Intelligent System for Arbitrage Situations Searching in the Cryptocurrency Market

Roman Levus[1], Andrii Berko[1], Lyubomyr Chyrun[2], Valentyna Panasyuk[3] and Mykhailo Hrubel[4]

[1] *Lviv Polytechnic National University, S. Bandera street, 12, Lviv, 79013, Ukraine*
[2] *Ivan Franko National University of Lviv, University street, 1, Lviv, 79000, Ukraine*
[3] *West Ukrainian National University, Lvivska Street, 11, Ternopil, 46004, Ukraine*
[4] *Hetman Petro Sahaidachnyi National Army Academy, Heroes of Maidan Street, 32, Lviv, 79012, Ukraine*

Abstract

This paper contains research work and development details of an information system for finding and executing arbitrage opportunities in the cryptocurrency market. In particular, the cryptocurrencies concept and its functional aspects are reviewed. Furthermore, the factors regarding price formation, the idea of arbitrage and its methods in the cryptocurrency market are presented. During the design of the system described in this paper, a comparative analysis of analogue systems and systems similar to the designed one is performed. The advantages of those systems have been taken into consideration during the software development of the projected system and the found solutions of identified problems of such systems. The most optimal type of system using the methods of system analysis is determined for the designed one among the chosen analogues, for which a tree of goals was built as well as to highlight the main goals and specific actions for each stage of system development to fully perform, selected evaluation criteria, etc. Data-flow diagrams have been created to describe the primary process that runs in the system and smaller ones, which do a necessary job for the successful functioning of the immediate circle and the system in general. Also, a diagram of the hierarchy of tasks is built. Before the direct development of the software of the designed system, a comparative analysis is performed to determine which programming language is best suited for such system, necessary frameworks and libraries to use, integrated development environment and other software to be used in the development of the system.

As a result of the development of the system, a client-server architecture application running on the Node.js platform is created. The main task that solves the developed software is finding and executing arbitrage opportunities in the cryptocurrency market among a wide range of popular cryptocurrency exchanges. The server component of the system performs data processing and computing of all the necessary tasks. The system is managed via a web page or bot in Telegram messenger. The paper contains user instructions regarding the usage of the developed software, its primary purpose, available functionality, logical structure, features, etc.

Keywords 1

The cryptocurrency market, information system, arbitrage opportunity, arbitrage transaction, arbitrage situation, projected strategy, software solution, API key, cryptocurrency exchange, telegram messenger, web application, hierarchy analysis method

MoMLeT+DS 2021: 3rd International Workshop on Modern Machine Learning Technologies and Data Science, June 5, 2021, Lviv-Shatsk, Ukraine
EMAIL: romanlevus@gmail.com (R. Levus); Andrii.Y.Berko@lpnu.ua (A. Berko); Lyubomyr.Chyrun@lnu.edu.ua (L. Chyrun); v.panasiuk@tneu.edu.ua (V. Panasyuk); m.g.grybel@gmail.com (M. Hrubel)
ORCID: 0000-0002-4855-1472 (R. Levus); 0000-0001-6756-5661 (A. Berko); 0000-0002-9448-1751 (L. Chyrun); 0000-0002-5133-6431 (V. Panasyuk); 0000-0002-4820-6935 (M. Hrubel)

1. Introduction

Arbitrage is a logical sequence of actions on a particular asset, leading to an increase in its number. For example, this happens if the same asset is traded at different prices in different markets.

The concept of arbitration is far from new and has existed for an extended period. Banks and other financial institutions worldwide have been using this practice in the stock markets, Forex and others for decades. The cryptocurrency market is no exception to the existence of arbitrage opportunities. Features of the functioning of cryptocurrencies, the youth of the market, and many trading platforms make it particularly attractive for financial speculators and traders. Binance, Coinbase, Kraken, Bittrex are just a few of the list of popular cryptocurrency exchanges that are unique platforms for trading cryptocurrencies. These exchanges provide their users with the opportunity to apply a software solution for conducting trading. Each user has access to the API, which is a set of methods for interacting with the exchange. Thus, the trader can receive and analyze data from the business and conduct various operations in automatic mode, having compiled and programmed an algorithm for performing actions. Therefore, it is possible and appropriate to create an information system that can identify arbitrage opportunities in the cryptocurrency market and conduct them independently.

