



2022

Decimal Yellow Paper

version 2.0

Content



1. Introduction	3
2. Decimal services overall structure	4
3. Stack of technologies	7
3.1. Programming languages	7
4. Cosmos SDK	8
5. Tendermint	10
6. DEL emission	12
8. Multisignature	16
9. CRR	18
10. Formulas for determining coin value	19
11. What a masternode is	21
13. Explorer	24
14. Console	26
15. Status	30

16. Wallet	32
17. Address format (Bech32)	33
18. Console client	35
19. Roles	37
19.1. Validator	37
19.2 The delegate	38
19.3. Coiner.	38
19.4. User	39
20. Delegation	40
21. NFT and SFT	42
23. Cross-chain swap	46
23. Help / FAQ	50
24. Block structure	51
24.1. Data structure	52
24.2. Block	53
24.3. Title	54
24.4. Transactions	55
25. References	56

1. Introduction



This document contains a technical description of the Decimal blockbuster, which is being developed by our development team. When designing the document, we aimed to give a general idea of the project, as well as provide a more detailed description of the key elements of Decimal.

The document is written in technical language, but without excessive immersion in the nuances of technology and technical solutions. If you are looking for more general information about the project, please read Decimal White Paper (<https://decimalchain.com/DWPeng.pdf>).

2. Decimal services overall structure



The figure below shows the general architecture of Decimal. The system-forming elements are directly block blocks, a database structured as a chain of blocks. Within the system blockmen physically exists in the form of database replicas, which are stored on each of the complete nodes (master, validator).

In order to route the high number of read requests to the blockchain, and actually to the blockchain replicas, we have organized the following structure.

There are a number of services in the network - Workers, which collect data coming directly to the blockchain. Data are blocks and transactions of different types: sending, buying, selling, creating coins, etc. Several identical services are required in case some of them fail, no bit of information should be lost in this part of the process, however, as in any other part of the process.

Sampling of data at the output of workshops goes to the indexer, which structures, sorts and indexes the incoming information, after which all data are stored in the main

(leading) database (Master). The data are then copied multiple times and placed in slave storage (Slave).

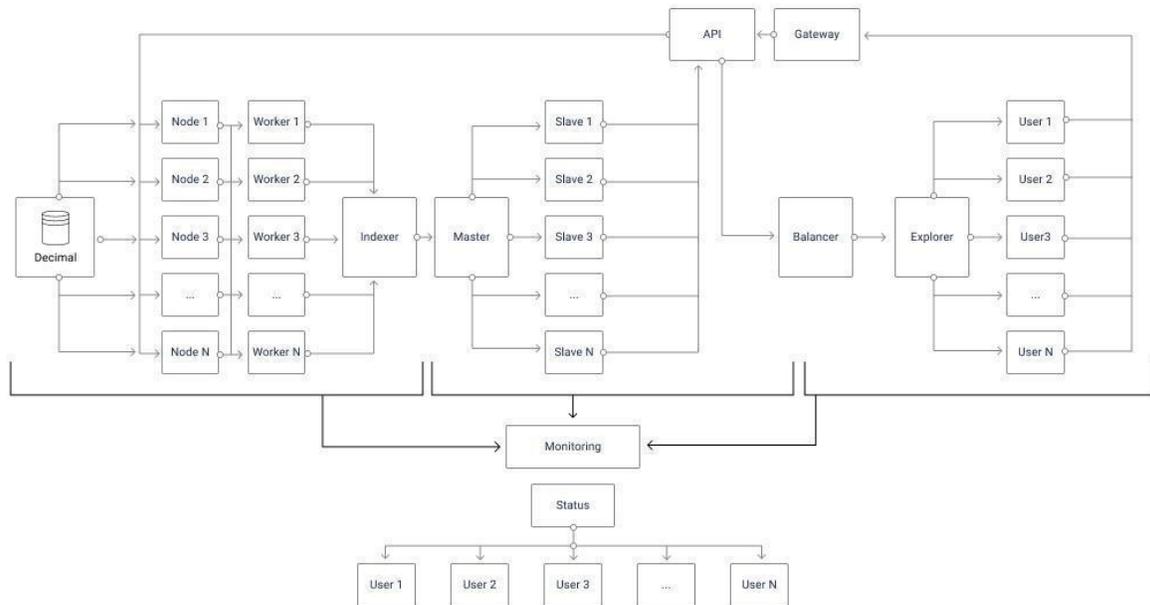


Fig. 1 - general architecture of Decimal.

Explorer users make requests to search for all kinds of information about a blockchain. All these requests go through a special balancer that evenly distributes the load and sends the corresponding requests for reading from the Slave storage.

Thus, data buffering and read access channel from the blockchain with high bandwidth is performed.

The system also has a channel for writing information to the blockchain. By means of a special Gateway API interface, which is directly connected to the system nodes, the work of Console, desktop wallets and wallet applications for mobile is provided. The structure of all these services is fully decentralized, users own seed-phrases and private keys. After formation of corresponding transactions (requests on record in the block-keeper), the user signs them with his signature and sends them to the network. Then the transactions are processed, verified, included in blocks and after reaching a consensus between the validators, they are written into a chain of blocks with replication on each full node.

There is a special service Status in Decimal architecture. The monitoring service monitors, collects and provides general network parameters.

3. Stack of technologies



3.1. Programming languages

For correct compatibility with the Cosmos SDK and Tendermint, we chose Golang as the programming language for the implementation of Decimal's functionality, namely masterclass software (validators).

To write the backend modules, we chose TypeScript, which is strictly typed and convenient in the development process, compiles in JavaScript, runs in modern browsers and is compatible with NodeJS. Specifically, TypeScript has wrappers (Workers) and indexer (Indexer).

To implement wallet desktop applications, the Decimal team used ElectronJS, which allows cross-platform desktop applications based on JavaScript, HTML, and CSS.

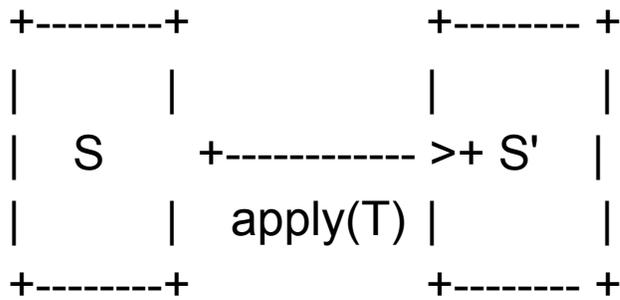
4. Cosmos SDK



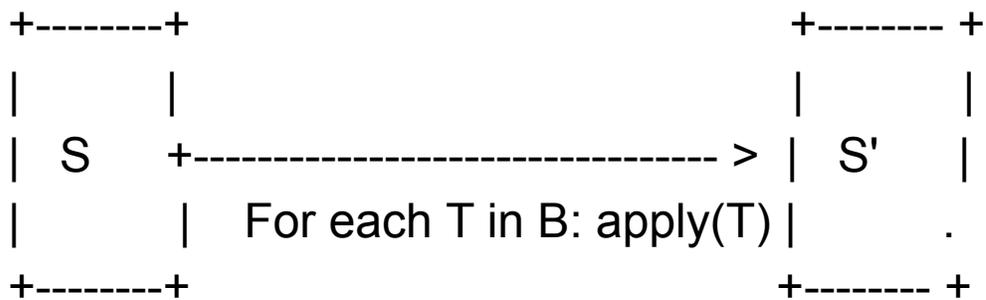
Blockchain Decimal is developed on the basis of Cosmos SDK - a framework (a set of tools) for developing blockchain applications focused on specific tasks (problems). Cosmos SDK provides a secure and reliable solution to most common tasks for block committees, such as organizing network communication between nodes of the network and ensuring a reliable consensus between the nodes involved in the formation of blocks of the network. In the Cosmos SDK, this is achieved through active use of the Tendermint Core library (more below).

