Technical White Paper

(Some details might be added)

Decentralized financial system

CREDITS

Version 1.6/26 nov 2017

Contents

Abstract

CREDITS platform is a decentralized financial system for the direct interaction between participants on peer-to-peer (P2P) principles. The platform expands the potential of using financial services on the basis of a distributed ledger, self-executing smart contracts, and CREDITS cryptocurrency. The system is aimed to unite all participants on one site, providing them with a platform for creating and using financial services; where everyone can both offer a service and use it. Thanks to a well-defined and balanced technological system, the CREDITS platform offers a new technical solution and a new conceptual model of networking participants' interaction for the development of modern decentralized financial services.

Introduction

A fully peer-to-peer arrangement for service delivery systems which allows for forming financial services: money transfers, currency and value exchanges, crediting, funding and other services directly between participants. Everything is provided without additional intermediaries, according to a principle – one of the equal participants – to other system participants. As a result, everyone gets cheaper, faster and better services.

The world is moving towards the direct interaction between people on peer-to-peer principles – equal to equal. A revolution happened! This is clearly seen by the overturn in the mass media: until the 1990s, newspapers, magazines, and TV were the main providers of information. Today, opinion leaders are bloggers, found on Youtube channels and social networks, money is invested in crowdfunding and ICO, and information is stored in decentralized cloud systems.

The financial industry, perhaps, is one of the few industries that lags behind, that resists the introduction of decentralization and direct interaction between participants. Although, technically, it is much easier to create decentralized financial services than it is to create unmanned vehicles.

A corresponding technological environment is required to create a system of decentralized financial products and services based on distributed ledger:

1. High execution speed (in seconds), together with the capability to handle a huge number of transactions simultaneously (hundreds of thousands per second) at a low cost of each transaction (for micro payments and non-cash transactions).

2. Development of a system where all participants and items necessary for the qualitative financial decentralized services are combined: personalization of users, KYC, credit history bureau, settlement centers of fiat money, withdrawal and cashing of cryptocurrencies and so on

These are two big and basic tasks that currently hamper the development of peer-to-peer financial products.

We present you a solution to these tasks, we implemented with the help of the CREDITS financial system.

CREDITS single technological decentralized platform can combine all participants of financial services, safely and quickly execute all transactions using the principles of a distributed ledger. Self-executing smart contracts and the principles of a federative voting system provide unlimited opportunities for all participants to create unique interactions of various financial products. The platform opens a new huge market and a new potential for using blockchain projects and services in financial and other sectors that could not be used previously because of speed and transaction cost limitations.

1. Network ledger

Definitions

1. A system is a set of decentralized network nodes performing processing, saving transactions, executing and confirming the terms of smart contracts, processing requests from third-party systems, providing information data when requested.

2. A network node is a computer where a complete network client is installed, connected to a common system, verifying transactions and writing them to the ledger.

3. A ledger is the list of transactions confirmed by the system and stored on all network nodes.

4. A transaction is the system item, denoting a request to perform a smart contract method or any action on the network and recording the results in a blockchain system.

5. A smart contract is the system item, computer protocols that facilitate, verify or ensure compliance with the terms of interaction. They usually have a user interface and often emulate the logic of contractual relationships. The key property of a smart contract is its decentralization and its independence from a central source.

6. A smart contract method is the program code responsible for calculating the result of work of the smart contract terms and recording it to the ledger.

7. A contracting party is the final network participant and the system user.

Network Nodes

We use several types of nodes, depending on their purpose to build a decentralized network based on free access and node connection:

1. A common node (ОУ) is the node participating in transaction verification for validity but has a minimum trust factor. It is also a candidate for the role of a trusted node and the node of the current processing in the next cycle of node role selection in the network.

2. A trusted node (ДУ) is the node participating in transaction verification and has the maximum trust factor (1), is a candidate for the role of the node of the current processing and common node. This node cannot become trusted during a mathematically calculated number of selection and voting cycles among nodes. The mathematical calculation depends on the number of nodes and the network complexity.

3. The main node (ГУ) of the network is the node participating in verification and responsible for adding transactions to the transaction ledger block. This node cannot become trusted or the node of the current processing during a mathematically calculated number of voting cycles, which mathematical calculation depends on the number of nodes and the network complexity.

The system uses a trust factor – an absolute fractional numerical value from 0 to 1, expressed in mathematical terms of the number of trusted nodes +1 to the total number of nodes in the network. The maximum number of trusted nodes cannot exceed 50% of network nodes.