With the rapid development and popularization of cryptocurrencies in recent years, many new stories of virtual money have appeared, which has led to the emergence of a large number of cryptocurrency exchanges. The phenomenon of decentralization of cryptocurrencies, the lack of proper regulation by the state, high volatility, etc. – all these factors create challenging conditions for the existence of a single price and, accordingly, form many arbitrage opportunities in the cryptocurrency market.

The purpose of the work is to create an information system that identifies arbitrage opportunities in the cryptocurrency market among a wide range of exchanges and has the functionality to automatically conduct such transactions with the user's consent and report on completed transactions.

The task of the work is to analyze literature and other sources closely related to the subject, disclose the main concepts and terms; describe arbitration methods, in particular in the cryptocurrency market; conduct a comparative analysis of existing trading systems, highlight the main advantages and disadvantages, implement their strengths and solve the identified problems during the development of a software solution for the system. The object of research is the phenomenon of Arbitration in the cryptocurrency market. The subject of the study is methods of finding, conducting and eliminating arbitrage opportunities in the cryptocurrency market. The practical significance of the results consists of applying data obtained based on collection and analysis by the system for information or management purposes with the possibility of automatic arbitration transactions by the user.

2. Related works

2.1. Cryptocurrency. Definition of the concept

Throughout its history, money had changed a lot and has gone through many stages of development: starting with its very appearance, when its function was performed by various kinds of items: shells, precious metals or stones and the banal conduct of barter, exchange of one product for another, to a significant displacement of cash from circulation and the transition to contactless payments. Today, the rapid development of Information Technologies has given an impetus to the next stage of changing money and even the beginning of transforming the economic system as a whole. One reason for such significant changes is the emergence of electronic money and cryptocurrencies [1].

The first cryptocurrency to be born is Bitcoin. The concept of cryptocurrency is usually defined as a virtual currency that works on encryption methods for generating monetary units, verifying its transfer. It does not depend on a central management body, such as a bank [2]. One of the main features of cryptocurrency is its decentralization due to blockchain technology and the absence of the need to attract third parties to conduct transactions since the network ensures equality of all participants (peer-to-peer technology) [3]. However, cryptocurrency cannot be classified as a classic category of money due to the lack of legal status in many countries [4].

In October 2008, a developer or group of developers (unknown for sure) under the pseudonym Satoshi Nakamoto presented the so-called White Paper, which described the concept of the Bitcoin network and aspects and principles of its functioning. In early 2009, the first version of the source code of this system was published on the Internet and subsequently, the first transaction was successfully carried out. The transfer of funds in the network is carried out using special software and the principle of asymmetric encryption. The public key is the wallet address, a generated string of characters and numbers that does not require input identification data from the user, thus ensuring anonymity. Thanks to blockchain technology, transaction transparency is also provided: you can track all wallet operations, but it is pretty challenging to establish a connection with the owner. The number of new address generations per user is unlimited. Transactions on the network are irreversible [5].

With the advent of the first cryptocurrency, other projects soon appeared. Since the source code of the Bitcoin network is freely available, some are almost a complete copy of the so-called fork. However, some cryptocurrencies are radically different in architecture. An example of such a network can be considered Ethereum, which occupies the second line among the popularity and capitalization of the cryptocurrency market. Ethereum is an open software platform based on blockchain technology that allows you to create decentralized applications, implement so-called smart contracts, which is a specific contract, the execution, modification or termination of which is carried out only with the help of pre-defined software tools. Such instructions are created using the Solidity programming language, similar to JavaScript [6].

