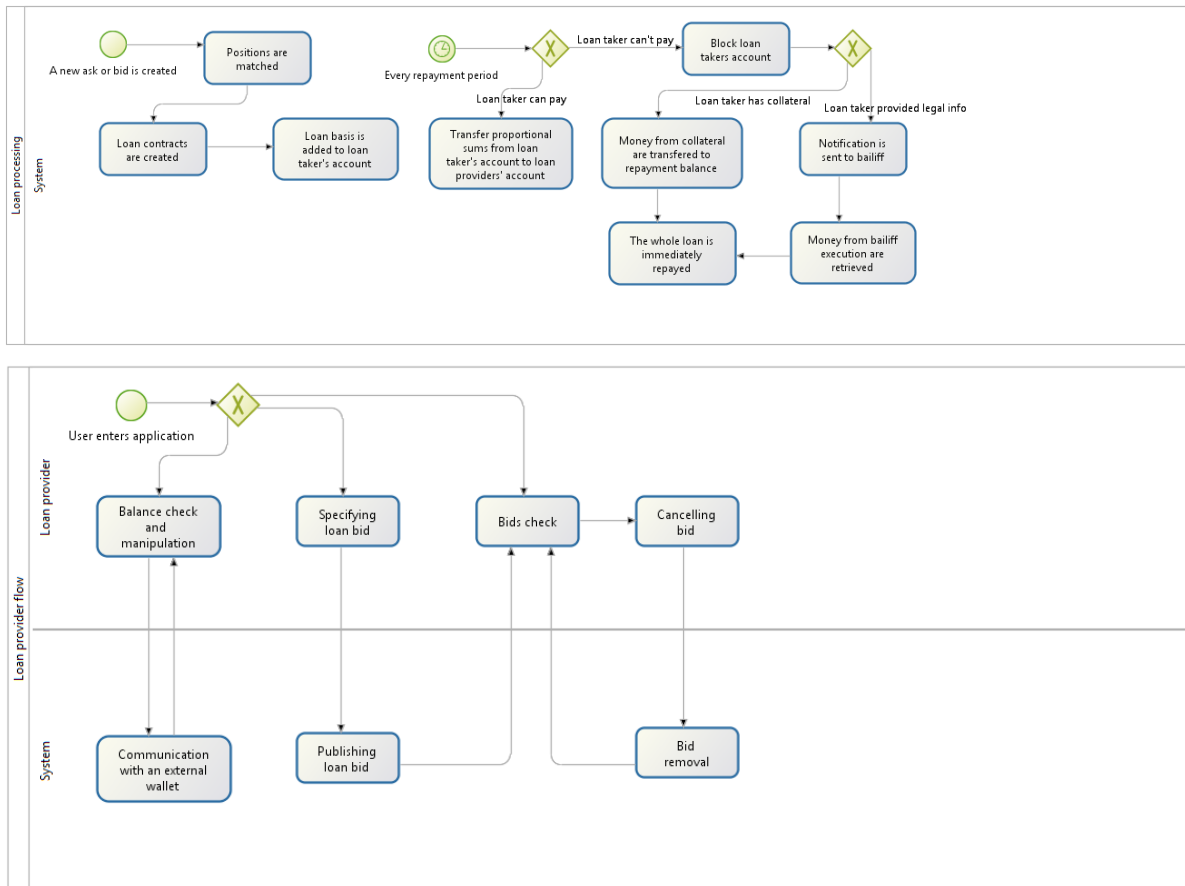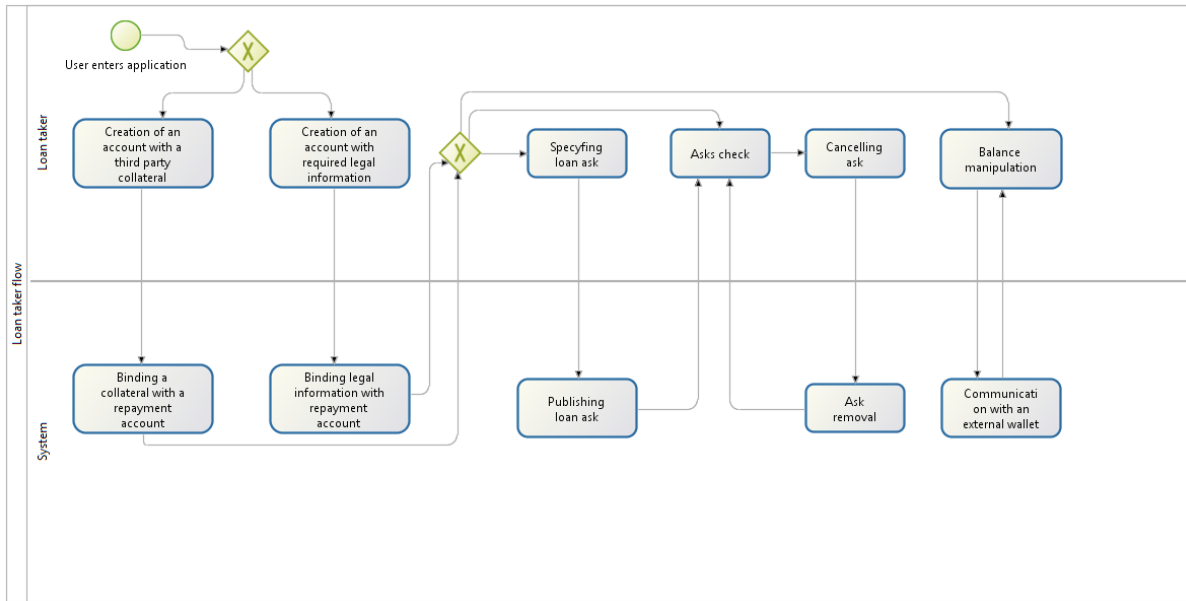
Requirements

