ISSN 2224-087X. Електроніка та інформаційні технології. 2023. Випуск 22. С. 69–78
Electronics and information technologies. 2023. Issue 22. P. 69–78

УДК 004 DOI: https://doi.org/10.30970/eli.22.7

ПІДХОДИ В МОДЕЛЮВАННІ СМАРТ-КОНТРАКТІВ НА ОСНОВІ ETHEREUM

Ю. Цудзенко

Львівський національний університет імені Івана Франка,
вул. Драгоманова 50, 79005 Львів, Україна
yura9989@gmail.com

Смарт-контракт являється одним із видів цифрової угоди між декількома сторонами, без посередників. Для реалізації смарт-контрактів розроблена ціла низка технологій та інструментів. У статті наведено та досліджено метод створення смарт-контрактів у середовищі Ethereum за допомогою мови програмування Solidity, а також середовища програмування Visual Studio Code. Проведено аналіз часових трендів смарт-контракту Ethereum використовуючи мову програмування R та середовище розробки R-Studio.

Ключові слова: смарт-контракт, блокчейн, Ethereum, біткоїн.

1. Вступ

Блокчейн (англ. blockchain ланцюг блоків) - це відносно нова технологія, що являє собою впорядкований ланцюжок даних, які є блоками. Блокчейн є простою і водночас добре продуманою технологією зберігання полів даних. Зазвичай блокчейн використовується для запису транзакцій, відстеження активів і встановлення довіри між учасниками віртуального цифрового договору. Завдяки застосуванню біткоїнів і криптовалют блокчейн досягнув популярності. Блокчейн зараз використовується майже в усіх сферах, включаючи медицину, логістику, системи захисту, азартні ігри, створення токенів. Укладання договорів між двома сторонами є найбільш доцільним застосуванням блокчейну. Розроблено сотні блокчейнів із власними специфікаціями та додатками, зокрема найпопулярнішими із них являється Bitcoin, Litecoin, Monero.

На рис. 1 показано схематичне зображення блокчейну.

Кожен блок містить власний унікальний хеш, який геренерується на основі даних. Ці дані зберігаються в блоку за алгоритмом шифрування SHA-256, що являється унікальним цифровим підписом. Кожен блок містить набір транзакцій, які створені та зберігаються в системі. Крім того, кожен блок містить мітку часу, посилання на попередній блок який ідентифікується за його хеш-значенням [1,2]. Таким чином дані зашифровуються і складаються в ланцюжок, завдяки цьому блокчейн отримав свою назву. Блокчейн можливо реалізувати будь-якою мовою програмування. Для демонстрації роботи блокчейну, використовуючи мову програмування JavaScript та середовище розробки VS Code, реалізовано блокчейн. Декілька унікальних даних, зокрема власні ініціали і дату створення збережено у блоку блокчейна. Результат збереження даних у блокчейні видно на рис. 2, також видно, що кожен блок має свій унікальний хеш(hash) по якому його можна ідентифікувати і посилання на попередній блок(precedingHash).