Thanks to building on the Cosmos SDK, Decimal Block Core is compatible with all blockchains in the Cosmos Network, which already has more than 100 projects: <https://cosmos.network/ecosystem/apps/> .

In its essence, a blockchain is a replicated machine of interrelated states.



The machine of states is a concept of computer science, according to which a machine can have several states, but only one at any time. There is a state that describes the current system state, and transactions that trigger state transitions.



Given state S and transaction T, the state machine returns a new state S'.

In a blockwise context, the state machine is deterministic. This means that if a node runs into a specified state and repeats the same transaction sequence, it will always have the same end state at its output.

The Cosmos SDK gives developers maximum flexibility in determining the state of their application, transaction types and state transition functions.

5. Tendermint



Tendermint is an advanced consensus solution based on BFT (Byzantine Fault Tolerance). BFT consensus guarantees correct operation of a computer network as long as at least $2/3$ of the network nodes involved in block formation (validators) work correctly.

Tendermint consists of two main technical components: consensus mechanism engine and application interface. The consensus mechanism engine, called Tendermint Core, ensures that the same transactions are recorded on each machine in the same order. The application interface, called Application Blockchain Interface (ABCI), allows transactions to be processed in any programming language.

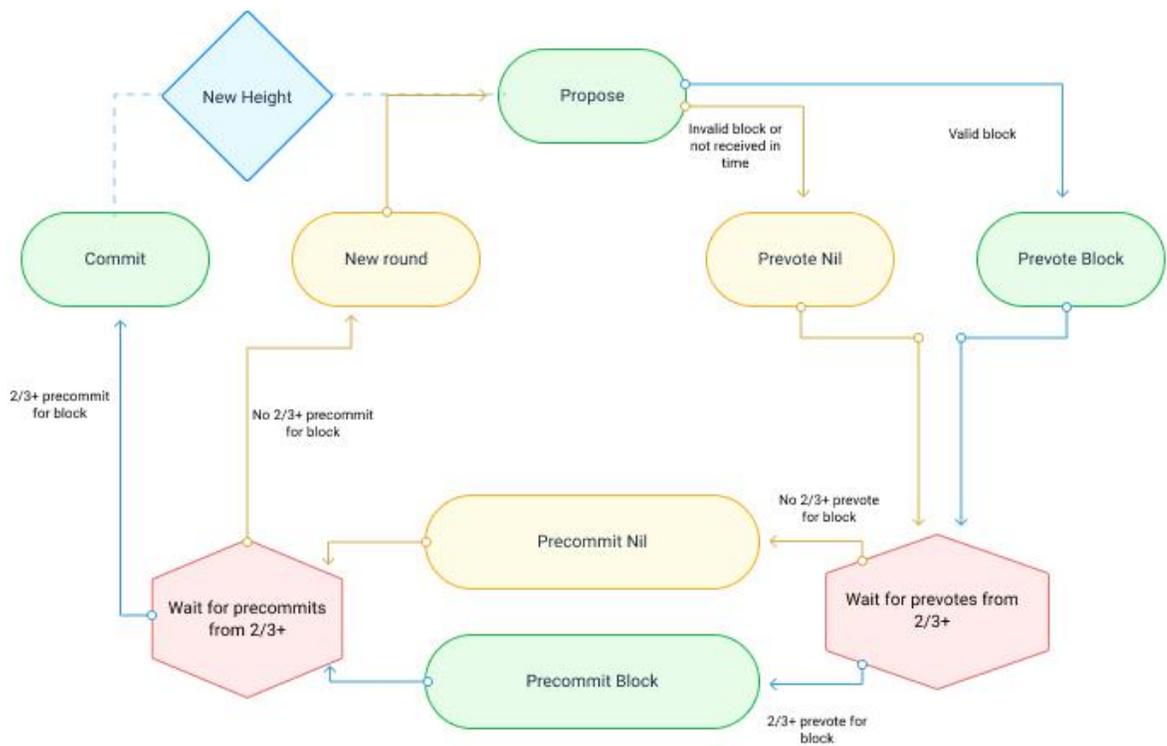


Figure 2 - Consensus building process in Tendermint.

In other words, Tendermint provides efficient relaying of blockchain changes across the network, ensuring that each node on the network has the same transaction log and blockchain state.

The pBFT (practical Byzantine Fault Tolerance) consensus mechanism is a key part of the Decimals blocklayer and we use it without any changes.

6. DEL emission



Native coin Decimal, called DEL. In the tDEL test network. DEL is emitted during the generation of each block. Block Reward = Base Reward per block + Total Reward of all transactions in the block. The initial base reward for a block will be 50 DEL. And then every 432 000 blocks (about 30 calendar days) it will increase according to the following algorithm:

- first 12 months (1st year) - increase by 5 DEL;
- next 12 months (2nd year) - increase by 17 DEL;
- next 12 months (3rd year) - increase by 29 DEL;
- next 12 months (4th year) - increase by 41 DEL;
- next 12 months (5th year) - increase by 53 DEL;
- next 12 months (6th year) - increase by 65 DEL;
- next 12 months (7th year) - increase by 77 DEL;
- next 12 months (8th year) - increase by 89 DEL;

The table below shows the basic unit compensation during the first two years of operation of the Decimal blockchain.

Month	Reward, DEL	Month, 2021	Reward, DEL
August, 2020	50	August, 2021	122
September, 2020	55	September, 2021	139
October, 2020	60	October, 2021	156
November, 2020	65	November, 2021	173
December, 2020	70	December, 2021	190
January, 2021	75	January, 2022	207
February, 2021	80	February, 2022	224
March, 2021	85	March, 2022	241
April, 2021	90	April, 2022	258
May, 2021	95	May, 2022	275
June, 2021	100	June, 2022	292
July, 2021	105	July, 2022	309

Table 1 - base compensation for the block.

When starting the network in the genesis block will be made pre-mine DEL, which will be 200 000 000 DEL. In this case each of the 4 starting validators will receive 40 000 000 DEL (in total 160 000 000 DEL).

The remaining 40,000,000 DEL will be put up for sale and sold to the investors of the project.

7. Types of transactions



The following types of transactions are implemented in Decimal:

- 1)
- 2) Send (sending coins);
- 3) Buy (buying a coin);
- 4) Sell (selling a coin);
- 5) Sell All (full sale of coins);
- 6) Multisend (sending to multiple recipients);
- 7) Delegate (delegating a coin to a validator);
- 8) Unbond (recalling a coin from a validator);

- 9) Redeem Check (repayment of the check);
- 10) CreateMultisig (creates a multi-signed address);
- 11) CreateTransaction (creates an output transaction from a multi-signed address);
- 12) SignTransaction (unique identifier of transaction multisignals);
- 13) CreateCoin (creates a coin);
- 14) DeclareCandidate (creates a candidate for validators);
- 15) EditCandidate (editing of candidate data for validators);
- 16) SetOnline (activation of the validator);
- 17) SetOffline (deactivating the validator).).

