The National Research University Higher School of Economics
Moscow
DSBA Department

REPORT

Subject «Virtual Settlement»
Topic «...»

Group	191
Student	Xu Jingtao
Supervisor(workshop)	...
Submission Date	2020

Moscow

Contents

1. Introduction 3
2. The objectives of Corporate settlements mechanism 3
3. Requirements of the mechanism 4
4. Overview and Comparative analysis of sources 5
5. Project plan 10
6. Terminologies and Definitions 13
Links of Reference materials 14

1. Introduction

Over the centuries, wave after wave of new payment technologies has emerged to meet societal demands. Coins, banknotes, cheques and credit cards were each innovation in their own day (Giannini (2011)). In recent years, there has been much buzz in the global market about Blockchain (Distributed Ledger Technology). Since this technology field represents a significant opportunity for every ecosystem to establish leadership in areas of cross-border payments, trade finance and so on, a group of leading banks decided to step into the picture and contribute to the development of D-apps and DLT.

Blockchains can be generally classified as: public chains, alliance chains, and private chains. These three classifications divide the blockchain technology into three types of applications, and the degree of openness of a decentralized application is actually very important to determine whether it is completely opened, semi-opened or completely closed, because the degree of openness is different, its implementation will also be different. In our project, we are going to consider the semi-opened networks, especially related to banking system.

The social distancing measures caused by the Covid-19 pandemic have dramatically increased the speed of the shift toward digital payments, and it is carrying security issues along with it. Hence, one fundamental task of blockchain-based settlement is dealing with the settlement risk. The implementation of such settlement has to set proper block sizes and computation times to ensure its ability to trace cash flow and make it easier to enforce financial regulations.

2. The objectives of Corporate settlements mechanism

2.1. Goals

This R&D project “Corporate settlements mechanism based on smart contracts” provides us an opportunity to understand the core principles of decentralized applications (D-Apps) and blockchain technology. This report is intended as a contribution to the improvement of the existing "abstract deals" concept on the R-Chain platform. Our goal is to develop and implement a mechanism for a group of companies based on R-Chain abstract business objects.

From an individual viewpoint, a feasible implementation of virtual settlements requires two type of instruments:

- First one method is to implement central accounting smart contract.
- Second way is to develop a mechanism which deals with dedicated node (Oracle).

Moreover, it will be necessary to determine advantages and disadvantages of both approaches, as well as the possible boundaries of their applicability.

2.2. Roles of participants

This is a two-member team project. Another team member is Ranga Baminiwatte (2nd year student of Data Science & Business Analytics) who does the analysing and testing part. I am dedicated in practical programming tasks and development. However, at the final stage of our project, once we have done the development of our settlement mechanism and smart contracts, we will use Raiffeisen server as the testbed to apply and check the functionalities of our work. Finally, we will come up with conclusions.

3. Requirements of the mechanism

There are certain functional and non-functional requirement of the mechanism that we are going to design and develop.

3.1. Functional

- To make settlement within the platform using virtual accounts.
- To calculate a future balance of the virtual account at the date of settlement, and to confirm or refuse the deal based on the calculation.
- To determine a need of credit funds and to form a loan request for the virtual credit funds. To execute a deal of a virtual lending. To repay a virtual loan.
- To determine a need of refill the virtual account with fiat currencies (traditional payments methods, for example bank account). To prepare and make a fiat payment.

3.2. Non-functional

- Performance: up to 100 deals by 1 hour, up to 500 status changes by 1 hour, up to 1000 executed payments by 1 hour
- Number of participants for 1 deal - up to 6
- Each participant should have an opportunity to work with multiple accounts

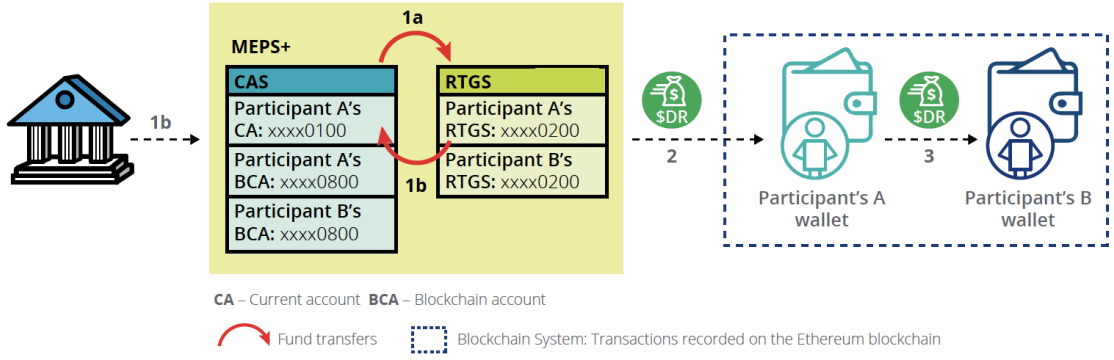


Figure 1. Sample structure[UBIN project]

The sample structure of virtual settlement in UBIN project illustrates the basic idea of a sufficient flexibility and functionality in the usage of Business applications, including a proactive monitoring system(In figure 1, Blockchain system).