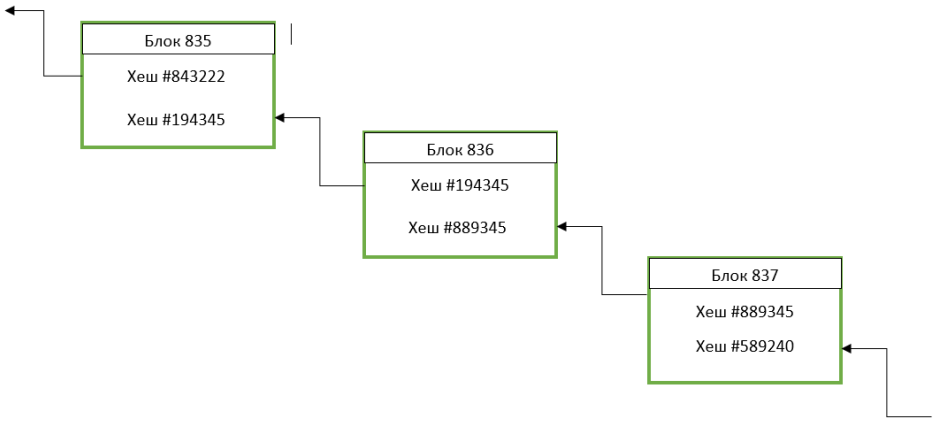


Рис. 1. Схематичне зображення блокчейну

```
CryptoBlockchain {
  blockchain: [
    CryptoBlock {
      index: 0,
      timestamp: '01/01/2020',
      data: 'Initial Block in the Chain',
      precedingHash: '0',
      hash: 'a12fb633f9f8f8e163f9a98e13eb2d47e81d5fc9c2c88d98893263bc38865389'
    },
    CryptoBlock {
      index: 1,
      timestamp: '01/02/2023',
      precedingHash: 'a12fb633f9f8f8e163f9a98e13eb2d47e81d5fc9c2c88d98893263bc38865389',
      hash: '34d99dd9a39a53c4032f9bbc181afcb6d75f1b994c3a2338b65488cdf8a678d9'
    },
    CryptoBlock {
      index: 2,
      timestamp: '02/02/2023',
      precedingHash: '34d99dd9a39a53c4032f9bbc181afcb6d75f1b994c3a2338b65488cdf8a678d9',
      hash: '77ea364e0159e9b481e77e9e2b06c03ff34b5fffd1178ceaa31a41036812aa64'
    }
  ]
}
```

Рис. 2. Результат виконання блокчейну

Смарт-контракт – це частина передової технології, яка може використовуватися в екосистемі блокчейну для механічного узгодження, виконання та забезпечення виконання умов юридично обов'язкової угоди[3]. Порівняно зі звичайними контрактами, смарт-контракти пропонують переваги зниження ризику транзакцій, зменшення витрат на адміністрування, обслуговування та підвищення ефективності корпоративних процесів, оскільки вони часто розміщуються на блокчейні та захищені ним [4].

2. Огляд інструментів

Ethereum - це криптовалюта, а також платформа, що дає змогу створювати децентралізованих сервісів та працює на основі смарт-контрактів [5]. Однією із найрозповсюдженіших мов програмування для створення смарт-контрактів є Solidity. Solidity – це контрактно орієнтована мова програмування високого рівня розроблена у

2014 році, що працює на основі віртуальної машини Ethereum(EVM) [5] із синтаксисом подібним до JavaScript, для розробки смарт-контрактів має статистичну типізацію, підтримує наслідування, включає цілу низку допоміжних бібліотек.

Truffle – це популярна платформа для розробки блокчейн-додатків на основі Ethereum, також використовується для оптимізації процесу розробки. Truffle надає набір інструментів і утиліт для створення, тестування та розгортання смарт-контрактів.[6]

VS Code – це засіб для створення та редагування веб-застосунків, програм.

MetaMask – це онлайн криптовалютний гаманець, що підключається до блокчейну Ethereum, взаємодіє з його екосистемою, та дозволяє не завантажувати весь блокчейн на свій пристрій[7].

На рис. 3 зображено схематичну взаємодію сервісів один між одним.