2.2. Price formation in the cryptocurrency market

Before the appearance of money in the form in which it exists today, without performing the function of a "measure" of value, the value of goods was determined in specific equivalent values of another commodity. Today, the purchasing power of money depends on the specifics of the sphere of circulation, where it is used and can be exchanged for authentic goods. The principle of supply and demand determines the price of a particular cryptocurrency and any other physical or virtual asset. Pricing is affected by the need for people to exchange cryptocurrencies for other goods. Since virtual assets do not directly depend on a particular country's economic processes or policies, a significant factor affecting its price is the interest of people. Public interest in new, decentralized money and contactless payment is growing every year. Distrust of the government, political instability, significant loss of purchasing power of fiat currencies against the background of economic crises – all these factors force people to look for an alternative and diversify their assets, particularly by investing in cryptocurrencies [7]. Attractive features of many cryptocurrencies, in particular Bitcoin, for ordinary users are conducting transactions on the network without intermediaries and a limited number of units in circulation, which means the almost complete absence of inflation. The maximum number of coins is 21,000,000. The first cryptocurrency is mined by mining – a process in which the user uses specific equipment to solve a cryptographic equation. For this, he receives a reward in the form of a certain amount of BTC. Every four years, the value of the prize drops, and mining becomes more complex, which in theory should stimulate the growth of the price of cryptocurrencies. The direct factor for price formation is the mining of Bitcoin itself (the cost of equipment and its efficiency, electricity, reward value, etc.). So, by analogy with precious metals and minerals, the first cryptocurrency is also characterized by a limited resource. Thus, Bitcoin is a kind of "digital gold"[8].

Just over a year after the launch of the Bitcoin network, the first recorded exchange of cryptocurrency for a physical commodity took place: this happened on May 22, 2010, in Florida, USA. A local programmer paid 10,000 BTC for two pizzas. This event later went down in history under the name "Bitcoin Pizza Day" [9]. Today, cryptocurrencies are already accepted by thousands of establishments and services worldwide. Their number is growing every day: restaurants, cafes, gas stations, electronics stores and online game purchases, hosting services, mobile phone replenishment, and so on. There are already thousands of cryptocurrency conversion points and terminals in the world that can be used to purchase cryptocurrency for cash. In particular, many of them are already located on the territory of Ukraine. Also, payment card systems are developing that work like regular bank cards, converting from the user's account the amount in the selected cryptocurrency, equivalent to the payment value, into fiat money, allowing you to pay in any store where regular bank cards are accepted [10]. The beginning of the formation of the cryptocurrency market began with the appearance of the first cryptocurrency – Bitcoin. Although it was difficult to estimate its value at first due to the then low prevalence, the price did not exceed a few cents. Bitcoin experienced significant changes in the price a year after the network's launch, with the appearance of the first cryptocurrency exchange – Mt.Gox. And already in early 2011, the cost of the first cryptocurrency crossed the mark of 1 US dollar.

Over time, Bitcoin has become widespread, and since 2012, more and more retailers have started implementing cryptocurrency payments as an alternative payment method. At the end of 2013, the price of 1 BTC crossed the 1,000$ mark for the first time, thanks to the popularization of Bitcoin in China, the opening of new exchanges and the beginning of cryptocurrency mining on an industrial scale.

The cryptocurrency market has experienced many ups and downs in its history. In the beginning, the cryptocurrency industry was subjected to numerous attacks that significantly affected pricing. So, in 2014, due to the lack of a proper security system and proper management, the most popular exchange at that time, Mt.Gox, filed for bankruptcy due to the theft of about 850,000 BTC. This is the most significant cryptocurrency theft in history. Users are still trying to get their funds back today. After these events, the price for 1 Bitcoin dropped to 580$ and continued its prolonged fall for almost two years to 315$. In early 2017, the price of Bitcoin returned to 1,000$. The emergence and development of new projects, such as Litecoin, Ripple, Ethereum and others, the boom of the so-called initial coin offering (English ICO – initial coin offering) - a form of attracting initial investment in a project similar to the initial public offer of shares on the stock market, contributed to another rapid growth in the capitalization of the cryptocurrency market. Over the years, interest in cryptocurrencies has grown not only among ordinary users but also among financial institutions worldwide. A significant event was introducing Bitcoin futures contracts on the Chicago Mercantile Exchange, which made it possible to attract traders from traditional stock and Forex markets to the cryptocurrency market and significantly increase trading volumes. In December 2017, the cryptocurrency market's capitalization reached a peak, and the price for 1 Bitcoin reached almost 20,000$ – its historical maximum [11]. Today, more and more countries are trying to legally regulate and legalize working with cryptocurrencies and their legal status, and financial institutions from different parts of the world are developing and implementing systems that would enable ordinary users to invest in cryptocurrencies and use them. The emergence of hundreds of cryptocurrency exchanges, margin trading, futures, derivatives and other trading instruments indicates an increase in public interest in cryptocurrencies. In recent years, there has been a correlation with stock indexes of securities, which means the arrival of investors and traders from traditional markets. The cryptocurrency market's capitalization as of the end of April 2020 is about 265 billion US dollars [12]. Projects of state-owned digital currencies are emerging, and testing of the national cryptocurrency has even begun in China, which indicates the inevitable transformation of the global economic system using digital money.