The complete system should provide following functions:

1. Loan providers and loan takers should be able to:

   - check and transfer funds to and from their accounts using external wallet based on Ethereum network.
   - specify the following parameters in their loan bids and asks: Basis, Duration, Interest, whether it has collateral.
   - check their open bids and asks as well as the whole public exchange.
   - cancel all of their open bids and asks.

2. Loan takers should be able to:

   - specify legal informations which could be used by bailiff if repayment did not happen. At the same time the information should remain confidential so long there is no need to use them.
   - bind his account with an account of a third party. The additional account will always hold a sum required to repay the whole loan when needed.
   - repay the loan earlier.

3. Every user should be able to:

   - access the application having only internet connection.
   - maintain his anonymity while using the system.

Architecture
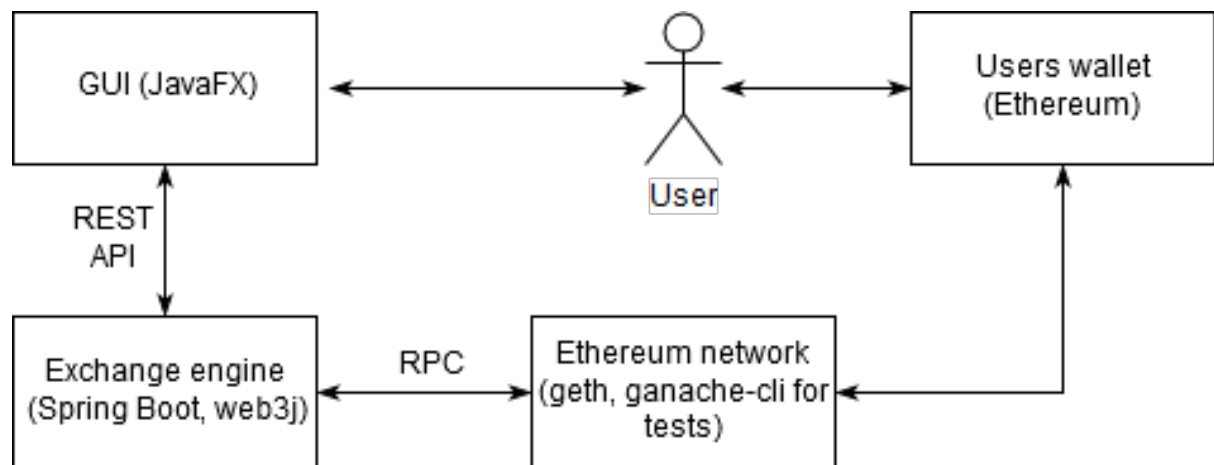
The diagram presents parts of the application and the respective technologies used to implement them. The whole projects is based on Java and its libraries binding it with Ethereum network. Below are presented types of smart contracts and how they interact with each other.
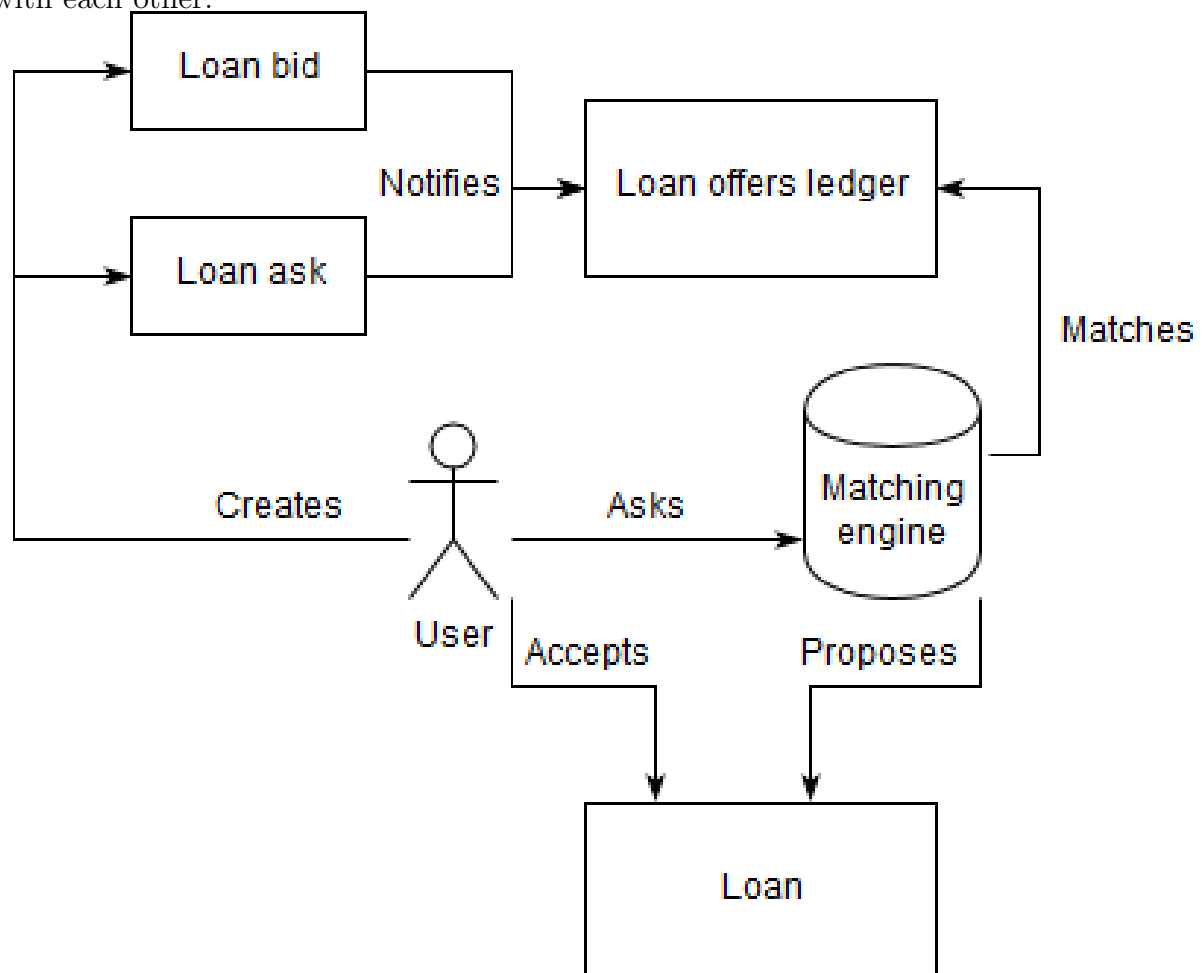
Ethereum network