The Last Saved Block

The Common ledger of Blocks (CRB) is the synchronized state of the entire common ledger of blocks in all system nodes.

By the ledger block contents, we mean a unit of information stored that contains a hash code of the previous block and a list of data related to this ledger with the associated number of the previous block. Upon receipt of the block from another node, it takes its place in the common ledger of blocks according to the number. This saves network bandwidth.

During the synchronization, only the block number is checked first. If the block is missing at this node, it is downloaded and saved.

As a result, the system at any time contains the latest up-to-date copy of the ledger. We name it the last ledger (LR). It is automatically created by the node responsible for the ledger formation upon reaching a consensus. This block is sent to all the system nodes in order to maintain the up-to-date uniformity of the ledger state in all the system nodes.

Each node is associated with all other nodes in the network and constantly exchanges new blocks with transactions with them, so as to always maintain the relevant information. All blocks form a set of transactions-candidates waiting to be added to the ledger. At the same time, each server generates assumed sets of the candidates for other servers and the proposed set of transactions. A decision is made upon checking, whether to add them to the ledger.

As a result, it is possible to store the ledger data multiple times on multiple servers – the system nodes, and all information is protected. The more nodes in the system, the more reliable and independent it is.

Synchronization of nodes

Each new node is launched and synchronized after determination definition and thorough trust verification. To improve the information processing rate, all processes are handled simultaneously, independently of each other. If there are no incoming variables, then an empty ledger store is created – a space is reserved in RAM for further simplified access. In the case of the required ledger not being available, a request is sent to trusted nodes to receive all transactions made for the synchronized account.

If the input parameter is an object characterizing the transaction, then the search in all running synchronization threads is started. The operation results in a numeric code – the position number in the trusted node ledger for the current thread or the error number if the value is less than zero. If the thread method ends with a connection error, then the thread completely ends.

2. Network Consensus

The consensus in CREDITS is a method of group decision-making. With the aim to develop final solutions acceptable for all network nodes.

Consensus Comparison

The definition of the principles of the decentralized CREDITS ledger to compare different types of consensus:

- ledger availability (nodes can write data to the ledger and read them from it at any time);
- Modifiability by all participating network nodes;
- Consistency of all system nodes (all nodes see an absolutely identical version of the ledger, which is updated after changes);
- Resistance to separation (if one node becomes inoperable, this does not affect the operation of the entire ledger).

Compared parameter	Credits specific dPoS and BFT	PoW	PoS

The principle of identifying the node that generated the block.	Calculation of the mathematical function. Confirmation of storage of the last ledger copy and BFT	Performing an iterative calculation of the mathematical function, with varying complexity.	Search for the maximum stack among participants (competing nodes).
Attack 51%.	Unlikely, since it is necessary to have a complete ledger in resources and a computational power to calculate, and the trusted nodes are selected dynamically.	Probable, but will be very expensive in terms of the use of resources.	Probable, but expensive, because of the need to increase one's own stack.
Compensation for the work done at the site for adding to the ledger/blockchain.	Calculated automatically, depends on the commission per operation.	Provided fix for block mining.	Provided fix for block mining.

The Concept of the Main Network Node

All network nodes are decentralized and none of them have priority. It is required to define a network node that will process the queue of transactions stored at different network nodes. After that, it must enter a newly generated transaction block into the ledger.

The CREDITS platform uses its own combined protocol to increase the speed of transaction processing, to provide complete security of data storage, processing and transfer of transactions. The protocol is based on the calculation of the mathematical function of all ledger transactions, applying the Proof of Work principles. It accurately determines the storage of the latest up-to-date copy of the ledger and software at this node (Proof of Capacity), by calculating the checksum of the values of the entire contents – the hash code. The size of files is determined as well, as the proof that this is the latest, up-to-date copy and a hash code of the latest transaction recorded in the system.

To become the main network node, the node searches for the value of the hash function that it calculates based on the last stored ledger. We organize a healthy competitive environment between the network nodes for the opportunity to become the main node; to generate and store a new ledger.

After calculating the function and obtaining the result, it is sent to all network nodes for verification. The result contains a timestamp of the calculation and a value based on the calculation of the function of the ledger files and software. All nodes receive the calculated value, compare the calculation time allocated for the search for the main network server, verify it and confirm the trust factor of the node, and also confirm its opportunity to participate in the competition – to become the main network node.