8. Multisignature



Multi-signed addresses will be available in Decimal. This is very convenient when allocating funds or simply making a joint decision when there are several independent participants and the conditions for consensus between them. For example, funds from a shared wallet will only be withdrawn if 80% of the participants agree (4 out of 5).

The multi-signature will work almost like a smart multi-signature contract in an Ethereum blockbuster. Three types of transaction will be implemented to implement the functionality - CreateMultisig, CreateTransaction, SignTransaction:

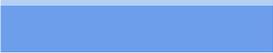
CreateMultisig creates a wallet with multisignature, indicating the owners, their voice weight and threshold value (for example, 3 out of 5 or 2 out of 3). In this case, the address with the multisignature is generated with the addition of "salt" to allow you to create many addresses with the same parameters, and stored in the storage.

CreateTransaction creates a transaction to output the specified number of specified coins to the specified address, the transaction creator immediately signs this transaction. Each such transaction is assigned a unique identifier.

Other owners of a multi-signed wallet may request unconfirmed transactions at this address, their parameters and identifiers. After that they simply send a transaction of SignTransaction type to a block-keeper, where in the parameters there will be a unique identifier of the transaction from p. 2. After checking the signature the number of collected "votes" of the transaction is increased in accordance with the weight of the subscriber in p. 1. If the threshold value is reached, the transaction is executed, the funds are transferred, the balance of the address with the multisignature and the recipient's address are changed.

As a result of this scheme of interaction between the multi-signature participants, we exclude offline communication between them. At least it becomes optional.

9. CRR



Distinctive feature of the Decimal Blockchain is the specificity of the economic model of coins. All the advantages are derived from it: a simple issue of coins, the ability to exchange them for any other coins within the network, payment of commission by any Decimal ecosystem coins.

In addition to traditional coins, the DEL native coin serves as a collateral. Each caste coin is backed by a guarantee in the form of a certain amount of DEL. The size of this warranty in relation to the total issue of a cash-in-trust coin directly affects the value curve of the coin.

The CRR parameter is set when the coins are created and the value curve remains unchanged forever. When market conditions and balance change, the demand/supply value of the coin moves up or down the calculated curve.

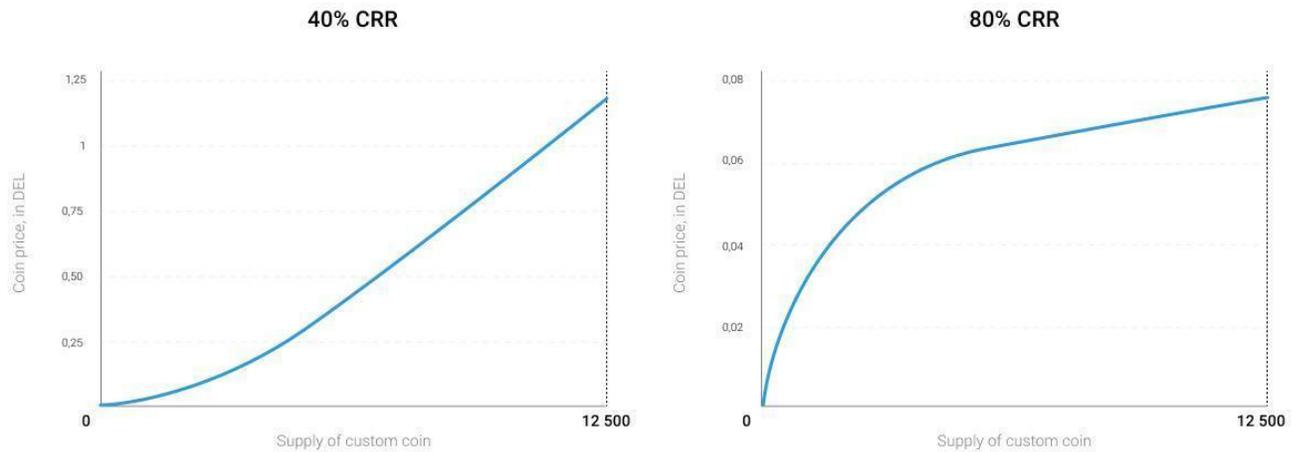


Figure 4 - coin value at different CRRs.

10. Formulas for determining coin value



The key parameters in the Decimal network are Reserve, CRR and the total number of issued coins. All of them are also involved in the calculation of the purchase or sale price.

The very existence of two formulas is a consequence of the nonlinear nature of changes in coin value.

Calculation of coin purchase price:

$$\text{Cost} = \text{Reserve} * (-1 + (((\text{Buy} + \text{Issue}) / \text{Issue})^{(100 / \text{CRR})}).$$

where

Reserve - current reserve in DEL;

I want to buy - the number of coins to buy;

Issue - total number of coins;

CRR - coefficient of Permanent Ratio to the Reserve.

Calculation of coin sale value:

$$\text{Cost} = \text{Reserve} * (1 - (1 - \text{sell} / \text{issue}) ^ (100 / \text{CRR})).$$

where

Reserve - current reserve in DEL;

I want to Sell - the amount of coins to buy;

Issue - total number of coins;

CRR - coefficient of Permanent Ratio to the Reserve.

11. What a masternode is



There are 2 types of network nodes in Decimal. A full node and a partial node (normal).

Masternode (aka Full Node) is a Decimal network node that stores a replica of a blockchain and participates in consensus building.

The masternode is a hardware-software complex, which acts as a validator in the network. The hardware is connected to the Internet and directly to other validators to ensure the main task - consensus.

Requirements for equipment:

4GB RAM - RAM capacity;

1 TB SSD - hard disk capacity and type;

x64 2.0 GHz 4 vCPUs - CPU characteristics.

Each Decimal network mastermind stores a full copy of the blockchain. All transactions, all blocks, starting with the genesis block, all messages. This copy is called a replica. It is identical to the replicas on each of the other masterworks.

Besides, additional services and services that are necessary for establishing connection with other masterworks

via the Gossip protocol, verifying user transactions, forming blocks from transactions, writing everything and everything directly into the local block listener replica are deployed on the masterworks.

Each master craftsman works under strict conditions of punishment/ encouragement.

Remuneration is based on correct and reliable performance. The amount of reward is proportional to the total sum of stakes of each master breed. The bigger is the validator's stake, the bigger part of rewards for the block will be received by this validator (masternode).

Penalties are imposed on the incorrect operation of the mastermind. For example, inaccessibility of a validator within 12 blocks of the last 24 is punished by 1% of the total value of the validator stake. And the total sum of the stake includes all the funds delegated to this validator.

Validators who attempt fraudulent actions in the consensus building process are also penalized. Namely, when signing 2 different blocks during the rounds of verification and voting for a block of candidates. This is a serious violation, which can lead to fork of a chain of blocks. In case of fork appearance part of network users will be oriented on one variant of block state (transactions, account balances), while other part of users will be oriented on the second variant of block chain already with other state and balances. In this case the validator will be punished with 5% fine. Plus all coins

delegated to this validator will return to their owners (its stake will be reduced).

One more time,

- 1) Inaccessibility within 12 blocks of the last 24 blocks - 1% stake penalty + validator shutdown;
- 2) Double signature - 5% stake penalty and forced splitting of coins.

It should be noted that penalties are not transferred anywhere, but simply burned. Reducing the total issue of DEL.

