International Science Journal of Engineering & Agriculture
2022; 1(3): 98-109
https://isg-journal.com/international-science-journal-of-engineering-agriculture/
doi: 10.46299/j.isjea.20220103.8
ISSN: 2720-6319

INTERNATIONAL SCIENCE GROUP

UML MODEL OF THE PROPERTY RIGHT DISTRIBUTION MODULE USING NFT FRACTIONALIZATION BASED ON BLOCKCHAIN TECHNOLOGY

Yehor Rudytsia[1], Nataliia Bogdanova[1]
[1]Department of Information Systems and Technologies, National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute»
ORCID 0000-0003-0797-7247
Email address: yehorrudytsia@gmail.com, n_bogdanova@ukr.net

To cite this article:
Yehor Rudytsia, Nataliia Bogdanova. UML model of the property right distribution module using nft fractionalization based on blockchain technology. International Science Journal of Engineering & Agriculture. Vol. 1, No. 3, 2022, pp. 98-109. doi: 10.46299/j.isjea.20220103.8.
Received: 07 17, 2022; **Accepted:** 07 24, 2022; **Published:** 08 01, 2022

Abstract: The article proposes UML diagrams of the property right distribution module using NFT fractalization based on blockchain technology, which allow to develop software for the implementation of the modern way of fractions generation according to the ERC-1155 standard. A blockchain is a distributed database that is shared among the nodes of a computer network. As a database, a blockchain stores information electronically in digital format. Blockchains are best known for their crucial role in cryptocurrency systems, such as Bitcoin, for maintaining a secure and decentralized record of transactions. The innovation with a blockchain is that it guarantees the fidelity and security of a record of data and generates trust without the need for a trusted third party. This standard was chosen for the impossibility of offering fractions for sale on other platforms, ensuring the uniqueness of the proposed decentralized application. The main reason for the development is to create a module located on the network blockchain, ensuring the security of functionality with cryptographic algorithms from one of the most secure blockchain — Ethereum. The task of development is to implement functionality that ensures the fulfillment of all functional requirements, the most important of which are the creation of a unique NFT to the user's property, its Fractionalization and the sale of fractions to other users.

Keywords: Blockchain, elliptical cryptography, unique token, Fractionalization, property right.

1. Introduction

Online commerce has come to rely almost exclusively on financial institutions known as trusted third parties to process electronic payments. While the system works reasonably well for most transactions, it still suffers from the inherent weaknesses of a trust-based model [1-6].

Completely irreversible transactions are not really possible because financial institutions cannot avoid mediation disputes. Intermediation costs increase transaction costs by limiting the minimum practical transaction size and cutting off the opportunity for small, random transactions [7].

These costs and the uncertainty of payment can be avoided by using physical currency, but until 2008 there was no mechanism for making payments over a communication channel without a trusted party.

The concept of a decentralized digital currency, as well as alternative applications such as property registries, has been around for decades. Anonymous electronic cash protocols of the 1980s and 1990s, mostly based on a cryptographic primitive known as Chaum blinding, provided a currency with a high degree of privacy, but the protocols largely fell out of favor due to their reliance on a centralized intermediary. In 2005, Hal Finney introduced the concept of "reusable proofs of work", a system that uses ideas from b-money along with Adam Beck's computationally complex Hashcash puzzles to create a cryptocurrency concept, but again fell short. relying on reliable computing as a backend. In 2009, Satoshi Nakamoto first put decentralized currency into practice by combining established primitives for managing ownership using public-key cryptography with a consensus algorithm for tracking who owns coins, known as "proof of work."

From a technical point of view, the ledger of a cryptocurrency such as Bitcoin can be thought of as a state transition system, where there is a "state" consisting of the ownership status of all existing bitcoins and a "state transition function" that takes a state and a transaction and outputs a new state , which is the result. In a standard banking system, for example, a state is a balance, a transaction is a request to move $X from A to B, and a state transition function decreases the value of A's account by $X and increases the value of B's account by $X [8].

Cryptocurrencies are playing an increasingly important role in our society. ERC-20 fungible tokens [9] and ERC-721 non-fungible or unique tokens (NFTs) [10] are becoming more common. In this work, the attention is focused on irreplaceable tokens and their use for digitization of entities (assets).