4. Overview and Comparative analysis of sources

4.1. Background.

The success of Bitcoin is testimony to the fact that a peer-to-peer decentralised digital currency is feasible. Besides, Ethereum demonstrated nicely the need for a decentralized application development platform. If we say Bitcoin and Ethereum are "The Blockchain v1.0 and v2.0", then R-Chain can be definitely considered as "The Blockchain v3.0" – such thing will establish an ecological blockchain environment.

From an individual viewpoint, the current implementation of Raiffeisen bank's R-Chain platform provides end-to-end deals management. However, settlements between the participants of the deals are carried out within ordinary bank settlement services outside the R-Chain, while only payment documents are transmitted through the platform as a justification for performing the corresponding operations. Currently, there are several decentralized platforms which are developing to implement p2p payments between participants (both banks and their clients). However, in such platforms, payments are implemented separately from the deal's management. In addition, payment execution process is associated with a significant exchange of documents. To obtain a new service quality, it is desirable to combine in one platform both the management of the deals and the execution of payments, and also the possibility to manage efficiency of the participants' funds while ensuring their liquidity.

4.2. Progress so far.

By 6th of Fabray 2021, we have done several initial programming tasks:

- Set up a private blockchain network on our computers.
- Learning how to work with Truffle.
- Development of Account and Deal smart contracts

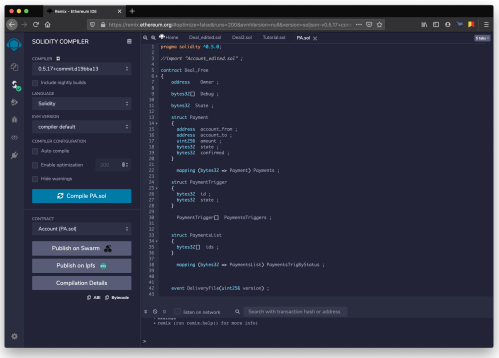
We chose Solidity to develop smart contracts, since it is hugely important for a smart contract developer to be proficient in solidity (Programming Language). Solidity is an advanced programming language which was shaped by few common languages, such as C++, JavaScript and Python. In addition, we use Remix-IDE as our main tool to write and compile our smart contracts. (https://remix.ethereum.org)

4.2.1 Remix-IDE

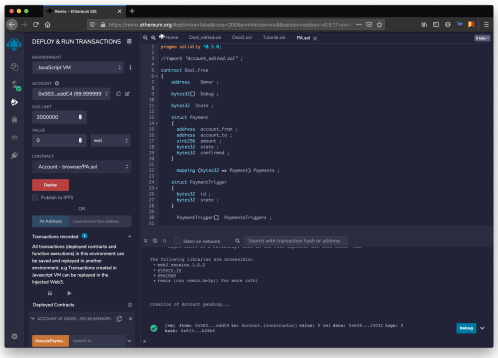
After setting up the private network and knowing the programming language Solidity, we moved on to the implementation of two essential tasks:

- Account – smart contract
- Deal -smart contract

In the beginning, our mentor provided us a basic structure of smart contracts (Deal and Account). We have written the corresponding smart contracts (They are attached in the resource code section).

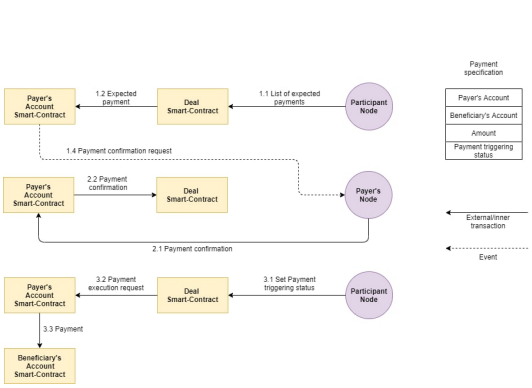


(a) pic1. Compiling smart contracts

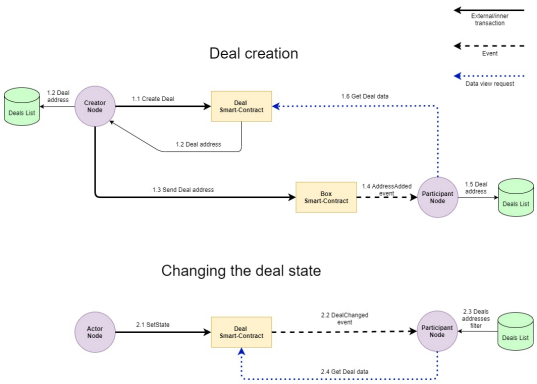


(b) pic2. Deploying smart contracts

Figure 2. Remix-IDE



(a) pic1. Basic structure of two mentioned smart contracts



(b) pic2. The workflow of Deal.sol

Figure 3. Mechanism of the virtual settlement(transaction)