After receiving approval from all network nodes (BFT principals), a list is formed of nodes that correctly calculated the value of the function and contains a timestamp. The node that received the correct result and had it approved in the fastest time, becomes the main network node of the moment.

The SHA-2 algorithm concept is used to calculate the hash sum of the file.

Hash functions of the SHA-2 family are built on the basis of the Merkle-Damgard structure.

The initial message after the addition is divided into blocks, each block is divided into 16 words. The algorithm passes each message block through a cycle with 64 or 80 iterations (rounds). At each iteration, 2 words are converted, and the rest of the words define the conversion function. The results of each block process are summarized. The sum is the hash function value. However, the internal state is

initialized based on the results of the previous block processing. Therefore, it is impossible to independently process blocks and sum up the results.

Equipment of Network Nodes

We are striving to build a platform with the fastest possible transaction processing characteristics, so we propose using a material incentive to maintain network nodes in the best condition: high-performance server equipment and high internet bandwidth.

As a material compensation, the owner of the main network node will receive remuneration in CREDITS currency from a number of commissions per transactions of this processed ledger. The rest (½) is intended for the trusted nodes, which take part in BFT consensus. The percentage can be changed, as well as separated to the rate formation system through federative voting by the network nodes, after the initial coin offering for at least three years.

As a result, we encourage server owners to keep this server on hardware of the highest performance and to maintain a high-quality, high-speed communication channel.

Consensus Building

As a result, we have the main network node selected by all nodes. The main tasks of the main node are: obtaining transactions in the candidate status to add to the ledger from all nodes, processing them, building the last relevant ledger and sending a newly built ledger to all network nodes. The process of transaction handling and building of the last relevant ledger is precisely the search for a consensus solution. The result of building the last relevant ledger is the consensus solution.

The whole process can be divided into the following stages:

1. Search for the main network node;
2. Building of trusted nodes;
3. Receiving the list of transactions and building a list of candidates for addition to the ledger;
4. Processing the list of candidates, voting of nodes (trusted and common nodes have different weight factors (trust factor);
5. Removal from the list of candidates of unconfirmed transactions that have not been verified or having a negative confirmation;
6. Building a list of confirmed transactions to be added to the ledger;
7. Adding transactions to the ledger with the timestamp and hash code of the block that contained the transaction;
8. Sending the block with transactions to all network nodes. When received, it is added to the registries of all nodes.

Building and Initiating the ledger

The whole process can be described in the following sequence:

1. The end user of the network in the system generates a transaction.
2. When all conditions of the smart contract specified therein are met, the user initiates the action (transaction) through calling the required method using the platform software.
3. To follow the fundamental principles of the blockchain, the kernel of validators keeps track of synchronization and invariance of the latest ledger version.
4. At the time of building the consensus, all transactions received during the cycle are collected in the block.
5. A number is assigned to the block, consisting of a timestamp and a node identifier converted into a hash code, and then the block is placed in the consensus module.

6. After compilation of the white list of candidates, not only the hash of the transaction is written to the ledger, but also the hash of the block, to always certify the source based on it.
7. This hash is a kind of signature of the block and the one who created this block with transactions.
8. After consensus building using a federative search algorithm, the transactions added to the block are passed to the validator's kernel to be written to the ledger.

Transactions not Included in the Register

Transactions not included in the list of ready transactions are marked as rejected. Information about this is immediately displayed at the sender (initiator) of the transaction.

Transactions not included in the ledger remain in the set of candidates and are stored in the network nodes. All the new transactions received by the server at the time of consensus also arrive there, and then the search process begins anew. Such a continuous cyclic operation of the network allows conducting transactions for a fairly short period of time while maintaining a high degree of reliability and relevance of information.

3. Transaction Processing

Transactions

A transaction is the minimum unit of the system informing the platform of the execution of contract methods or direct transfers between accounts without creating a smart contract, followed by placement of the result in the peer-to-peer network.

Consensus Building

The system uses a federative model to build a consensus – voting of trusted validator nodes, and also the consensus building algorithm – an algorithm for passage of a finite-state automaton. Consensus works by cycles (time steps), per time step, transactions are extracted and placed in a pool (one-dimensional array). After being placed in the pool, all transactions are sent to trusted nodes in order to receive a response. If the response is received, then the transaction for which the request to add was sent, can be added to the ledger of this validator. After that, it is sent to the next validator in the network. When consensus is built – at the end of the chain where the transfer legality is fully confirmed, the transaction is sent to validation with a mark for writing and saving to the ledger.