12. A regular node



The second type of node is an ordinary node. These are all members of the network, except for masterminds (validators).

These are ordinary users of the network: purse holders, cash-in coin issuers, businesses and individuals, coin delegates and others.

Each of them owns private keys to their wallets and uses all services and applications available in the Decimal ecosystem. They keep DEL and other caste coins on their non-caste

wallets. They send them to other users, delegate them to the validators they like, and receive rewards from validators.

In general, ordinary nodes have all the functionality available, except the obligation to participate in consensus building and to store a replica of the blockchain.

13. Explorer



A wide range of Decimal users need to always understand what's going on in a blockbuster network at a given time, what its main parameters are. Users want to double-check and search for information about their transactions, blocks, commissions, rewards, etc.

So we developed the Explorer unit.

Assuming that users will be interested in a very large amount of information about the status and processes in the Decimal network, namely transaction details, blocks, validators and their parameters, issued coins, etc., we must take care of the availability of all this data and ensure that it is displayed correctly and quickly. As time passes and the blockchain increases, it will become more and more difficult to process

requests. Significant time delays are possible when sampling data directly from blockchain replicas. But we organize storage in a database that can satisfy a huge number of requests and guarantee to provide the information needed by users.

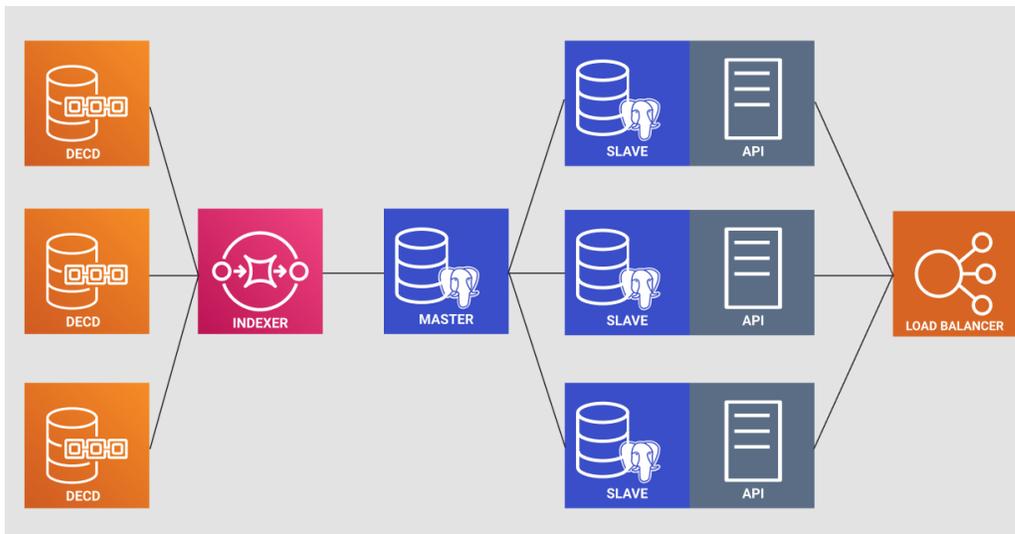


Fig 5 - architecture of Decimal explorer block.

The figure above shows the architecture of our solution. The process is organized as follows.

State changes in the blockchain generate events (events, events), which are monitored by special services (Workers, workers). These services parse all incoming information from blocks and transactions and transfer it to the Index, in which data is sorted and indexed. The ordered data are then written to the PostgreSQL database. They are written to the master database (Master) and duplicated on the slaves (Slave) to ensure safety.

All requests from the Explorer are received by the Slave Databases through a Load Balancer that organizes load balancing evenly through a software interface (API). The Load Balancer is horizontally scalable, i.e. slave databases can be added if the load increases. In this case, the masterminds where the blockchain replicas are directly stored are not loaded, because they are architecturally separated from the requests of users and external services. Through indexing and partitioning, databases are able to provide information about any events and states in the blockchain with minimal delay, regardless of the size of the database itself.

14. Console



Decimal already at the start of the project provides the user with a number of services that reveal the main features. We consider it very important to provide convenient access to all services. Therefore, all of them have been brought together on the same site. It is called the Decimal Console (<https://console.decimalchain.com>).

Wallet (<https://console.decimalchain.com/wallet>) - purse address, balance of your funds in DEL, list of coins in possession, list of completed transactions and their parameters, functionality of sending coins.

Conversion (<https://console.decimalchain.com/convert>) - service for exchange, purchase and sale of any coins in the Decimal network.

Delegation (<https://console.decimalchain.com/delegation>) - interaction with validators, transfer of their coins in order to increase their stake and get reward from them. You can also revoke your delegated coins here.

Delegating user funds to validators is a key feature of the Decimal block listener. With a large number of coins we want to provide the user with a convenient tool for organizing delegation. So we have implemented automatic delegation. At the preliminary stage, we offer to form, either offline or online, a package of transactions and configure a separate transaction to start automatic start of delegation.

Masternode

(<https://console.decimalchain.com/masternode>) - the service organizes the process of connecting and running the validator functionality. The number of validators in Decimal is regulated and increases in proportion to the network growth. The strategist will have 4 validators and 12 slots of additional slots, i.e. the maximum number of validators at the start is 16. Then every calendar month 4 validators will be added.

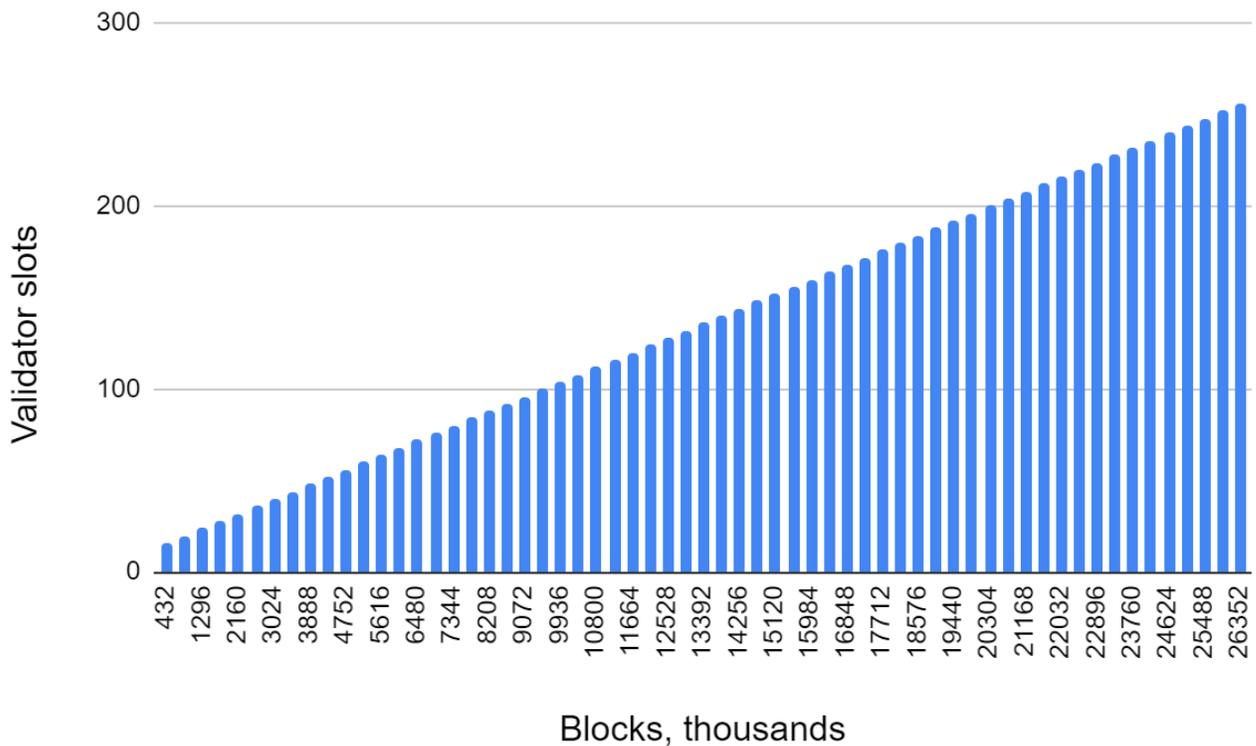


Figure 6 - increase in the number of slots for validators.