4.2.2 Implementation

In the diagram `3a`, it is a complete procedure of deals' settlements. From arrow 1.1 to 1.2, deal smart contract requires a function which can efficiently store numerous payments and their details. After the expected payments have been sent to their payer's account smart contract, payer's node (in 1.4) can decided whether this payment is confirmed by using function `changeState()`. Finally, if payers confirm to set deals, the information will be sent back to Deal smart contract. In the Deal smart contract, the state of corresponding payments is changed to "Y" (means that payments are ready to be executed). One noteworthy feature is the payment triggering statue, which is stored in the mapping data structure (`mapping` from states to payment addresses), can avoid to check the state of each payment. The approach could greatly optimize the time complexity of payment execution. Diagram `3b` illustrate the internal actions of Deal creation and Changing the deal state which are implemented in these two functions.

4.3. Source Code

You can link to the entire source of this project thought this hyperlink:

https://github.com/JingtaoXu/VirtualSettlement

The codes below only contain the declaration of all key functions.

```
pragma solidity ^0.5.0;
contract Deal
{
  address    Owner ;

    bytes32[]  Debug ;

    bytes32  State ;

    struct Payment
    {
      address  account_from ;
      address  account_to ;
      uint256  amount ;
      bytes32  state ;
      bytes32  confirmed ;
    }

    mapping (bytes32 => Payment) Payments ;

    struct PaymentTrigger
    {
      bytes32  id ;
      bytes32  state ;
    }

    PaymentTrigger[]  PaymentsTriggers ;


    struct PaymentsList
    {
      bytes32[]  ids ;
    }

      mapping (bytes32 => PaymentsList) PaymentsTrigByStatus ;



    event DeliveryFile(uint256 version) ;

    event ChangedState(bytes32 state) ;
    event Paid(bytes32  id, address  account_from,
               address  account_to, uint256  amount) ;
```

```
    constructor() public
    {
      Owner=msg.sender ;
      State="NEW" ;
    }

// Key functions:
function PutPayment(bytes32  id_, address  account_from_, address  account_to_,
                      uint256  amount_, bytes32  execute_state_) internal
{ ...   }

function LoadPayments(bytes32[] memory Payments_)  public
  { ...   }

  function ConfirmPayment(bytes32 _id) public
  { ...   }

  function SetState(bytes32  state_) public
  { ...   }

}

contract Account
{
 address    Owner ;
     int256    Balance ;

      struct Payment
    {
      bytes32  id ;
      address  account_to ;
      uint256  amount ;
      bytes32  accepted ;
    }

    struct Deal
    {
      mapping (bytes32 => Payment) Payments ;
      bytes32                      Used ;
    }

    mapping (address => Deal) Deals ;

    //unique payment
    //mapping (bytes32 => mapping(bytes32 => Payment)) Paymentss ;

    event  AcceptPayment(bytes32  id, address  deal, address  account_from,
```

```
                        address  account_to, uint256  amount) ;

    constructor() public payable
    {
            Owner = msg.sender ;
            Balance = 10000;
    }

    // Key functions:

    function PutPayment(bytes32  id_, address  account_from_, address  account_to_,
                        uint256  amount_) public
    { ...   }

    function PaymentAccepted(address  deal_, bytes32  id_) public
    { ...   }

    function GetPayment(address deal_, bytes32  id_)  public view returns
                        (address, uint256, bytes32 retVal)
    { ...   }

    function Income(uint256 amount) public
    { ...   }

    function ExecutePayment( bytes32 id_ )  public returns (bool retval)
    { ...   }

}
```

5. Project plan

Пункт Task	Исполнитель Responsible	Ментор Mentor	План Planned date	Реально Actual date
Анализ методических материалов Analysis of methodological materials			14.12.2020	14.12.2020
Развертывание приватной сети Ethereum на личных компьютерах студентов. Deploying a private Ethereum network on students' personal computers		Liz Azadova	28.12.2020	28.12.2020
Изучение работы с приложением Truffle Learning how to work with the Truffle app		-	28.01.2020	28.12.2020
Разработка смарт-контрактов - перевод средств между счетами Development of smart-contracts: transferring virtual funds between accounts		Pavel Bolotov Nadezhda Radchenko	03.02.2021	01.02.2021
Разработка смарт-контрактов - интеграция механизма перевода средств со смарт-контрактом управления сделками Development of smart-contracts: integration of the funds transfer mechanism into a deal smart-contract		Pavel Bolotov Nadezhda Radchenko	17.02.2021	