2. Current state of the problem and ways to solve it

Currently there are not a lot of fractionalization decentralized protocols. Some of these protocols are Fractional Art and Unicly.

Fractional Art is a decentralized protocol that provides collective ownership and management of one or more NFTs [11]. The Fractional platform enjoys notable support from crypto venture capitalists. After raising $500,000 in a pre-seed round led by Robot

Ventures in March 2021, the platform raised an additional $7.9 million in a seed round in early August led by Paradigm.

Advantages of the Fractional Art platform:

- decentralized protocol;
- it is possible to become a partial owner of NFT, when buying a whole unique token does not have enough cryptocurrency;
- you can decide what to do with your fractions (for example, add the parent NFT to the digital gallery);
- purchase a faction by choosing from a variety of iconic digital art collections consisting of one or more NFTs.

Disadvantages of the Fractional Art platform:

- NFTs represent only works of art.

Unicly is a decentralized protocol that enables collective ownership and management of NFT collections [12] and was developed to address some of the problems identified in previous NFT fractionalization attempts. First, the problem of being able to crush only individual NFTs at a given time was a major limiting factor for previous protocols. An important advantage of this platform over its counterparts is the possibility of Fractionalization not only of individual tokens, but also of entire collections. This capability allows fractional holders to represent a more diversified set of markets and positions them for significantly greater returns than what can be achieved from holding shares of a single token.

Advantages of the Unicly platform:

- decentralized protocol;
- possibility of Fractionalization of entire collections;
- the possibility of creating fractions in the form of standard ERC-1155 tokens, unlike ERC-721 [13, 14].

Disadvantages of the Unicly platform:

- NFTs represent only works of art.

Today, most of the existing tokenization protocols have one significant problem – the impossibility of distributing the rights to the ownership of one or another token. The solution proposed in this work solves the above-described problem of distributing the rights to the property of a physical or electronic object by the method of NFT (Non-fungible token) Fractionalization into a constant number of fractions.

3. Main part

3.1. Description of the activity process

The process of user activity begins with the creation of a unique token (NFT), indicating the corresponding Internet link, which is either an electronic property (for example, a picture) or a legal confirmation of the right to property (relating to non-electronic properties).

After creating the token, the user can put his NFT for sale. That is, NFT that has not yet been fractionalized can also be purchased by paying the amount of cryptocurrency specified by the token owner when creating it. Next, the user can fractionalize his Token. Fractionalization automatically cancels the offer to purchase the entire NFT that was created. The process of Fractionalization consists in dividing a whole token into a constant number of parts, or fractions. When splited into fractions, a new token is created on the blockchain network. This token is of the ERC-1155 [15, 16] standard. That is, a new smart contract with its reading and writing methods is published on the blockchain network. The ERC-1155 standard is interchangeable. Fractions are represented by tokens, each of which is interchangeable. After the Fractionalization process, an offer for the sale of all fractions (FNFT) is created - the address of the cryptocurrency and its amount are indicated. The purchase of fractions consists in sending the set amount of the corresponding cryptocurrency to the address of the smart contract on the blockchain on which the fractions are stored. The cryptocurrency sent from the smart contract is sent to the wallet of the user who owned the fractions. The offer for sale is created precisely from the address of the owner of the fractions.

Thus, it is possible to buy not the whole token (NFT), but only part of the fractions (FNFT).

The user can offer fractions for sale and buy fractions of other users. If there is no need for the further sale of his fractions, the user can cancel his offer.

3.2. Input data for the proposed system

The inputs are information about the entity, the unique ownership token the user intends to create, and the information needed to put the newly created token up for sale.

This information includes the address of the cryptocurrency for which the user puts their newly created unique token for sale, as well as the amount of that cryptocurrency. Since the software was developed in the typed programming language “Solidity”, each input parameter will have a data type specified.

3 types of data are used in the input data to the main functions of the software whicj are "creating a token", "putting a token for sale", "token fractionalization", "putting fractions for sale": uint, address, string.

Address (solidity) is a unique sequence of numbers and letters that is very similar to an email address. It refers to a specific destination on the network where cryptocurrency can be sent. The idea is to generate a unique address for a person every time he or she needs to receive cryptocurrency.