Before you can become a validator, you have to declare your candidacy. Validators with the largest stake size participate in the consensus mechanism. The stake is recalculated for each block. Accordingly, the candidate is selected based on this parameter, as well as on quality indicators (online availability, no fines) of his work.

We deployed a Decimal test network that is architecturally completely identical to the main network. This is the ability to debug the startup and management processes of the master. Click on <https://testnet.console.decimalchain.com> for more information.

Coin issue (<https://console.decimalchain.com/issue>) - here the functionality of coin creation is implemented. The algorithm is simple, fill in five fields and the coin is created. To understand how the value of your custom coin will behave when buying, selling and exchanging, we have made an additional service - Calculator (<https://calculator.decimalchain.com>). In it you create a virtual coin, perform the corresponding transactions and observe the change of coin value.

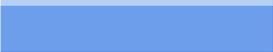
Broadcast (<https://console.decimalchain.com/broadcast>) is a service for sending offline transactions generated by users to the Decimal network. Since security is a key priority, we offer a tool that will eliminate any possibility of compromising users' private data. Here we generate a **nonce** parameter for the user, which is included in the transaction. Once the transaction is created offline, the user can send it to the network without having to worry about their private keys.

Status (<https://status.decimalchain.com>) - this resource displays the main Decimal global metrics at the current time, such as the status of the network, services and services of the blockman, the number of coins issued, etc.

API & SDK (<https://help.decimalchain.com/api-sdk>) is a description of the program interface for interaction with Decimal services and tools for building applications based on the Decimal blockchain.

Help / FAQ (<https://help.decimalchain.com>) is a traditional section needed to explain technical and non-technical details, various guides, manuals, answers to frequently asked questions, etc. Despite the fact that everything is simple from the point of view of use in Decimal, there are many different technologies and nuances on the technical level. We strive to make it easier to learn Decimal and help the widest possible range of users.

15. Status



As mentioned above, the service Status is organized on the basis of the monitoring service. It consists of several parts deployed on Decimal capacity and network nodes:

- 1) monitoring server;
- 2) database;
- 3) web-interface;
- 4) demon agent, at the monitoring sites.

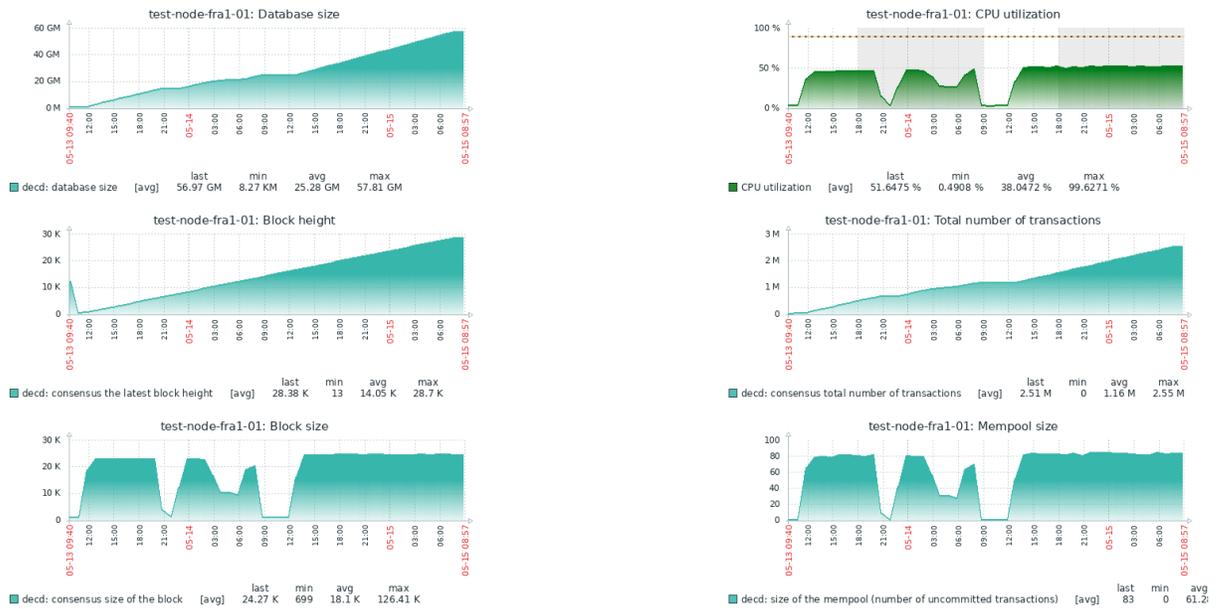


Fig 7 - an example of monitoring at the Decimal development stage.

To get acquainted with the service click on the link <https://status.decimalchain.com>

Information about transactions, blocks, commissions, validators, time parameters, coins is collected directly from the blockchain.

For example, for the **Network** parameter, we check every 30 seconds if new blocks are formed on the validator nodes. If new blocks are formed, the network is in the **"Active"** status.

For the **Explorer** and **Gateway** services, we check whether they are running on servers and whether they are available (responding or not).

16. Wallet



All official Decimal wallets are decentralized, i.e. non-custodial. Seed phrases and private keys are stored strictly on the user side and decimal has no access to them in any way. The entire responsibility for the safety of these private data and funds on wallets lies with the owners of wallets.

At the start of the project the following wallet applications are available:

1) for browser desktop <https://wallet.decimalchain.com>;

2) for browser desktop via console

<https://testnetconsole.decimalchain.com/wallet>;

3) for Android devices

<https://play.google.com/store/apps/details?id=com.chain.decimal&hl=en>;

4) for iOS devices

5) Windows, MacOS and Windows desktop version

Private keys are generated based on the BIP39 standard. Elliptical curve **secp256k1**.

The Seed Phrase consists of **24** words.

17. Address format (Bech32)



One of the specific features of blockchains (first Bitcoin, and then many others) are address formats. They represent a sequence of letters in the Latin alphabet and numbers. The problem is that this makes it very difficult for users to read them correctly.

Blockbuster enthusiast and developer Pieter Wuille suggested upgrading the format of addresses in the Bitcoin network. This proposal is known as BIP 173¹ or bc1 addresses and as of May 2020 it was successfully implemented² in a significant number of cryptographic projects, including those outside the Bitcoin Blockchain.

At the moment the implemented changes are known as Bech32 address format.

The whole Decimal team, supporting the Bech32, will provide it at the start of the project.