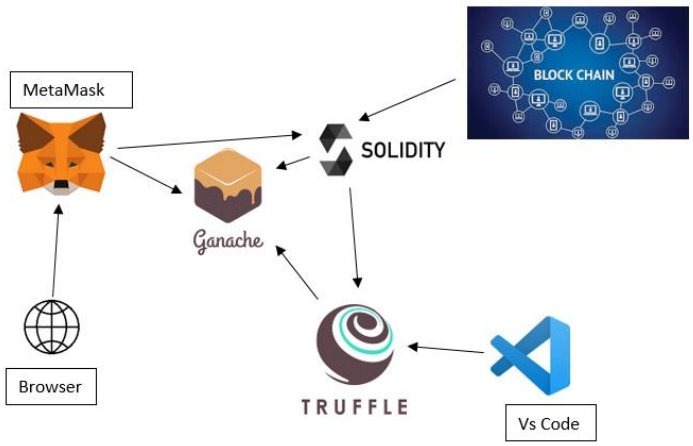


Рис. 3. Схема взаємодії сервісів для створення смарт-контракту

Для наглядного прикладу реалізовано смарт-контракт для продажу квитків, за допомогою мови програмування Solidity. Необхідною умовою для створення смарт-контракту є відповідна версія Solidity. Як було написано раніше синтакс Solidity є аналогічним об’єктно орієнтованим мовам програмування. На рис. 4 продемонстровано лістинг коду реалізації смарт-контракту продажу квитків, який передбачає, що користувач маючи певну кількість квитків, має можливість продати його за вказану суму у криптовалюті.

```
pragma solidity >=0.4.22 <0.9.0;

uint256 constant TICKET_AMOUNT = 10;

contract Tickets {
  address public seller = msg.sender;

  struct Ticket {
    uint256 price;
    address seller;
  }

  Ticket[TICKET_AMOUNT] public tickets;

  constructor() {
    for (uint256 i = 0; i < TICKET_AMOUNT; i++) {
      tickets[i].price = 1e17; // 0.1 ETH
      tickets[i].seller = address(0x0);
    }
  }

  function buyTicket(uint256 _index) external payable {
    require(_index < TICKET_AMOUNT && _index >= 0);
    require(tickets[_index].seller == address(0x0));
    require(msg.value >= tickets[_index].price);
    tickets[_index].seller = msg.sender;
  }
}
```

Рис. 4 Лістинг коду реалізація смарт контракту квитків

Solidity має спеціальні змінні (msg, block, tx) які завжди існують у глобальному просторі імен і містять властивості отримати доступ до інформації про виклик транзакції та блокчейн. Ці змінні дають змогу робити пошук адреси джерела, кількості ефіру та надісланих даних разом з виконаною транзакцією. Виконання транзакції робиться за допомогою міграції. Міграції слугують для відслідковування адрес у блокчейні, реалізацію міграції зображено на рис. 5.

migrations > 2_tickets.js > ...

```
const Tickets = artifacts.require("Tickets");

module.exports = function (deployer) {
  deployer.deploy(Tickets);
};
```

Рис. 5 Лістинг міграції смарт-контракту

Для запуску смарт-контракту необхідно створити сервер, який є віртуальним середовищем для запуску Ethereum, вказавши порт на якому буде прослуховуватись Ethereum. Genache прив’язується до сервера запуску смарт-контракту Ethereum, він дає змогу детальніше проаналізувати виконані транзакції, блоки блокчейну, що знаходяться у Ethereum. На рис.6 зображено баланс гаманців до виконання транзакцій, на якому видно що всі мають по 100 eth.

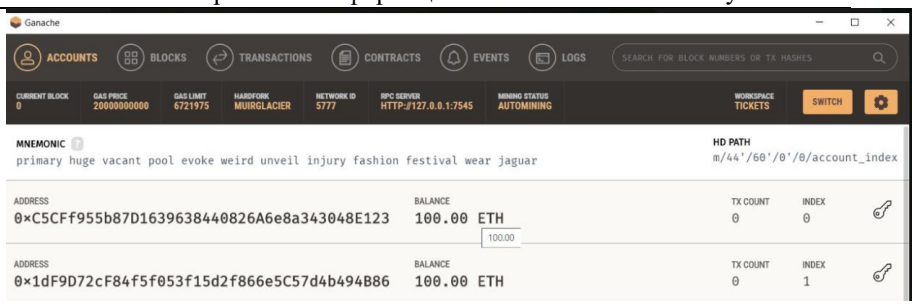


Рис. 6. Баланс до виконання транзакцій

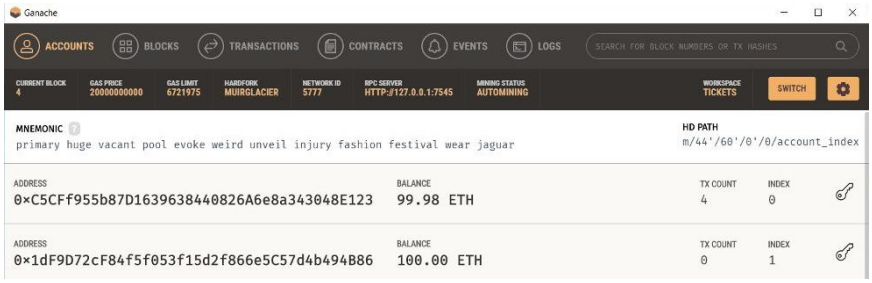


Рис.7. Баланс після виконання транзакції

На рис. 7 зображено результат після виконання транзакції між двома гаманцями, при якому було продано 10 квитків. Виконання транзакції між 2 гаманцями зайняло 650 мілісекунд, що становить 0.65 секунди. Після виконання транзакції її видно у вкладці Transaction, спочатку транзакція ініціалізується, після цього відбувається транзакція. Кожна виконана транзакція, що зображено на рис.8 має унікальний ідентифікатор(хеш транзакції), по якому можна ідентифікувати з якого гаманця було здійснено транзакцію[8].