The input data for creating a unique token are:

- blockchain address of the owner of the unique token being created. Data type: address;
- URI (unique resource identifier) of the token (Internet link to approved ownership). Data type: string;
- blockchain address of the cryptocurrency, for which the newly created token can be purchased. Data type: address. If a null address is specified, the cryptocurrency of purchase will be set to the native Ether currency;

- denomination of the token for which the newly created token can be purchased. Data type: uint.

Input data for unique token`s fractionalization are:

- ID of the unique token that will be fractionated. Data type: uint;
- the name of the fractional token. Data type: string;
- fractional token symbol. Data type: string;
- the number of factions (parts) into which the right to own the entity will be divided. Data type: uint;
- blockchain address of the cryptocurrency, for which the newly created fractions can be purchased. Data type: address. If a null address is specified, the purchase cryptocurrency will be set to the native Ethereum currency of the Ether network;
- how much will one fraction (the smallest piece of property rights) cost. Data type: uint.

3.3. Output data

The output data of the described functions are:

- unique token;
- an offer to purchase a unique token was created;
- created factions (a new type of token that has its own address in the Ethereum network blockchain);
- an offer to purchase a unique token has been created

3.4. The structure of arrays of information

For the most part, data on the blockchain is stored in the "mapping" data type. It is represented by a "key" and a "value" to which the "key" maps.

Fractional NFT smart contract:

When a unique token (NFT) is fractionated, the information is stored in the "FNFT Mapping", where the unique token number maps to a data structure that stores:

- fractional token on the blockchain;
- the number of factions.

3.5. Description of the functional model

The specification of the functional behavior of the system is presented in the form of a diagram of use cases. The structural diagram of the use cases is given below. The diagram shows all the functions of the system and describes the actors who will use them. Only one actor will interact with the system: the user is a person who can work with the main functions of the system. The use case diagram is in Fig. 1.

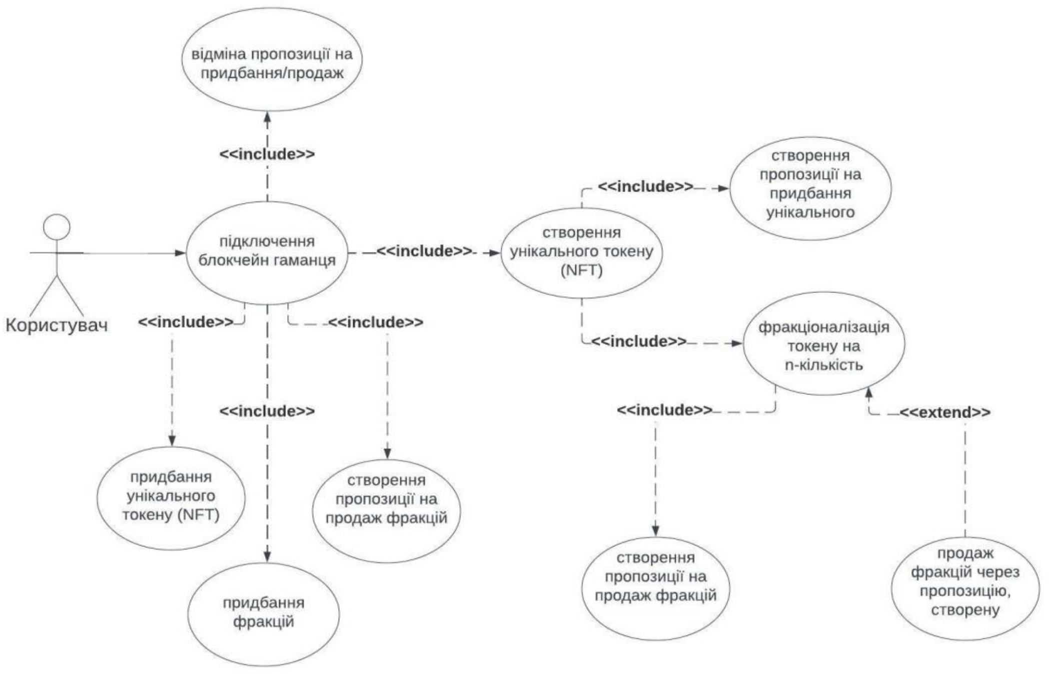
відміна пропозиції на придбання/продаж
<<include>>
<<include>>
створення пропозиції на придбання унікального
підключення блокчейн гаманця
<<include>>
створення унікального токену (NFT)
Користувач
<<include>>
<<include>>
<<include>>
фракціоналізація токену на n-кількість
<<include>>
<<include>>
<<extend>>
придбання унікального токену (NFT)
створення пропозиції на продаж фракцій
придбання фракцій
створення пропозиції на продаж фракцій
продаж фракцій через пропозицію, створену

Figure 1. Model of use cases.