The Bech32 address is no longer than 90 characters long and contains:

¹ <https://github.com/bitcoin/bips/blob/master/bip-0173.mediawiki>

² https://en.bitcoin.it/wiki/Bech32_adoption

- 1) The part that's easy to read by humans. This includes data that may need to be transmitted or that have anything to do with the owner of the address, a minimum length of 1 character. For example, by default "bc" is used for mainnet addresses, and "tb" for testnet addresses.
- 2) A divider that always equals "1". If "1" is allowed inside the human readable part, the separator is the last of the characters "1".
- 3) The data portion is at least 6 characters long and consists of alphanumeric characters only, except for "1", "b", "i", and "o".
- 4) Checksum. The last six characters of the data part form a checksum and contain no information.
- 5) All letters are lowercase, although it is possible to convert them into uppercase letters for generating QR code.



Figure 8 - address structure of the Decimal user.

18. Console client



In order to facilitate the process of creating and debugging certain processes or scenarios of interaction with Decimal, whether it is on-line or off-line services, execution of scripts, execution of block requests, sending any types of transactions or receiving service information and parameters, our team prepared a console client.

File to run the client is `deccli`.

The client acts as a REST server, which provides the ability to create a wide range of requests to the blocker.

Essentially, the console client is a "light node", or "light daemon" and redirects requests to the full node, and therefore does not require synchronization with the blocker (does not store a replica of the blocker), takes up very little disk space, but has a rich and comprehensive functionality.

```
timofvy@timofvy: ~
File Edit View Search Terminal Help
timofvy@timofvy:~$ deccli q coin list
- CRT
- DICK
- SEMA
- TESTCOIN
- TIMOFCOIN
- TIMOFCOIN1
- VJACHCOIN
- VJACHCOIN1
- WEBCOIN
- tDEL

timofvy@timofvy:~$ deccli q coin get tDEL
title: Test decimal coin
constant_reserve_ratio: 0
symbol: tDEL
reserve: "0"
limit_volume: "0"
volume: "201045150000000000000000000000"
```

Fig 9 - interface of the console client, coin list and coin get commands.

```
timofvy@timofvy: ~
File Edit View Search Terminal Help
timofvy@timofvy:~$ deccli q account dx1t600z7lhfh5334dt7fjcy5xjqm5gm3x4q66rua
|
address: dx1t600z7lhfh5334dt7fjcy5xjqm5gm3x4q66rua
coins:
- denom: tdel
  amount: "100369718953725054730840"
- denom: timofcoin
  amount: "50000000000000000000"
- denom: timofcoin1
  amount: "979996446009055421813"
- denom: vjachcoin
  amount: "50000000000000000000"
- denom: vjachcoin1
  amount: "2038222263830883461047"
public_key: dxpubiaddwnpepqtzkzd4ryvjlk6aq6pwd028g0s0pgm92nlgn2rgmammpl8eupt4u2vwxc8gj
account_number: 16
sequence: 30
```

Fig 10 - console client interface, account information

19. Roles



19.1. Validator

A member of the Decimal network who keeps a blockchain replica on his equipment, provides transaction verification, chain block formation, and participates in the consensus building process. One of the main characteristics of the validator is the voice force (Voting power), which is directly proportional to the size of the stack of this validator. The bigger is the validator's stack size in absolute numbers and the bigger is its share in the aggregate stack of all validators, the more voice power it has, the more often it produces blocks, the more reward it receives for its work.

The Validator must ensure the reliability of its work, be available on the network, correctly perform its duties in the consensus building procedure, guarantee access of the Decimal network internal services to its block listener replica, ensure the integrity of stored data.

Functioning as a validator is not only the receipt of remuneration for work, but also the risk of being fined for improper performance of duties. A validator risks losing part of its stake as a result of fines, as well as reputational losses if

users stop trusting it with their funds for delegation. Reputational losses can have a very strong impact on validators.

Validator stack consists of own funds and funds delegated (leased) to it by other members of the network.

19.2 The delegate

A Decimal member who leases his funds to a validator. The transfer is not done literally. The amount of transferred funds is simply blocked in the account of the deregulator and he can not dispose of it until he disconnects his funds.

The transfer process is called delegation. The essence of it is to transfer the rights and responsibilities of the user. A delegate increases the validator's stake and it turns out that the validator performs his duties on behalf of the delegate, within the amount of delegated funds. Accordingly, rights and responsibilities apply to both the receipt of remuneration and fines. Rewards and penalties are distributed in proportion to the delegate's share of the total stake of the validator. If the validator receives a 5% fine, the delegate will receive the same 5% fine automatically.

19.3. Coiner.

A member of the Decimal chain, which produces its own caste coins. As you already know from Decimal White Paper,

the applications for crypto coins are limitless. We have made the coin making process exceptionally simple and fast.

After setting initial parameters, such as the size of the reserve in DEL, the number of coins issued and the permanent reserve ratio (CRR), a special transaction is sent to the network. This completes the process. You get a ready coin with full functionality.

Moreover, you do not have to worry about the technical mechanisms of caste coin price formation. Any exchange operations involving your coin will be based on mathematical formulas embedded in Decimal software and automatic blockchain algorithms. More demand, higher coin price and vice versa.

Any caste coin can be delegated to a validator. It can be exchanged for any other Decimal casino coin or DEL.

Transaction fees are also paid in any cashier's coin at your disposal.

19.4. User

All other members of the Decimal network are simple users. We view each user as a potential liner, delegate and validator. After successful experience in storing Decimal coins, sending them, paying for goods and services, as well as other applications in real life, the step towards creating your own coin will be easy. Our mission and task is to remove all barriers to the ordinary user on the way to their goals.

20. Delegation



Delegation is the process of binding user coins (DEL or any custom coins) to the validator(s). The process is performed using a special transaction **Delegate** (see [Types of transactions](#)). The user delegating the coins is called a **Delegate**. After binding the coins, the delegated funds are blocked in the user's account, i.e. they are not sent anywhere, but are **NOT** displayed on the user's balance. However, these funds are displayed in the common validator stake, increasing its weight in relation to other validators and the strength of the consensus mechanism voice. The validator is not able to manage the delegated funds in any way, does not have access to them, cannot spend them or withdraw them.

Recall that among the validators organized competition - validators with a large stake are most often involved in consensus-building and receive more remuneration.

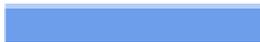
Decimal has an unbond operation for delegated coins. This transaction can be initiated and sent to the network either by the user, or automatically by the validator or software. In the

first case, the funds will be immediately available on the user's balance. In the second and third cases, the funds will be available to the user in 432,000 blocks (~ 30 days).

Since validators are fined for poor quality or incorrect work (see [What a masternode is](#)), the number of delegated coins may decrease by the amount of fines. The delegate is personally responsible for his/her choice of validator, so this choice must be made very carefully.

Coin delegation takes place in so-called slots. Each validator has 1000 slots. Each individual coin is delegated to its own slot. That is, all DEL coins will be placed in one slot, and for each next coin there will be a next slot. There is no limit on the number of coins in a slot. But if the delegate has all 1000 slots full, the next delegate must send funds **NOT** less than in the last slot. If he sends more, the funds in slot number 1000 are forcibly separated and immediately appear on the owner's balance.

21. NFT and SFT



NFT (Non-fungible token) - unique tokens that confirm ownership of a digital asset. Unlike fungible tokens such as DEL, each NFT instance is unique and cannot be replaced by another similar token.

SFT (Semi-fungible token) - semi-fungible tokens are produced in limited collections. Such tokens are interchangeable only with other tokens of the same collection.