Transaction Processing

To achieve the decentralized nature of the system, each server must have both ledger storage and also be a fully-fledged handler of all transactions.

The system uses the concept of system kernels. By kernels, we mean a data handler that performs a specific production task, regardless of the availability and operability of the remaining system components. Each kernel, at the input, at the time the task is executed, receives a list of variables for processing. And always gets a result at the output – positive, any other or an error. As a result, the system kernel always contains the response code, in addition to the main data set. This structure is required for the highest possible speed of each process, which must work independently of each other.

Ledger Entry Structure

To achieve significant ledger performance, but at the same time, without compromising security, we propose to use a ledger database without building the Merkle tree from the hash code of the previous block and the transaction result.

Merkle tree (TTH – Tiger Tree Hashing) is a type of hash function used to check the integrity of data, to obtain a unique identifier of the chain, and to restore the sequence. The data is divided into small parts – blocks that are individually hashed using Leaf Tiger Hash, then the Internal Tiger Hash is calculated from each pair of hashes one-by-one. If the hash does not have a pair, then it is transferred to the new chain unchanged. Next, Internal Tiger Hash is calculated again in the chain for each pair. This procedure is repeated until there is one hash left.

When the ledger is operated using Merkle trees, the transaction processing speed is very low, and the load on computing resources is very high. In our opinion, this is not a rational use of data storage.

CREDITS ledger Structure

We offer to abandon Merkle trees and use the transaction ledger in the CREDITS system; with each entry consisting of a hash code of the transaction block to add to the list of candidates in addition to the ledger. Also, the entry has the node identifier and the timestamp when it was generated. The ledger entry contains the transaction direction, its initial and final accounts, the type of write-off, the number of write-off units, the type of depositing, and the number of depositing units. This principle increases the speed of transaction processing, increases the complexity of illegitimate ledger change and excludes possible changes in the ledger entry with hindsight.

Block Size

The unit of time is the cycle of searching for the main and trusted nodes, and the cycle time is calculated depending on the network complexity. Per unit of time, the network contains N transactions generated and transferred for processing to the network from the end of the previous cycle, until the start of the next cycle, to obtain the status of "Candidate to be added to the ledger." The transactions selected from network N are placed on the block. The block size depends on the number of transactions in it.

Search for Transaction Participants

CREDITS peer-to-peer network can be represented as a graph, with user accounts in the form of vertices and a multitude of possible transactions in the form of directed edges that connect two vertices (account). Since all edges have an initial and a terminal vertex, you can always construct an oriented graph (orgraph).

If we take the following conditions for identification:

- Any transaction always has a sender and a receiver;
- Any vertex (account) can always be connected to another vertex with a directed edge (transaction);
- Any vertex of the graph (account) has a finite number of directed edges (incoming and outgoing transactions).

In connection with the foregoing, we can say that the orgraph contains the required route for fulfilling the necessary transaction conditions and construct a simple chain. Since it is a finite sequence

of vertices, where each vertex (except the last) is connected to the next vertex in the sequence by an edge.

Data Transmission Channel

Each channel of communication between the main network node and the common node of the CREDITS network is a separate thread (multithreading), within which data is sent in encrypted form when the transaction is executed.

To ensure network security, all data between the validator nodes is transmitted in an encrypted form, and each connection between nodes is low-level based on the network library. If the data transfer occurs with an error, the thread should be automatically interrupted, the corresponding entry is placed for writing to the logging system, and then to the log file. Data is transmitted through typified variables. Transmitted data are encrypted using the symmetric RC4 algorithm. Since this algorithm works under a common secret key, this key is transferred when a connection is created between nodes and is transmitted in an encrypted form in accordance with the Diffie-Hellman algorithm.

The RC4 algorithm, like any stream cipher, is built on the basis of a pseudo-random bit generator. The key is written to the generator input, and pseudo-random bits are read at the output. The key length can be from 40 to 2048 bits. Generated bits have a uniform distribution.

The Diffie-Hellman algorithm allows two parties to receive a common secret key using a channel unprotected from listening through but protected from communication channel change. The received key can be used to exchange messages using symmetric encryption. The algorithm is based on the complexity of computing discrete logarithms. In it, as in many other algorithms with a public key, the calculations are performed modulo to a certain large prime number P.