According to the defined use cases, functional requirements were identified and their priority was established. Functional requirements are given in table 1.

Table 1.
Functional requirements of the module for the distribution of ownership rights using fractalization of NFTs based on blockchain technology

Use case	Functional requirement	Priority
Connection of the a blockchain wallet	The system provides an opportunity to connect the blockchain wallet of the user	High
Creation of a unique token (NFT)	The system provides an opportunity to create a unique token (NFT) tied to an entity that the user owns	High
Creation of an offer to purchase a unique token	The system provides an opportunity to create an offer to purchase a unique token	High
Fractionalization of the token into n-number of parts (fractions)	The system provides an opportunity to fractionalize the token into n-number of parts (fractions)	High
Creation of an offer for the sale of fractions	The system provides an opportunity to create offers for the sale of fractions	High
1	2	3

Continuation of Table 1.

Selling of the fractions through an offer created by another user	The system provides an opportunity to sell fractions through an offer created by another user	Medium
Creation of an offer for the sale of fractions	The system provides an opportunity to create an offer for the sale of fractions	Medium
Purchase of Unique Token (NFT)	The system provides an opportunity to purchase a unique token (NFT)	Medium
Purchase of fractions	The system provides an opportunity to purchase fractions	Medium
Canceling of the offer to buy/sell	The system provides an opportunity to cancel your purchase and sale offer	Medium
Ability to view created purchase and sale offers	The system provides an opportunity to view the created offers for purchase and sale	Medium
1	2	3

A sequence diagram describes all possible use cases for the software.

3.6. Software architecture

Fig. 2 shows a diagram of classes (smart contracts):