2.3. Arbitration. Features of Arbitration in the cryptocurrency market

Eliminating the price difference with profit extraction by performing a specific sequence of actions that start and end with the same asset, which, if completed, leads to its increase at the end of this sequence, is called arbitrage. For example, if we buy share A for 10$ on exchange X and sell it on exchange Y for.10.30$, we will get 0.30$ in profit, excluding commissions [13].

The concept of arbitrage is not new, and various types of price discrepancies exist in all financial markets, including stock exchanges and currency markets. This trading principle has been used for many years by banks and other financial institutions around the world. For example, currency arbitrage involves buying and selling currency pairs from different brokers to take advantage of the price difference to make a profit [14]. The cryptocurrency market is no exception to Arbitration, where these discrepancies are even more significant compared to traditional markets. Today, there are about 4,000 cryptocurrencies. A side effect of this was a similar increase in the number of cryptocurrency exchanges. Binance, Coinbase, Kraken, Bitfinex, Bittrex are just a few of the many popular businesses that exist. Low level of state regulation, decentralization, a large number of speculators seeking to make money, high volatility – all these factors create difficult conditions for the existence of a single price and make arbitration in the cryptocurrency market particularly attractive [15].

Arbitrage in the cryptocurrency market almost entirely works on the same principles as in traditional markets, but with a different asset. Significant price discrepancies between exchanges, on a large scale, often occur against the background of political instability. So, in August 2019, in Argentina, against the backdrop of a sharp drop in the national currency, the price for 1 Bitcoin was 4% higher than on international platforms in terms of US dollars. But even if you remove extreme economic and political events, price differences among exchanges occur more often than they seem. There are many reasons for arbitrage situations in the cryptocurrency market: from a sharp collapse or increase in prices on one of the exchanges, the price reaction to the news, to the so-called "fat-finger error", when a user accidentally entered incorrect data from the keyboard or mistakenly clicked the mouse, which led to the execution of an order at an undesirable price or volume [16].

Operations related to crypto fraud trading are conducted on unique platforms – cryptocurrency exchanges (or crypto exchanges). A crypto exchange operates similarly to a currency or stock exchange [17]. Discussions introduce the concept of a currency pair – a pair consisting of two currencies that are traded against each other. The first currency is usually called the base currency, and the second – the quota [18]. For example, if you want to buy BTC for USD, you need to find the BTC/USD pair. In addition to cryptocurrency/fiat currency pairs, cryptocurrency/cryptocurrency pairs are also popular on most exchanges, which are often even more in demand than fiat money pairs due to certain features of this market. All user requests fall into the so-called "exchange glass" - a list of recommendations for buying and selling a particular cryptocurrency, organized by price level. Such a list lists the amount of a particular cryptocurrency that is sold or offered for purchase at each price, forming the depth of the market [19]. The critical feature of cryptocurrency exchanges is that users can easily quickly access the vast majority of them. At minimal cost, transfer the purchased cryptocurrency to their wallet for storage, thereby not depending on third parties or another exchange, which brokers and currency exchanges or stock exchanges cannot boast of, due to high commissions, lack of decentralization of traded assets, a fixed work schedule and a long transaction time. Arbitration methods that can be applied in the cryptocurrency market are presented in the paragraphs below.