На рис. 9 зображено створені блоки блокчейну, які містять інформацію про дату створення, та gas параметр, який є одиницею для вимірювання об’єму зусиль, що необхідні для виконання операції у мережі Ethereum, як видно кожен з блоків має свою унікальну кількість gas необхідних для виконання [9-10]. Необхідною умовою для успішності проведення смарт-контракту є добре написані тести, що дадуть змогу швидко перевірити написаний контракт і уникнути помилки у майбутньому. Провівши кілька дослідів з різними параметрами виявилось, що на виконання одного смарт-контракту в середньому витрачається до 1 секунди часу, в залежності від переданих параметрів.

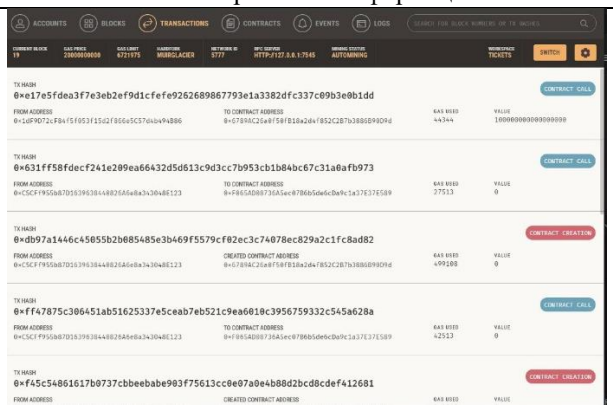


Рис.8. Список виконаних транзакцій

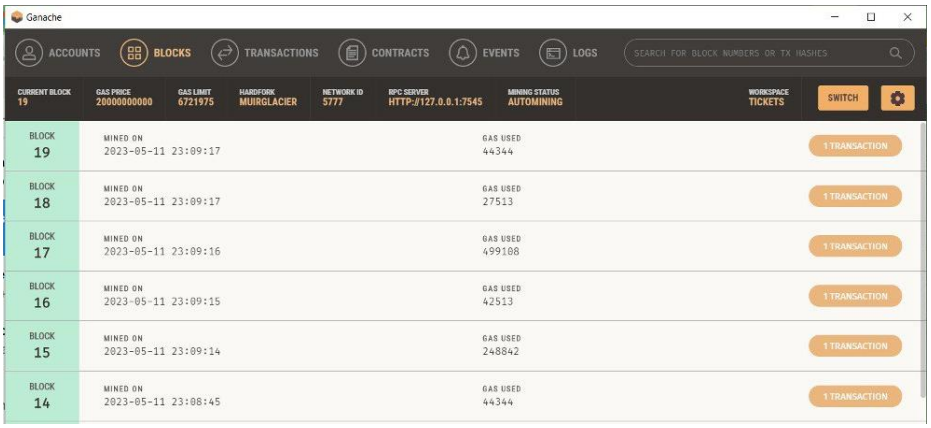
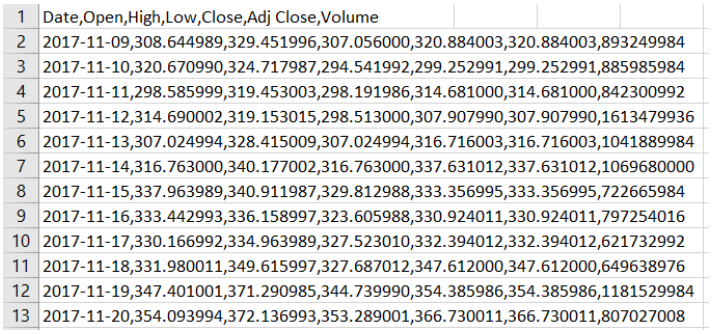


Рис. 9. Список створених блоків

Для більш надійнішого виконання смарт-контракту на основі Solidity надійним рішенням є використання методу онлайн перевірки після його розгортання. Перевірка полягає у виявлені і забезпеченості правильності виконання смарт-контракту. Проте використання такого методу вимагатиме запит до користувачів, чи погодяться вони із змінами у блокчейні, а це у свою чергу матиме вплив на всю мережу Ethereum.