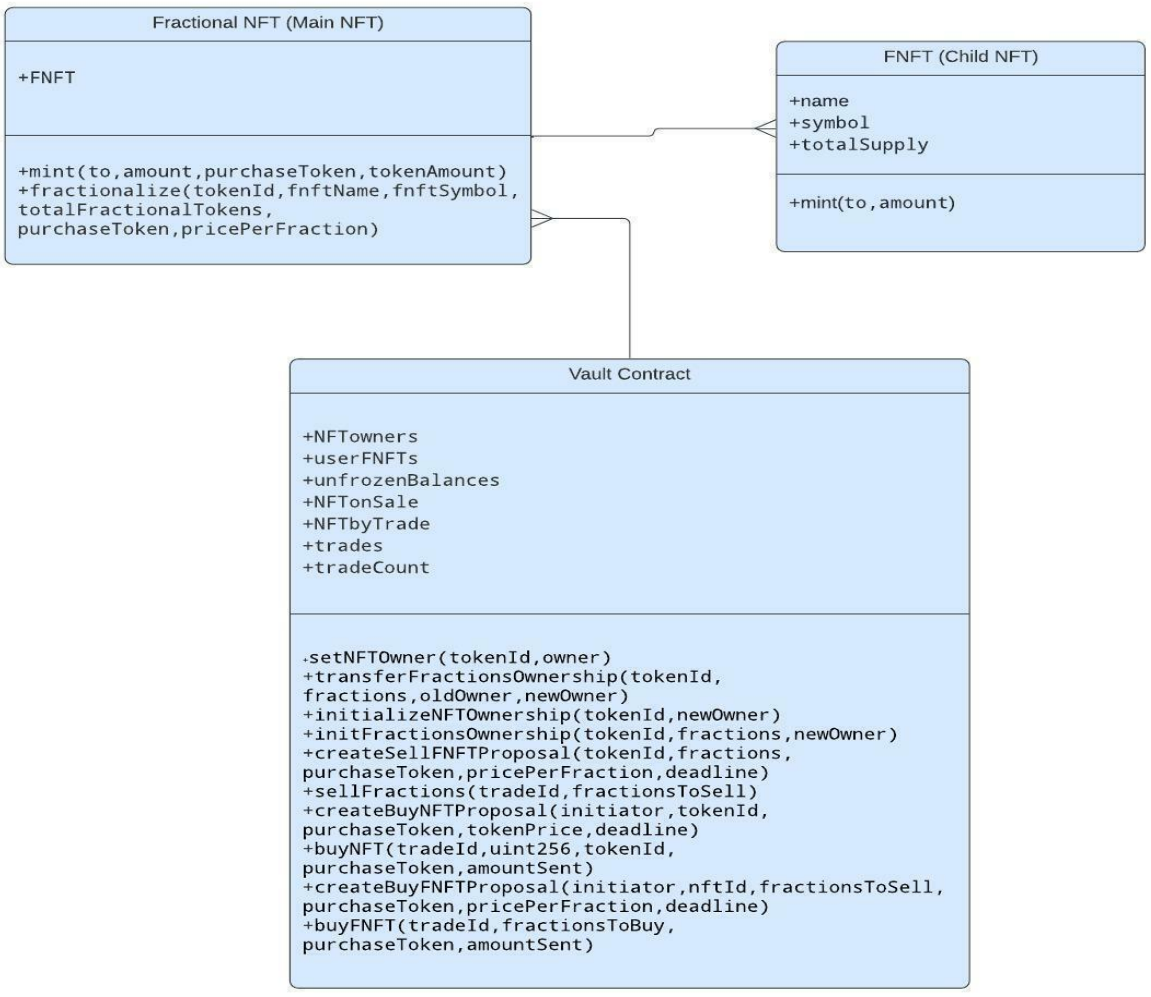
Fractional NFT (Main NFT)
+FNFT
+mint(to,amount,purchaseToken,tokenAmount)
+fractionalize(tokenId,fnftName,fnftSymbol,
totalFractionalTokens,
purchaseToken,pricePerFraction)
FNFT (Child NFT)
+name
+symbol
+totalSupply
+mint(to,amount)
Vault Contract
+NFTowners
+userFNFTs
+unfrozenBalances
+NFTonSale
+NFTbyTrade
+trades
+tradeCount
+setNFTOwner(tokenId,owner)
+transferFractionsOwnership(tokenId,
fractions,oldOwner,newOwner)
+initializeNFTOwnership(tokenId,newOwner)
+initFractionsOwnership(tokenId,fractions,newOwner)
+createSellFNFTProposal(tokenId,fractions,
purchaseToken,pricePerFraction,deadline)
+sellFractions(tradeId,fractionsToSell)
+createBuyNFTProposal(initiator,tokenId,
purchaseToken,tokenPrice,deadline)
+buyNFT(tradeId,uint256,tokenId,
purchaseToken,amountSent)
+createBuyFNFTProposal(initiator,nftId,fractionsToSell,
purchaseToken,pricePerFraction,deadline)
+buyFNFT(tradeId,fractionsToBuy,
purchaseToken,amountSent)

Figure 2. Diagram of classes (smart contracts).

Fractional NFT smart contract: a contract-factory of unique tokens that are tied to the entity being digitized. Performs the functions of creating a unique token (NFT) and its fractionalization. The sharding data is recorded in the Mapping, which stores the address of the sharded token in the blockchain and the number of shards.

FNFT smart contract: a contract-factory of fractionated tokens, i.e. those parts (fractions) into which ownership rights of a unique token are divided after fractionalization.

Vault smart contract: Performs token storage functions, P2P offers are managed through this contract. Users can put NFTs up for sale by specifying the cryptocurrency and its quantity for which the token can be purchased. When putting FNFT fractions up for sale, you must specify the number of fractions that will be put up in the purchase offer. Through the Vault smart contract, a user can purchase NFTs or FNFTs by accepting one of the offers or by creating their own. When creating your own offer, funds for fractions are temporarily blocked in the smart contract.

3.7. Function specification

Fractional NFT smart contract:

Address on the blockchain (Ropsten Ethereum): 0x151f3d432041672AE17cdc767D1d2E979c8Dd656.