2.3.1. Spatial arbitration

This technique involves extracting profit from the difference in prices for buying and selling a particular cryptocurrency traded on two different exchanges. For example, if exchange a can offer BTC for 6,500$ and the price on sale B reaches.6,700$, a trader can take advantage of this 200$ by buying on exchange A and selling on exchange B – effectively moving funds from one platform to another. Given the degree of decentralization in cryptocurrencies, such disagreements may occur more often than you might think. Do not forget about the commission for conducting trading operations and transaction costs for withdrawing cryptocurrency from exchanges, the time of cryptocurrency transfer is also essential. The longer it is, the more risky the opportunity to make a profit is [20]. An example of performing spatial arbitration is shown in Fig. 1.

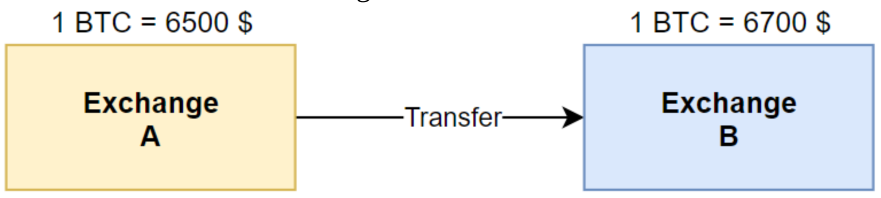


Figure 1: Example of spatial arbitration

2.3.2. Intra-exchange arbitration

A special feature of this method is that it occurs within only one exchange. Its principle is to conduct a sequence of trading operations starting and ending with the same currency. For example, in three markets traded on the exchange, with different quotas, the following rates are: BTC/USD = 6726.94, ETH/USD = 141.58 and ETH/BTC = 0.0214. We sell 1 BTC for USD, get 6726.94 USD, for the received USD, we buy 47.5133 ETH and sell ETH for BTC – get 1.0167 BTC, the profit is 0.0167 BTC. This type of arbitration is also called "Triangular" (Fig. 2) [21].

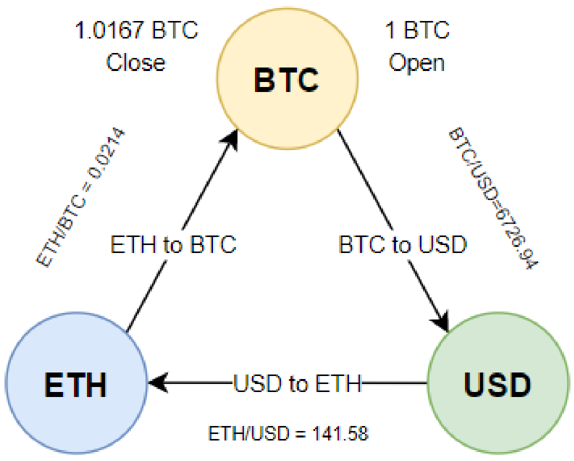


Figure 2: Example of intra-exchange arbitration

It is worth noting that this method of arbitration requires speedy data collection and analysis and, accordingly, the execution of all necessary trading operations. It is also essential to take into account trading commissions and types of order execution. To find such arbitrage situations, various algorithms are often used to find the shortest path in a weighted graph, from one vertex to all the others.

2.3.3. Statistical arbitration

This method of arbitrage is considered a high-tech strategy, usually involving mathematical modelling. This type of arbitration is based on maintaining a historical ratio between the value of identical or similar assets on different trading platforms. This method does not require the transfer of cryptocurrency from one exchange to another, thereby reducing the time for performing arbitrage operations by several times, and works as follows: when the price difference between two exchanges is higher than their historical ratio, two functions will be performed simultaneously: selling on the first exchange and buying on the second, and vice versa – when the price difference is lower than the historical ratio. When the price ratio approaches its typical values, positions are closed. Using this method, market risks are almost nonexistent since the aggregate function consists of multidirectional parts on the same asset, which protects against unpredictable market fluctuations. The only risk of such a strategy is a change in the historical ratio value between instruments [22].

It should be noted that arbitration in the cryptocurrency market is a rather risky type of earnings, which depends on financial literacy, user skills in working with cryptocurrency, experience, speed of action, and often even banal luck [23].