3. Часові тренди смарт-контракту Ethereum

Даними для дослідження часових трендів смарт-контрактів на основі Ethereum використано із відкритого порталу набору даних Kaggle [11]. Набір даних представляє собою вартість смарт-контракту Ethereum починаючи з 9 листопада 2017 року по 25 березня 2022 року на міжнародній фінансовій біржі. База даних містить 1599 рядків даних та 8 унікальних стовпців значень смарт-контракту Ethereum. Базу даних збережено у форматі csv, що є форматом для представлення табличних даних і щоб його простіше було використовувати для аналізу даних. На рис. 10 продемонстровано як виглядає список збережених даних смарт-контракту Ethereum у форматі csv.



1	Date,Open,High,Low,Close,Adj Close,Volume
2	2017-11-09,308.644989,329.451996,307.056000,320.884003,320.884003,893249984
3	2017-11-10,320.670990,324.717987,294.541992,299.252991,299.252991,885985984
4	2017-11-11,298.585999,319.453003,298.191986,314.681000,314.681000,842300992
5	2017-11-12,314.690002,319.153015,298.513000,307.907990,307.907990,1613479936
6	2017-11-13,307.024994,328.415009,307.024994,316.716003,316.716003,1041889984
7	2017-11-14,316.763000,340.177002,316.763000,337.631012,337.631012,1069680000
8	2017-11-15,337.963989,340.911987,329.812988,333.356995,333.356995,722665984
9	2017-11-16,333.442993,336.158997,323.605988,330.924011,330.924011,797254016
10	2017-11-17,330.166992,334.963989,327.523010,332.394012,332.394012,621732992
11	2017-11-18,331.980011,349.615997,327.687012,347.612000,347.612000,649638976
12	2017-11-19,347.401001,371.290985,344.739990,354.385986,354.385986,1181529984
13	2017-11-20,354.093994,372.136993,353.289001,366.730011,366.730011,807027008

Рис. 10. Збережених даних Ethereum у форматі csv

Основними даними біржі смарт-контрактів Ethereum, що зображено на рис. 10 є:
- Дата створення транзакції (Date),
- Ціна першої операції з початку дня (Open),
- Максимальна ціна протягом дня продажу (High),
- Мінімальна ціна в день продажу (Low),
- Ціна з останньої операції у кінці дня торгівлі (Close),
- Ціна перед закриттям біржі (Adj Close),
- Кількість одиниць Ethereum яких було продано протягом дня торгівлі (Volume).

Для дослідження часових трендів використано мову програмування R, що є мовою програмування для статистичних обчислень та середовище розробки R-studio [12-13].

У таблиці 1 показано результат підрахунку значень вартості смарт-контракту Ethereum за весь період, використовуючи функцію summary із стандартного пакету функцій, а саме мінімальне значення, вартість за перший та третій квантиль, середнє та максимальне значення.

Таблиця. 1. Аналіз вартості Ethereum протягом періоду з 2017 по 2022

	Open	**High**	**Low**	**Close**	**Adj. close**	**Volume**
Min	84.28	85.34	82.83	84.31	84.31	6.217e+08
1st Qu.	196.43	201.42	188.85	196.61	196.61	3.154e+09
Median	386.37	396.50	375.45	386.45	386.45	9.525e+09
Mean	1026.06	1061.28	986.46	1027.56	1027.56	1.245e+10
3rd Qu.	1647.89	1721.58	1568.55	1659.37	1659.37	1.764e+10
Max	4810.07	4891.7	4718.04	4812.09	4812.09	8448e+10