The ability to create, send, receive, and burn NFTs and SFTs is built into the Decimal blockchain. Users can operate NFTs and SFTs from their wallet in the Console, as well as view public token demos (covers, demo-parts) in the Explorer.

In order to create your own NFT or SFT, on the page <https://console.decimalchain.com/nft> you need to specify the name of the token to be displayed in transactions, the title to be displayed in the Console, the reserved amount of DEL for further delegation, the number of issued tokens in the collection, as well as a brief description of the token. You also need to determine whether it will be possible to add new tokens to the collection and whether the token will be private and download the associated file.

Supported file formats:

- .png - raster graphic image format;
- .jpeg - raster graphic image format;
- .gif - raster graphic image format;
- .mp3 - audio recording format;
- .mp4 - video format;
- .ai - vector graphic image format;

Privacy is a property that NFT and SFT tokens can have. The property is defined at the time the token is created and cannot be changed later. Only the owner of a private token has access to the full size file associated with it, its cover and description via the console. Other users can only see data about transactions of private tokens through the Explorer, other data is not available to them.

For NFTs and SFTs without privacy feature, all users can view/listen to demos and covers (small images) in the Explorer. Only the owner has full access to the file, as in the case of private tokens.

In what form is the metadata about NFTs and SFTs stored?

The table below shows the data structure for NFT and SFT in Decimal.

Field	Data type	Sub-field	Data type	Field description	Data stored
id	number			Token ID	Numbers
slug	string			Unique NFT identifier	Numbers and characters
headline	string			Collection name	Characters
description	string			Collection description	Text
misc	object	coverPath	string	Cover file path	
		coverExtension	string	Cover extension	
		coverHash	string	Cover hash	
		audioCoverPath	string	Cover audio file path	
		videoCoverPath	string	Cover video file path	
cover	string			Collection cover	Base64 string (image coded)
status	string			Token status	'active' 'inactive' 'failed' 'banned'
isPrivate	boolean			Token privacy status	'true' 'false'
createdAt	date string			Date of creation	Timestamp
updatedAt	date string			Date of last update	Timestamp

Where is the data physically stored?

By default, files associated with NFTs or SFTs are stored on secure servers owned by Decimal, but users have the option to provide their own storage.

The storage has specific software and hardware requirements, so qualified users can organize their NFT/SFT storage through the API.

Ownership mechanism

In the request for NFT/SFT data through the SDK, a signature is formed (based on the user's mnemonic phrase), which is sent in the request to the gateway server. On the server, the signature is converted to a public address, then the Decimal address, which is used to check if NFT/SFT belongs to this address. Only if the check is successful the asset is returned. Otherwise, the asset is not returned.

Consider the example of calling the `getNft(<NftId>)` method used in the PHP SDK:

```
$result = $transaction->getNft('49501d55a30944bf7b3b72e618c1cc564cdeae');
```

If the user is the owner of the token, then he will receive the following response:

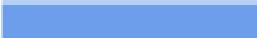
```
{
  asset: 'assets/JeqShgz5ySuYgrDr2f5Cz8MrKTSP2pyk_0299a.png',
  ...commonFields,
}
```

If the user is not the owner of the token, then he will receive the following response:

```
{
  asset: null,
  ...commonFields,
}
```

Thus, only users who currently own the token can access the file associated with the token.

23. Cross-chain swap



Cross-chain swap - a function of transferring a coin from the Decimal network to the Ethereum or Binance Smart Chain network, as well as in the opposite direction.

Possible transfer directions:

Decimal > Ethereum

Ethereum > Decimal

Decimal > BSC

BSC > Decimal

BSC > Ethereum

Ethereum > BSC

It is important to note that the initial blockchain in such operations is always Decimal. This means that it is impossible to introduce coins that did not previously exist in Decimal from other blockchains. The opposite is also true, if the coin was

created in Decimal, first sent to Ethereum and BSC, then it can then be sent back to the Decimal blockchain.

There is no exchange rate for this transaction. If you sent 10 DEL (Decimal), then you will receive 10 DEL (Ethereum) on the target blockchain.

Cross-chain swap fees

The cross-chain swap operation consists of two transactions - in the source and target blockchains. The sender pays the commission for both these transactions.

Cross-chain swap stages:

1. Initial blockchain transaction.

At this stage, the coin is sent to a special dedicated freezing address (if a native blockchain coin is sent), or to a smart contract address (if tokens are sent). Transaction costs are paid in the coin of the original blockchain by the sender.

If the source blockchain is Ethereum or BSC, then at this stage the smart contract burns acquired tokens to avoid coin duplication.

2. Target blockchain transaction setup.

At this stage, the information is formed by a transaction in the target blockchain.

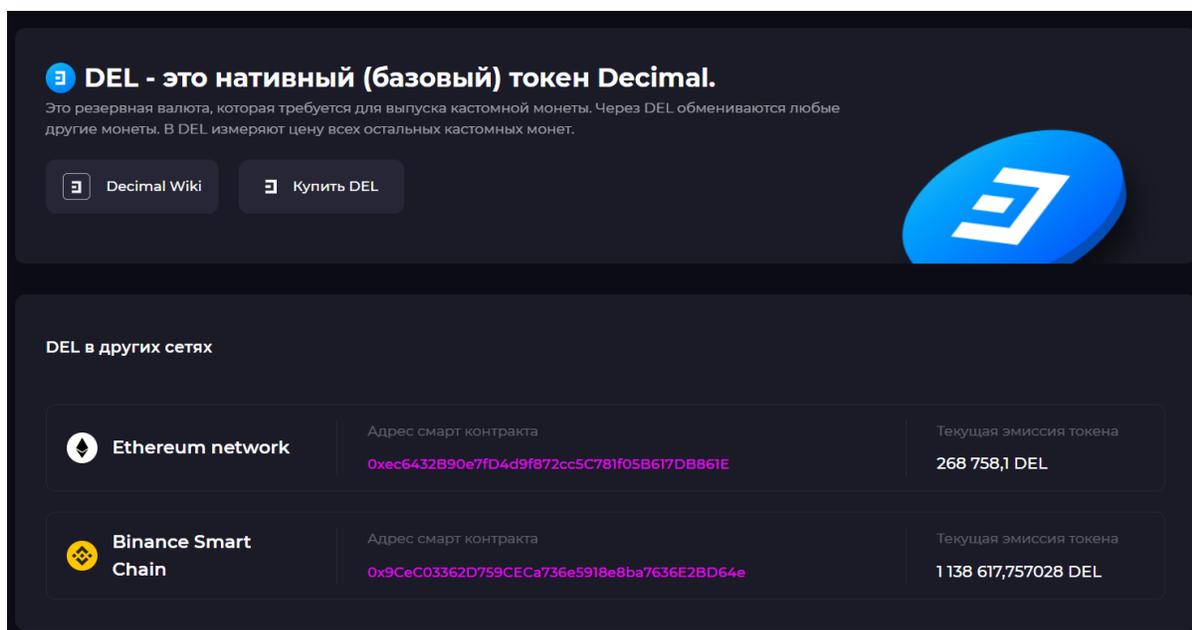
3. Cross-chain swap ending

A token is issued in the target blockchain, or a coin is unfrozen and credited to the recipient's address.

All steps are automatic. Including the creation of self-contracts for new coins in the Ethereum and BSC blockchains. At the same time, a link to the corresponding contracts appears on the page of the coin in the Explorer.