2.4. Automated trading. Overview of existing trading systems

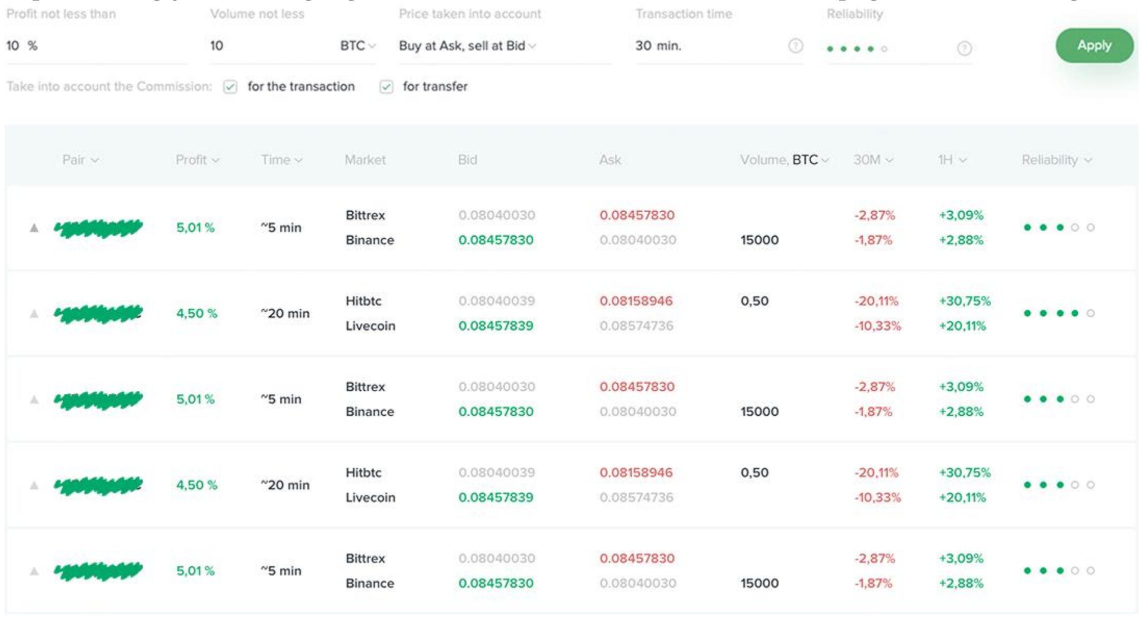
At the end of the last century, with the development of Information Technologies, the sphere of trading on the stock and foreign exchange markets also underwent computerization. As a result, almost all trading operations are carried out using electronic means: personal computers, tablets, smartphones, etc. Today, the user can perform all trading operations using the web interface of a broker or exchange, use special trading terminals with extensive functionality, or develop their software that will serve the algorithm of actions specified by the user without his participation.

At the beginning of the XXI century, automated trading, or the so-called "robot trading", began to develop rapidly. The Trader no longer has to monitor changes in the price and sit on the charts for days because it is possible to implement his strategy using unique trading systems. Such automated systems collect data from exchanges. Then, according to pre-set instructions, they perform a specific algorithm that serves the entire sequence of actions instead of the user, according to pre-set instructions [24].

The advantages of this approach to trading are the ability to obtain and analyze a massive amount of data and check many market conditions in seconds, which is beyond the power of a person. At the same time, the risk of manual errors when performing trading operations is also reduced. Most importantly, this eliminates the emotional impact on the trading process, which often leads to losses. Thus, algorithmic trading provides a systematic and unchanging approach to active trading, compared to a person. Today, many automated trading systems allow users to create their algorithms for trading to make a profit. Such procedures cover the performance of a specific sequence of actions, depending on the results of volume calculations using complex mathematical formulas and initial rapid receipt and processing of the necessary information. In particular, the use of such a system will be effective in finding and executing arbitrage opportunities in the cryptocurrency market. Access to the required data and the ability to automate trading operations is provided by cryptocurrency exchanges using the API (Application Programming Interface) – an application software interface that is a set of methods for working with the business [25]. Several popular trading systems for the cryptocurrency market are presented in the paragraphs below.