Methods that change state on the blockchain (Write methods):

- mint(address to, tokenURI_ memory string, purchaseToken address, uint256 tokenPrice). Description: A new unique token (NFT) is created for some entity that being identified by the URI;
- fractionalize(uint256 tokenId, string memory fnftName, string memory fnftSymbol, uint _totalFractionalTokens, address purchaseToken, uint pricePerFraction). Description: Fractions the token into the specified number of fractions. Can only be the owner of a unique token;
- approve(address, uint256 tokenId). Description: An address that validates this method allows the specified number of tokens of its own tokens to be used for a specified other address;
- record (uint256 tokenId). Description: Sends the given token number to the null address. After this operation, the marker with the specified number is blocked forever;
- burnFractionsSupply(uint256 tokenId). Description: Sends all fractions (FNFT) according to the given parent token number (FNFT) to the null address. After this operation, all factions with the given parent token number are blocked forever. This method can only be used by a user who owns all fractions of a certain NFT;
- pause(). Description: Suspends the operation of the smart contract. After calling this method, all operations with the smart contract remain temporarily unavailable;
- unpause(). Description: Restarts the smart contract. After calling this method, all operations with the smart contract become available;
- transferFrom(address from, address to, uint256 tokenId). Description: Sends a specific NFT from a "from" address to a "to" address;
- safeTransferFrom(address from, address to, uint256 tokenId). Description: Sends a specific NFT from a "from" address to a "to" address. Unlike the above-described method, this method, before transferring the token, performs an additional check of the address to which the token should be sent. Checks are made with the goal of not losing the token forever in case the address to which the token is to be sent is not able to process the ERC-721 standard;
- TransferOwnership (address newOwner). Description: Protect yourself from transferring ownership of a smart contract to another address. This method can only be used from the address of the current smart contract holder.

Methods that do not change state on the blockchain (Read methods):

FNFT(uint256 tokenId). Description: Zoom in to get the smart address

of the FNFT contract and the number of fractions that were created during the NFT fractionation process;

- VaultContract(). Description: returns the Vault smart contract address;
- balanceOf(address owner). Description: returns the number of NFTs located at the given address;

- isApprovedForAll(address owner, address operator). Description: Returns "true" if the operator has permission to shuffle all tokens of the "owner" address.

VaultContract smart contract:

Address on the blockchain (Ropsten Ethereum): 0x33a6C0C8dd5E5fe09cAd2E472c62338d847980f8.

Methods that change the state on the blockchain (Write methods):

- transferFractionsOwnership(uint256 tokenId, uint256 fractions, address oldOwner, address newOwner). Description: Transfers ownership of a given number of factions to a new owner (new address). It can be called only from the address whose account has at least as many fractions as specified in the function argument;

- initializeNFTOwnership(uint256 tokenId, address newOwner). Description: Initializes ownership of a unique token. When it is created, the method is called, passing as arguments the address that called the method;

- initFractionsOwnership(uint256 tokenId, uint256 fractions, address newOwner). Description: Initializes ownership of a specified number of factions. When it is created, the method is called, passing as arguments the address that called the method;

- createSellFNFTProposal(uint256 tokenId, uint256 fractionsToBuy, address purchaseToken, uint256 pricePerFraction, uint256 deadline). Description: Creates an offer to sell your factions (a certain parent token) to other users. The cryptocurrency for the future transaction is sent to the smart contract and remains blocked on the smart contract either until the transaction happens or until its deadline;

- sellFractions(uint256 _tradeId, uint256 fractionsToSell). Description: Sale of fractions according to the P2P agreement that was selected;

- createBuyNFTProposal(address _initiator, uint256 tokenId, address _purchaseToken, uint256 _tokenPrice, uint256 deadline). Description: Creates an offer to purchase a unique NFT token. Only the owner of the token can call this method. The method must specify the cryptocurrency for which the token can be purchased, the amount of cryptocurrency, and the deadline of the agreement, after which it becomes invalid;

- buyNFT(uint256 _tradeId, uint256 _tokenId, address _purchaseToken, uint256 amountSent). Description: Sale of a unique token (NFT) under the selected P2P agreement;