First, a certain natural number A, smaller than P, is selected in a special way. If we want to encrypt value X, then we calculate

$$Y = AX \bmod P.$$

And it is easy to calculate Y having X. The inverse problem of calculating X from Y is rather complicated. Exponent X is exactly called the discrete logarithm Y. Thus, knowing the complexity of calculating the discrete logarithm, the number Y can be publicly transmitted on any communication channel, since with a large modulus P the initial value X will be almost impossible to pick. The Diffie-Hellman algorithm to generate a key is based on this mathematical fact.

Any actions in the system are tied to the timestamp, the number of the previous block, the user's login, and the smart contract ID. This allows finding duplicates when executing. If a duplicate is found, then we take the first transaction from the pool, the rest are considered illegitimate.

Action in the System

An action in the system is a transaction that characterizes the simplest transfer of the value from account to account or the transfer of the result of the contract method to the validator, for the subsequent search for a solution in the consensus search subsystem.

In order to prevent duplication of the transaction in the same block with the same identifier, the system accepts an agreement that the only true and correct transaction is that which came first to the validator subsystem for processing. Since it is already recorded in the validator system that a transaction has already been made from the current account and there are no values left in the account to conduct the transaction, a consensus cannot be found. Thus, the problem of double waste is solved.

When the transaction is executed, information is received to the validator and confirmed, the information about the ledger status change is automatically distributed to all nodes from the trusted list, after which the ledger is synchronized.

In order to always have an up-to-date transaction ledger among all trusted nodes for the current validator node, it is necessary to synchronize the newly arrived transaction in the ledger of all nodes

each time. To solve this problem, a separate port for synchronization should be used (if there is such an opportunity). This opportunity will increase the speed of processing the information incoming to the validator kernel due to the distribution of the load on the port. The synchronization thread is always executed, it is cyclic. Priority for the allocation of RAM and CPU load (using CPU cycles) is lower than the average. The memory stores the last 1,000 operations and the state of accounts for them (in an encrypted form using a synchronous algorithm), this increases the speed of responding to requests from other validator nodes.

Adding a Transaction for Validation

Adding transactions to the ledger is called only from the validator subsystem immediately after consensus building and compiling a white list with the result of transactions saving in the ledger. Calling from third-party systems is impossible, in order to improve security.
Incoming parameters – the object that characterizes the transaction. The resulting value ResultValue<0 – execution is aborted with an error, the resulting value is a possible error code / 0 < ResultValue – the function was executed without errors, the result is the number of the entry in the ledger.

Incoming parameter – the object containing the unique label of the transaction, the sender, the recipient, the transferred value, the value correspondence, the desired value, the amount of the transferred value, the amount of the desired value and other system information that can be changed if necessary.

Cost of Transactions

The system uses the CREDITS currency, which serves:

- As an internal means of payment for the system usage;
- To exchange different currencies within the system;
- To exchange various values within the system;
- To create and process operations under smart contracts;
- To purchase information from third-party sources for services within the system.

The cost of a transaction can vary depending on the network load, on a particular user of the system, which can theoretically direct a huge flow of transactions at a certain peak time. We suggest using the material method and the impact on the system users to control the network load.

The cost of performing transactions in the first three years of the system operation will be set individually for different types of transactions and operations. In the future, an algorithm for the automatic generation of the transaction cost will be developed.

4. Smart Contracts

Introduction

A smart contract in the CREDITS system is an electronic algorithm that describes a set of conditions by which actions and events in the real world or digital systems can be associated.

To implement self-controlled smart contracts, a decentralized environment that completely excludes the human factor is required, and to use the transfer of the cost of a smart contract, a cryptocurrency independent of the central authority is required.

Entities

A smart contract in CREDITS consists of the following entities:

1. Property (public variables) – the system entity storing public data necessary for work of the contract in the CREDITS system.
2. Method is the CREDITS system entity responsible for observing the logic and sequence of actions when conducting the transaction (actions under the contract).

Participants in the CREDITS system sign the smart contracts using the method call that modifies the contract properties, by launching the processes for verifying compliance with conditions and coordination.

A smart contract comes into force after signing by the parties. To ensure automated fulfillment of obligations, an environment of existence is required fully automating the execution of contract terms. This means that smart contracts can exist only within an environment that has unimpeded access to the executable code to the smart contract items.

All contract terms must have a mathematical description and clear logic of execution. Thus, the main principle of a smart contract is complete automation and reliability of contractual relations between the parties.

Smart Contract Method

The CREDITS smart contract method is the system entity responsible for compliance with the logic and sequence of actions during the transaction (actions under the contract).