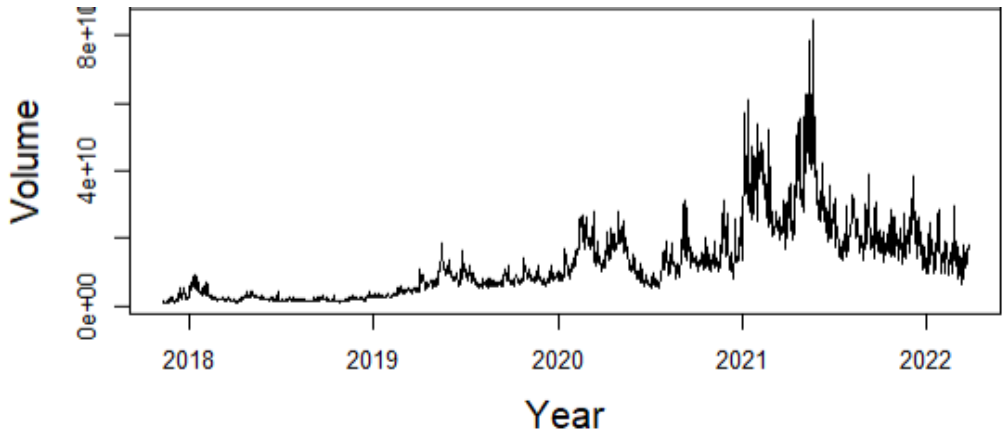


Рис. 11. Динаміка росту вартості Ethereum

На рис. 11 зображено динаміку кількості проданих смарт-контрактів Ethereum за період від 2017 по 2022 роки. Слід зазначити, що вартість Ethereum сильно залежить від кількості торгів на біржі. Ріст обєму торгів, що супроводжується ростом кількості транзакцій і відповідно розміром блоку в блокчейні. Одним із важливим показником інвестування коштів є фінансовий коефіцієнт ROI(від англ. return on investment) який ілюструє рівень прибутковості або збитковості вкладень[14]. Визначається ROI у відсотках, враховуючи суму інвестицій у Ethereum, визначається за формулою (1):

ROI = (прибуток + (ціна продажу - ціна придбання) / ціна придбання) * 100% (1)

Таблиця. 2. Прибуткові дні для інвестування коштів

Date	ROI
2015-08-11	50.809152
2015-08-13	49.534461
2016-02-11	33.718939
2017-05-19	33.46002
2016-01-23	31.151105

Таблиця. 3. Невдалі дні для інвестування коштів

Date	ROI
2015-08-08	-73.035443
2020-03-12	-42.308851
2021-05-19	-27.256029
2017-03-18	-27.123666
2016-06-18	-26.321642

Таблиця. 4. Невдалі місяці для інвестування

Year	Month	Monthly_ROI
2018	3	-53.685580
2015	8	-52.033112
2015	9	-45.426308
2018	11	-42.708751
2020	3	-39.207311

У таблиці 2 наведено результати досліджень прибуткових днів для інвестування, тобто у ці дні вартість Ethereum була найнижчою. У таблиці 3 наведено результати досліджень 5 невдалих днів для вкладень, тобто в ці дні ціна Ethereum була найвищою. Дослідивши Ethrereum у таблиці 4 наведено найбільш невдалі місяці для Ethereum відносно показника фінансового коефіцієнта. Чим нижчий показник ROI тим збитковішим є вкладення. Невдалі місяці припадають на серпень і вересень 2015 року та березень і листопа 2018 року.

4. Висновок

У даній роботі досліджено підхід моделювання, створення та написання смарт-контрактів за допомогою Solidity та допоміжних інструментів Genache, Truffle та середовища розробки Visual Studio Code. Реалізовано смарт-контракт, який передбачає продаж квитків. Загалом створення смарт-контрактів за допомогою Solidity вимагає добре розуміння об’єктно-орієнтованих мов програмування, проте більшість із наявних функцій у пакетах Solidity виконують однаковий функціонал. Унікальність Ethereum дає змогу програмам, що базуються на блокчейні виконуватись автономно, проте це також має свої недоліки. Також проведено аналіз часових трендів смарт-контракту Ethereum використовуючи мову програмування R та середовище розробки R-Studio.