Example (DEL)

<https://explorer.decimalchain.com/coins/del>



DEL - это нативный (базовый) токен Decimal.
Это резервная валюта, которая требуется для выпуска кастомной монеты. Через DEL обмениваются любые другие монеты. В DEL измеряют цену всех остальных кастомных монет.

[Decimal Wiki](#) [Купить DEL](#)

DEL в других сетях

Сеть	Адрес смарт контракта	Текущая эмиссия токена
 Ethereum network	<code>0xec6432B90e7fD4d9f872cc5C781f05B617DB861E</code>	268 758,1 DEL
 Binance Smart Chain	<code>0x9CeC03362D759CECa736e5918e8ba7636E2BD64e</code>	1 138 617,757028 DEL

DEL is a Decimal Network native (base) token
This is the reserve currency that is required to issue a custom coin. Any other coins are exchanged through DEL. All custom coins price is represented in DEL.

[Decimal Wiki](#) [Buy DEL](#)



DEL in other networks

 Ethereum network	Smart contract address <code>0xec6432B90e7fD4d9f872cc5C781f05B617DB861E</code>	Current token volume 268,758.1 DEL
 Binance Smart Chain	Smart contract address <code>0x9CeC03362D759CECa736e5918e8ba7636E2BD64e</code>	Current token volume 1,138,617.757028 DEL

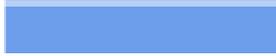
BSC Smart Contract

<https://bscscan.com/token/0x9CeC03362D759CECa736e5918e8ba7636E2BD64e>

Ethereum Smart Contract

<https://etherscan.io/token/0xec6432B90e7fD4d9f872cc5C781f05B617DB861E>

23. Help / FAQ



Our technicians have drawn up a technical description of Decimal's structure, technology and services. This makes it easier to understand internal processes, details of procedures and mechanisms embedded in the blockades. The list includes both the technical description, addressed to developers or business owners, and a wide range of ordinary users who are looking for answers to their questions and ways to resolve any difficulties in interacting with Decimal.

Click on the following link for more information

<https://help.decimalchain.com>

Articles and answers to questions will be regularly updated and enriched with new information as the community and the Decimal Blockade itself develop.

24. Block structure



Block is the fundamental essence of Blockchain. An element of a block chain that contains user transactions, DEL issuance transactions, validator signatures, commissions, hash of the current and previous block and other service information.

In the Decimal blockchain, blocks are generated approximately every 5.5 to 6 seconds.

The block contains between 0 and 10,000 transactions, each weighing approximately 180 bytes. The block is rewarded according to the [emission](#) model, at the start of 50 DEL with a subsequent increase every 432,000 blocks (~ 30 calendar days).

Blocks			
24,724	23 hours ago	135 txns in 5.74 sec.	50 tDEL
24,723	23 hours ago	135 txns in 5.58 sec.	50 tDEL
24,722	23 hours ago	134 txns in 5.70 sec.	50 tDEL
24,721	23 hours ago	120 txns in 5.68 sec.	50 tDEL
24,720	23 hours ago	15 txns in 5.82 sec.	50 tDEL
24,719	23 hours ago	135 txns in 5.62 sec.	50 tDEL
24,718	23 hours ago	135 txns in 5.64 sec.	50 tDEL
24,717	23 hours ago	135 txns in 5.85 sec.	50 tDEL
24,716	23 hours ago	135 txns in 5.59 sec.	50 tDEL

Figure 11 - Decimal blocks in the browser.

Decimal is based on the Tendermint engine. For more details about the unit design, see its specification (<https://github.com/tendermint/spec/blob/953523c3cb99fdb8c8f7a2d21e3a99094279e9de/spec/blockchain/blockchain.md>).

Here are some excerpts from the documentation.

24.1. Data structure

The block includes the following list of basic data types:

- Block

- Title (Header)
- Version
- Block Identifier (BlockID)
- Time
- Data (for transactions)
- Confirmations and votes (Commit and Vote)
- EvidenceData and Evidence

24.2. Block

A block consists of a header, transactions, votes and a list of evidence of violations (e.g. a double signature).

```
type Block struct {  
    Header Header  
    Txs     Data  
    Evidence EvidenceData  
    LastCommit Commit  
}
```

Note that **LastCommit** is a set of signatures of the validators that signed the last block.

24.3. Title

The block header includes metadata about the block and about the consensus, as well as data confirmation in the current block, the previous block and the result returned by the application:

```
type Header struct {  
// basic information about the unit  
  Version Version  
  ChainID string  
  Height int64  
  Time Time  
  
// information about the previous block  
  LastBlockID BlockID  
  
// block data hashes  
  LastCommitHash []byte // confirmations from  
validators from the previous block  
  DataHash []byte // Merkle tree of transaction hashes  
  
// application data hashes from the previous block  
  ValidatorsHash []byte // current unit validators
```

NextValidatorsHash []byte // validators of the following block

ConsensusHash []byte // consensus parameters for the current block

AppHash []byte // application status after transactions from the previous block

LastResultsHash []byte // root hash of all transaction data from the previous block

// information to achieve consensus

EvidenceHash []byte // conflicts in this block

ProposerAddress []byte // creator of this block

24.4. Transactions

Data is a wrapper of a list of transactions, which are byte arrays of any length:

```
type Data struct {  
    Txs [][]byte  
}
```

25. References



1. Jae Kwon, Tendermint: Consensus without Mining Draft v.0.6 (outdated), 2014.
<https://tendermint.com/static/docs/tendermint.pdf>
2. Practical Byzantine Fault Tolerance Miguel Castro and Barbara Liskov Laboratory for Computer Science, Massachusetts Institute of Technology, 545 Technology Square, Cambridge, MA02139
<http://www.pmg.csail.mit.edu/papers/osdi99.pdf>
3. Vitalik Buterin A Proof of Stake Design Philosophy
<https://medium.com/@VitalikButerin/a-proof-of-stake-design-philosophy-506585978d51>
4. Cosmos SDK
<https://docs.cosmos.network/>
5. Tendermint
<https://docs.tendermint.com/>

6. Hayek, Friedrich August von.

<https://ru.wikipedia.org/wiki/%D0%A5%D0%B0%D0%B9%D0%B5%D0%BA,%D0%A4%D1%80%D0%B8%D0%B4%D1%80%D0%B8%D1%85%D0%90%D0%B2%D0%B3%D1%83%D1%81%D1%82%D1%84%D0%BE%D0%BD>

7. Kaines, John Maynard.

<https://ru.wikipedia.org/wiki/%D0%9A%D0%B5%D0%B9%D0%BD%D1%81,%D0%94%D0%B6%D0%BE%D0%BD%D0%9C%D0%B5%D0%B9%D0%BD%D0%B0%D1%80%D0%B4>

8. BIP: 173 or bc1 addresses

<https://github.com/bitcoin/bips/blob/master/bip-0173.media/wiki>

9. Pieter Wuille lecture on new bech32 address format

https://www.reddit.com/r/Bitcoin/comments/62fydd/pieter_wuille_lecture_on_new_bech32_address_format/

10. Bech32, native SegWit address already used on the mainnet

https://www.reddit.com/r/Bitcoin/comments/74tonn/bech32_native_segwit_address_already_used_on_the/dqlogru/

11. Enterprise-class open source distributed monitoring solution.

<https://www.zabbix.com/ru/manuals>