Пункт Task	Исполнитель Responsible	Ментор Mentor	План Planned date	Реально Actual date
Проектирование API работы с платежами на базе БД (дополнение API работы со сделками) (привязывание платежей к сделке, подтверждение платежей, регистрация исполнения платежей) Design of payment API based on database: linking payments to a deal, confirming payments, registering facts of payments execution		Pavel Bolotov Nadezhda Radchenko	03.03.2021	
Разработка программного кода реализации API работы с платежами в Адаптере R-Chain (привязывание платежей к сделке, подтверждение платежей, регистрация исполнения платежей) Development of payment API based on database: linking payments to a deal, confirming payments, registering facts of payments execution		Pavel Bolotov	10.03.2021	

Пункт Task	Исполнитель Responsible	Ментор Mentor	План Planned date	Реально Actual date
Разработка смарт-контрактов - разработка различных вариантов механизма вычисления ожидаемого остатка Development of smart contract: mechanism of calculation the expected (future) balance (at least two options)		Pavel Bolotov Nadezhda Radchenko	24.03.2021	
Разработка смарт-контрактов - механизм предоставления и погашения кредитов Development of smart contract: mechanism of lending (loan issuance and repayment)		Pavel Bolotov Nadezhda Radchenko	24.03.2021	
Проектирование API фиатных платежей для обеспечения ликвидности виртуальных счетов Design of payment API: making fiat payments to provide a liquidity for virtual accounts		Pavel Bolotov Nadezhda Radchenko	07.04.2021	
Разработка программной реализации API фиатных платежей для обеспечения ликвидности виртуальных счетов Development of payment API: making fiat payments to provide a liquidity for virtual accounts		Pavel Bolotov	14.04.2021	

6. Terminologies and Definitions

[1] Blockchain – Blockchain technology is a decentralized, distributed virtual ledger that records the provenance of a digital asset.

[2] Blocks- A block is a permanent store of records which, once written, cannot be altered, or removed.

[3] Smart Contract - A smart contract is a computer program or a transaction protocol which is intended to automatically execute, control or document legally relevant events and actions according to the terms of a contract or an agreement.

[4] Mining – The process of securing and verifying bitcoin transactions.

[5] Genesis block- First block of a Blockchain

[6] Gas- The cost necessary to perform a transaction on the network.

[7] Bitcoin – The most popular digital cryptocurrency introduced in January 2009.

[8] Ether – The cryptocurrency used in Ethereum platforms.

[9] R-Chain - R-Chain is a fundamentally new blockchain platform rooted in a formal model of concurrent and decentralized computation. The R-Chain Cooperative is leveraging that model through correct-by-construction software development to produce a concurrent, compositional, and massively scalable blockchain.

[10] Geth - command-line interface (CLI) tool that communicates with the Ethereum network and acts as the link between your computer and the rest of the Ethereum nodes. Private

[11] D-Apps - A computer application that runs on a distributed computing system.

[12] Ethereum wallet - Applications that let you interact with your Ethereum account.

[13] Solidity - Solidity is an object-oriented programming language for writing smart contracts. It is used for implementing smart contracts on various blockchain platforms, most notably, Ethereum.

[14] Truffle - Truffle is a development environment, testing framework, and asset pipeline all rolled into one. It is based on Ethereum Blockchain and is designed to facilitate the smooth and seamless development of D-Apps.

[15] Proof-of-stake - Proof-of-stake is the most common alternative to proof-of-work. Proof-of-work wastes too many computational resources.

[16] Oraclize - Oracle is a service that aims to enable smart contracts to access data from other blockchains and the World Wide Web.

[17] Ethereum – An open-source, blockchain-based, decentralized software platform used for its own cryptocurrency, ether.

[18] Ethereum network - A Blockchain which is isolated from the Main Ethereum network. Ethereum Private Network is mainly created by organizations to restrict the read permissions of the Blockchain.

Links of Reference materials

[1] Nrayin Prusty Building Blockchain Projects — © 2017 Packt Publishing

[2] https://github.com/project-ubin

[3] Project Ubin: Central Bank Digital Money using Distributed Ledger Technology — https://www.mas.gov.sg/schemes-and-initiatives/project-ubin

[4] Remix documentation — https://remix-ide.readthedocs.io/en/latest/

[5] Solidity documentation — https://docs.soliditylang.org/en/v0.7.4/