Список використаних джерел

[1] *Ferretti Stefano, and Gabriele D'Angelo.* "On the ethereum blockchain structure: A complex networks theory perspective." Concurrency and Computation: Practice and Experience 32, no. 12 (2020): e5493.
[2] *Pavlyshenko Bohdan M*. "Bitcoin Price Predictive Modeling Using Expert Correction." In 2019 XIth International Scientific and Practical Conference on Electronics and Information Technologies (ELIT), pp. 163-167.
[3] *Singh Madhusudan, and Shiho Kim*. "Blockchain technology for decentralized autonomous organizations." In Advances in computers, vol. 115, pp. 115-140. Elsevier, 2019.
[4] *Buterin Vitalik*. "A next-generation smart contract and decentralized application platform." white paper 3, no. 37 (2014): 2-1.
[5] *Dannen Chris*. Introducing Ethereum and solidity. Vol. 1. Berkeley: Apress, 2017.
[6] *Liu Yue, Ju Yang, and Mingjun Liu.* "Recognition of QR Code with mobile phones." In 2008 Chinese control and decision conference, pp. 203-206. IEEE, 2008.
[7] Metamask, Retrieved from https://metamask.io.
[8] *Park Jong Soo, Ming-Syan Chen, and Philip S. Yu*. "Using a hash-based method with transaction trimming for mining association rules." IEEE transactions on knowledge and data engineering 9, no. 5 (1997): 813-825.

[9] *Hargreaves Andy, and Michael T. O'Connor*. "Solidarity with solidity: The case for collaborative professionalism." Phi Delta Kappan 100, no. 1 (2018): 20-24.
[10] *Na Dong Gyu, Jung Hwan Baek, Jin Yong Sung, Ji-Hoon Kim, Jae Kyun Kim, Young Jun Choi, and Hyobin Seo*. "Thyroid imaging reporting and data system risk stratification of thyroid nodules: categorization based on solidity and echogenicity." Thyroid 26, no. 4 (2016): 562-572.
[11] Kaggle studio. Retrieved from https://www.kaggle.com/
[12] R language. Retrieved from https://www.r-project.org/
[13] R-Studio. Retrieved from https://www.r-studio.com/
[14] *Vujičić Dejan, Dijana Jagodić, and Siniša Ranđić*. "Blockchain technology, bitcoin, and Ethereum: A brief overview." In 2018 17th international symposium infoteh-jahorina (infoteh), pp. 1-6. IEEE, 2018.

APPROACHES IN MODELING SMART CONTRACTS BASED ON ETHEREUM

Y. Tsudzenko

Ivan Franko National University of Lviv,
50 Drahomanova St., UA-79005 Lviv, Ukraine
yura9989@gmail.com

A smart contract is a type of digital agreement between several parties without intermediaries. In many areas of life, the smart contract is a breakthrough technology because it can change the perception of deals. At first sight, creating a smart contract may seem like a simple thing, but to create it, should to know and use a large number of technologies.

This article discusses method of creating smart contracts based on Ethereum using technologies such as the Ethereum network, the Solidity programming language, the Ethereum Virtual Machine (EVM), which takes the main role in creating and executing smart contracts and supporting instruments Ganache and Truffle. A smart contract has been implemented based on the sale of movie tickets using the previously mentioned technologies. It took 0.65 seconds to complete a transaction for the sale of 10 tickets between wallets. Once this transaction made, the smart contract cannot change, which makes it unique compared to other technologies. In general, creating smart contracts with Solidity requires a good understanding of object-oriented programming languages, but most of the functions available in the Solidity packages perform the same functionality. Ethereum's uniqueness allows blockchain-based applications to run autonomously, but this also has its drawbacks.

Also presented and investigated in the article an analysis of the time trends of the Ethereum smart contract out using the R programming language and the R-Studio development environment. Data of Ethereum smart contract for analysis taken from an open source Kaggle. The dataset contains 1599 rows of data and 8 unique columns of Ethereum smart contract values. The best days and months in relation to the ROI indicator investigated. Using this methodology and tools of creating smart contract can be used for further studies.

Keywords: smart contract, blockchain, Solidity, bitcoin, Ethereum, cryptocurrency.

Стаття надійшла до редакції 25.06.2023
Прийнята до друку 29.06.2023