The logic and sequence of actions are described by a program code (module) containing commands; their sequential execution allows to obtain the desired result. This code can handle system commands (for example, the assignment command), user commands (separately written functions), contract properties (statically or dynamically initialized variables available from any contract method), and methods from any other third-party contract available only to the owner of the connected (third-party) contract. For more popularization, the development is provided in scripting languages (for example, JavaScript).

The method (program code) allows the use of all widely used scripting language operators (commands) (assignment, conditional and unconditional jumps), the creation of functions and procedures (subroutines), connection of third-party libraries.

Virtual Executable Machine

The contract method of the CREDITS system is executed in the virtual environment of the system (Virtual Machine, hereinafter referred to as VM). When a method is called for a particular contract, VM allocates a memory area and loads the contract bytecode in it containing the methods and the variables initialized (or redefined when calling other contract methods). VM starts processing the method bytecode, at runtime, variables and code are loaded into its memory area, and commands are successively executed, their result is transferred to the peer-to-peer network for subsequent placement in the ledger.

The initiator of the execution method is the user of the system, on behalf of which this method is launched.

Value Term

CREDITS cryptocurrency is also an indicator of the value term of a contract unit to compare two completely different units and build a consensus when executing or accepting the contract by the parties. Instead of registering each separate value/gateway combination, CREDITS cryptocurrency serves as a bunch for effecting value transfers. This is possible because any value is liquid with respect to CREDITS currency, which means that any value can be liquid with respect to any other value.

Performing the Smart Contract Terms

The contract term in the CREDITS system is the values of the trigger (checked) fields required to close (complete) the contract.

Fulfillment of the smart contract terms is a procedure when the trigger (desired) fields are checked for an equivalent desired value. There are three possible ways to find a solution to fulfill the contract terms:

1. The contract is concluded between two or more parties for the transfer of value. In this case, the contract fulfillment is the provision of the cost equivalent of the value to the transferring party from the receiving party.
2. The contract is concluded between the parties for the transfer of value, but payment must be made upon fulfilling a certain number of conditions (for example, delivery of value to the receiving party).
3. A contract for conversion of one value to another with a cost equivalent in the form of CREDITS is placed in the system. In this case, the platform starts looking for the shortest possible path of exchanging one value for another through conversion in other contracts. Any fulfillment of the contract can be provided per one transaction, or per several transactions, which will provide an opportunity to collect the required quantity of units of value to complete the contract.

Data Sources

For correct and fully-featured work, checking and providing additional information, to make a more balanced and optimal solution, CREDITS uses third-party data providers. The need to introduce additional data sources into the system is due to the inadequacy of public information about one or several contract parties (for example, obtaining the borrower's credit status for making a decision to issue a credit).

To work with third-party information systems, the platform can call an integration bus, which by remote access generates a request to a third-party system (site) in a format for data presentation on a paid basis for the system participants with payment in CREDITS.

The request is sent in an encrypted form to ports and addresses provided by information systems other than the standard ones. The result of the request can be any response to the service containing the necessary information to make a decision, or an error code characterizing impossibility of receiving the required response and possible steps to eliminate the error.

5. Implementation Plan

Technical Plan of Project Implementation

	S1	S2	S3	S4	S5
	Pre-Alpha	Alpha	Beta	Release candidate	Release
Storage, Consensus mFA Consensus	FA : Key Design Implementation	mFA : Key Design Implementation PoW	mFA Optimization	–	–

		(Proof-of-Work) and PoC (Proof-of-Capacity)			
Data Store	Decentralization Ledger, NoSQL Store implementation	MessagePack History	–	Blockchain backup	–
CVM (Credits virtual machine)	Design and Implementation	Integration with ecosystem	Optimization	check errors	–
Third-party system	–	Design and Implementation	integrate to full system	–	optimisation
Inference Engine	Formal Specification and Key Design Elements	Reasoner Integration with Blockchain	Reasoner Optimization	–	–
User interface	Implementation	Web UX design	–	–	–
Wallet	Wallet Formal specification		UX design Application Test	–	Android, iOS, Desktop Wallets
RPC & REST API	Formal specification	Blockchain Explorer	–	Third-party explorer	–

CREDITS cryptocurrency

After launching the release version of the system, a fixed amount of 1,000,000,000 CREDITS will be issued. They will be exchanged for ERC20 standard tokens, issued at the initial token sale. They will be exchanged at a fixed exchange rate: 1 ERC20 standard token = 1 CREDITS monetary units.