2.4.1. Arby.Trade

Search service for arbitrage situations in the cryptocurrency market. It helps beginners and advanced users find the most profitable and reliable arbitrage opportunities. The system collects up-to-date and necessary search data online (prices, trading volumes, technical characteristics of currencies, etc.) from the most popular crypto exchanges, analyzes them and offers users the most favourable options for arbitrage transactions. The result of the system is displayed in the form of a table in which the user can set their parameters for the search – profit, transaction volume, transaction time, degree of reliability of the operation, as well as develop a list of exchanges where you need to search for transactions. The service provides search for spatial and intra-exchange arbitrage situations. The system is implemented as a web service that offers cross-platform performance. It is possible to send notifications about the discovery of arbitrage opportunities. Conducting arbitrage situations automatically and the possibility of implementing your trading algorithms is absent [26]. The main service page is shown in Fig. 3.



Profit not less than	Volume not less	Price taken into account	Transaction time	Reliability
10 %	10 BTC	Buy at Ask, sell at Bid	30 min.	● ● ● ● ○

Take into account the Commission: ✓ for the transaction ✓ for transfer

Pair	Profit	Time	Market	Bid	Ask	Volume, BTC	30M	1H	Reliability
[redacted]	5,01 %	~5 min	Bittrex Binance	0.08040030 0.08457830	0.08457830 0.08040030	15000	-2,87% -1,87%	+3,09% +2,88%	● ● ● ○ ○
[redacted]	4,50 %	~20 min	Hitbtc Livecoin	0.08040039 0.08457839	0.08158946 0.08574736	0,50	-20,11% -10,33%	+30,75% +20,11%	● ● ● ● ○
[redacted]	5,01 %	~5 min	Bittrex Binance	0.08040030 0.08457830	0.08457830 0.08040030	15000	-2,87% -1,87%	+3,09% +2,88%	● ● ● ○ ○
[redacted]	4,50 %	~20 min	Hitbtc Livecoin	0.08040039 0.08457839	0.08158946 0.08574736	0,50	-20,11% -10,33%	+30,75% +20,11%	● ● ● ○ ○
[redacted]	5,01 %	~5 min	Bittrex Binance	0.08040030 0.08457830	0.08457830 0.08040030	15000	-2,87% -1,87%	+3,09% +2,88%	● ● ● ○ ○

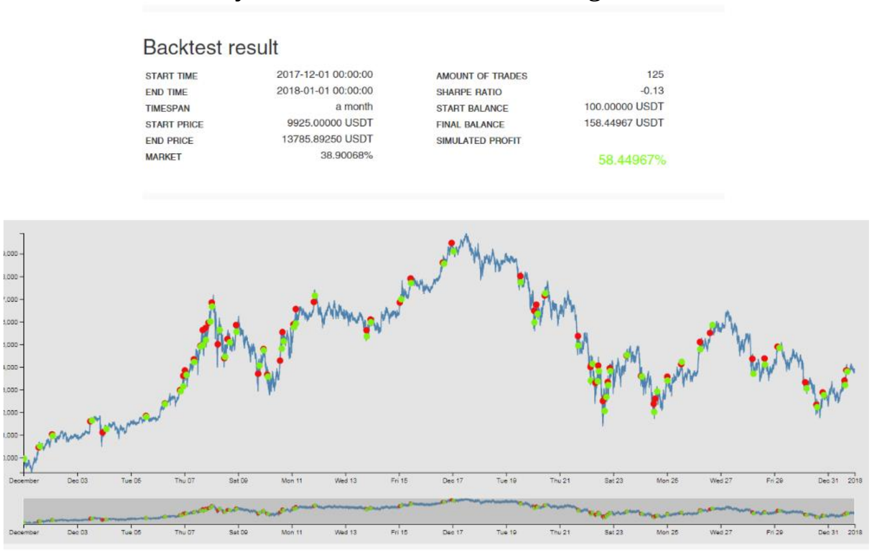
Figure 3: Arby.Trade interface

2.4.2. Blackbird

A trading system is written in C++ that detects arbitrage situations on more than five crypto exchanges and automatically executes them. The system's main task is to find and eliminate price differences in the market by selling cryptocurrencies on one exchange and buying on another. A low-market risk strategy is used, similar to statistical arbitrage, which involves trading activities involving parallel purchases and sales on two different exchanges without the need to transfer cryptocurrency from business to exchange. There is a minimal graphical interface. The main number of instructions from the user comes via the command line. [27].