Geth

WORK IN PROGRESS ()

Ganache-cli

WORK IN PROGRESS ()

Wallets

WORK IN PROGRESS ()

Solidity

Math

WORK IN PROGRESS (No decimals, Safe math)

Safety

WORK IN PROGRESS (Transfer funds, condition-computation-action, public operator)

Tests

WORK IN PROGRESS (js tests, solidity tests)

Java

web3j

WORK IN PROGRESS ()

Spring Boot

WORK IN PROGRESS ()

Visual design

Wallet balance   1 ETH

Deposit address   dbjsbqglwbgp23h523h9f2n23oinv23

Withdrawal

Withdrawal address   dbjsbqglwbgp23h523h9f2n23oinv23

Amount (ETH)   0,0

Network fee (ETH)   0,0

[Withdraw]

Time	Type	Amount	Fee	Address	Status	Wallet Balance
Dec 26, 2017, 1:00:00 PM	Deposit	0,1001 ETH	0,0 ETH	ETH	Pending	1 ETH
Dec 25, 2017, 1:01:00 PM	Withdrawal	-0,3045 ETH	0,0001 ETH	7465ovgvt324hf58374y549t	Completed	0,8999 ETH
Dec 24, 2012, 1:00:00 PM	Deposit	0,5009986 ETH	0,0 ETH	ETH	Completed	1,2044 ETH

Open offers

2 months ▼

1 month
2 months ✓
3 months
4 months
5 months
6 months
7 months

Collateral ☑

Interest	Amount	Total
5%	1 ETH	6 ETH
4.9%	2 ETH	5 ETH
4.8%	3 ETH	3 ETH

4.75%

Interest	Amount	Total
4.7%	1 ETH	1 ETH
4.6%	2 ETH	3 ETH
4.5%	3 ETH	6 ETH

EUR/USD

1.45000
1.40000
1.35000
1.33476
1.30000

20   27   Jul

Recent trades

Interest	Amount	Time
4.75%	1 ETH	18:01:01
4.9%	2 ETH	16:05:21
4.8%	3 ETH	15:01:01

Open position   [Bid] [Ask]

Basis
Duration
Interest

My positions

Duration	Collateral	Interest	Amount	Type
1 month	None	10%	1 ETH	Bid
2 months	Deposit	4%	5 ETH	Ask
2 months	None	14%	0,5 ETH	Bid

Outflows

Amount to be paid   10 ETH

Next repayment outflow   1 ETH

Inflows

Amount to be repaid   5 ETH

Next repayment inflow   0,5 ETH

End date	Type	Amount	Interest	Repaid amount	Next payment date	Next payment amount	Collateral
Dec 26, 2017, 1:00:00 PM	Inflow	5,01 ETH	1%	4 ETH(80%)	Dec 24, 2017, 1:00:00 PM	1 ETH	None
Dec 25, 2017, 1:01:00 PM	Outflow	0,54 ETH	2%	0,44 ETH(90%)	Dec 24, 2017, 1:00:00 PM	0,1 ETH	Deposit
Dec 24, 2012, 1:00:00 PM	inflow	0,1009986 ETH	3%	0,1 ETH(10%)	Dec 24, 2017, 1:00:00 PM	0,01 ETH	Legal

7

Implementation steps

Repository: https://github.com/Kadek/Bachelor

1. Deployment of a test Ethereum network.- DONE

   (a) Setting up network using geth. - DONE

   (b) Integration with electrum wallet. - DONE

   (c) Setting up network using ganache-cli - DONE

2. Construction of smart contracts backing loans. - DONE

   (a) Developing smart contract - DONE

   (b) Unit tests for smart contract - DONE

3. Construction of a matching engine for exchange. - WORK IN PROGRESS

   (a) Integration with a test network - DONE

   (b) Creating end-to-end flows for: - WORK IN PROGRESS

      i. Offering loans. - WORK IN PROGRESS

      ii. Matching loans.

      iii. Taking loans.

4. Creation of users.

5. Integration of collateral possibilities.

6. Construction of a graphical user interface.