- function createBuyFNFTProposal (address _initiator, uint256 nftId, uint256

_fractionsToSell, address _purchaseToken, uint256 _pricePerFraction, uint256 deadline). Description: Creates an offer to purchase FNFT factions. Only the faction owner can call this method. The method must specify the cryptocurrency for which the token can be purchased fractions, the amount of cryptocurrency, and the deadline of the agreement, after which it becomes invalid;

- buyFNFT(uint256 _tradeId, uint256 _fractionsToBuy, address _purchaseToken, uint256 _amountSent). Description: Sale of fractions (FNFT) under the selected P2P agreement;

- cancelTrade(uint256 tradeId). Description: Cancels the displayed agreement.

Methods that do not change the state on the blockchain (Read methods):

- balanceOf(uint256 nftId, address owner). Description: The current number of fractions, of a certain parent token, that the user owns;
- availableForTransfer(uint256 nftId, address owner). Description: The current number of fractions not blocked in trade (offers) of a certain parent token owned by the user;
- ownerOfNFT(uint256 tokenId). Description: Returns the address of the owner of the specified ID unique token (NFT);
- isNFTonSale(uint256 tokenId). Description: Returns "true" if the given unique token is for sale;
- getTradeId(uint256 tokenId). Description: Returns the trade number in which the given token is involved. If the token is not for sale, it returns 0;
- onERC721Received(address operator, address from, uint256 tokenId, bytes data). Description: This method is called from a contract that calls the "safeTransferFrom" method with the intention of making a token transfer to the given address. This method returns a signature indicating that this smart contract is configured to work with ERC-721 standard tokens. If this function is not available, a transaction error will occur when the transfer is attempted;
- onERC1155Received(address operator, address from, uint256 value uint256 tokenId, bytes data). Description: This method is called from a contract that calls the "safeTransferFrom" method with the intention of making a token transfer to the given address. This method returns a signature that indicates that this smart contract is configured and provides for work with tokens of the ERC-1155 standard; If this function is not available, a transaction error will occur when the transfer is attempted.

Smart contracts developed on the basis of designed UML diagrams represent a module that provides use cases described at the design stage.

4. Conclusions

The article proposed a UML model of the module of property right distribution by fractionalizing a unique NFT token into a constant number of fractions. This system allows users to share the right to property due to tokenization on top of the blockchain network.

The proposed solution is to divide a unique token into a constant number of child tokens (fractions) using the fractionalization method, thus adding an opportunity to share ownership rights to more than one person (an address in the blockchain network).

After analogues`s analysis, it was decided to use the ERC-1155 standard token as fractions to prevent the resale of fractions on other decentralized platforms.

A solution was implemented using modern standards in the field of cryptography. The security of the software product is based on such a section of asymmetric encryption as elliptic cryptography.

References:

1) Nakamoto, S., & Bitcoin, A. (2008). A peer-to-peer electronic cash system. Bitcoin.–URL: https://bitcoin. org/bitcoin. (date of access: 1.01.2022).
2) Buterin, Vitalik. "Ethereum: A next-generation smart contract and decentralized application platform." (2014). URL: https://fermatslibrary.com/s/ethereum-a-next-generation-smart-contract-and-decentralized-application-platform (date of access: 14.03.2022).
3) ERC-20: Token Standard. URL: https://eips.ethereum.org/EIPS/eip-20 (date of access: 5.01.2022).
4) ERC-721: Non-Fungible Token Standard. URL: https://eips.ethereum.org/EIPS/eip-721 (date of access: 22.01.2022).
5) Fractional Art. URL: https://fractional.art/ (date of access: 10.02.2022).
6) Unicly. URL: https://www.app.unic.ly/#/ (date of access: 11.02.2022).
7) Binance Academy. URL: https://academy.binance.com/en (date of access: 8.04.2022).
8) ERC-1155: Multi Token Standard. URL: https://EIPs.ethereum.org/EIPS/EIP-1155 (date of access: 10.05.2022).
9) Non-fungible Token (NFT). URL: https://academy.binance.com/en (date of access: 23.05.2022).
10) An Introduction to NFT Fractionalization. URL: https://www.nfttech.com/insights/an-introduction-to-nft-fractionalization (date of access: 10.02.2022).