2.4.3. Gekko

A free and open-source trading system is that supports several popular cryptocurrency exchanges. It is written in JavaScript and runs on Node.js. The main task of the plan is to automate your trading strategies. Gekko provides an opportunity to create trading strategies using popular technical analysis indicators or use ready-made ones. To check and identify the optimal strategy, it is possible to simulate trading using historical market data. There is also a system of plugins that allows you to add your functionality [28]. A view of the system interface is shown in Fig. 4.


Backtest result

START TIME	2017-12-01 00:00:00	AMOUNT OF TRADES	125
END TIME	2018-01-01 00:00:00	SHARPE RATIO	-0.13
TIMESPAN	a month	START BALANCE	100.00000 USDT
START PRICE	9925.00000 USDT	FINAL BALANCE	158.44967 USDT
END PRICE	13785.89250 USDT	SIMULATED PROFIT	
MARKET	38.90068%		58.44967%

Figure 4: Gekko Interface. Checking your strategy based on historical data

2.4.4. Zenbot

The trading robot is written in JavaScript and runs on the Node platform.js using the MongoDB database. Implemented support for a wide range of functionality is for use among more than 12 cryptocurrency exchanges. The system provides an opportunity to create your trading strategies based on technical analysis. The plugin support architecture offers the ability to add new discussions independently and develop new functionality and process. Zenbot provides the ability to create buy or sell orders with many configurations, which allows the system to work as a trading terminal. It is possible to conduct trading simulations to test strategies based on historical data. The system works according to the user's instructions via the terminal [29].

Most of the existing systems are paid to use, with the need for a monthly subscription fee. Some works allow you to develop, test and implement your trading strategies. However, few systems can identify or execute arbitrage opportunities in the market and even collect all the necessary data to implement such operations successfully. Also, there is no technical implementation of the transfer of cryptocurrencies from exchange to exchange automatically in any of the systems, which excludes the possibility of conducting classical spatial arbitration. The disadvantages of the designs are also a small number of supported exchanges and a fixed list of cryptocurrency pairs, which reduces the number of arbitrage opportunities found. Critical for arbitrage search systems is sending notifications about such situations and managing the system remotely.

Based on the conducted comparative analysis of trading systems for the cryptocurrency market, it can be concluded that the current system for finding and executing arbitrage transactions will be a cross-platform system with support for a wide range of exchanges and all possible cryptocurrency pairs, with the possibility of remote management, user notifications about such found situations, with the possibility of their settings, and most importantly with a component for automating the execution of a sequence of operations, which will fully collect information for successful arbitrage. Furthermore, implementing your custom and trading strategies is not mandatory since the method of spatial arbitration in the cryptocurrency market is chosen for the projected system.

Table 1
Comparison of system capabilities

Functionality / System name	Arby.Trade	Gekko	Blackbird	Zenbot
Collecting data on prices and volumes	Yes	Yes	Yes	Yes
Collecting technical data on cryptocurrencies	Yes	No	No	No
Search for arbitration situations	Yes	No	Yes	Yes
Ability to perform arbitrage transactions	No	No	Yes	Yes
Ability to implement your trading algorithms	No	Yes	No	Yes
Cross-platform	Yes	Yes	Yes	Yes
Push notifications	Yes	Using plugins	No	Using plugins

In the first chapter, information about the system being projected was reviewed and analyzed, which is the first step for its future development. The concept of cryptocurrency, principles and aspects of its functioning are revealed. The factors influencing price formation and the role of cryptocurrencies in everyday life are considered. The features of conducting arbitration in the cryptocurrency market are displayed, and popular methods are described. Existing trading systems are analyzed, their capabilities and functionality are reviewed, and strengths and weaknesses are highlighted. Based on the analysis, the necessary functionality and problems that the developed information system should solve are determined.

3. Material and methods

3.1. System analysis of the research object

To start developing an information system, it is necessary to analyze the research object, identify and classify several current problems, and find solutions. One of the methods for solving such problems is to build a goal tree. A goal tree is a hierarchical diagram consisting of relationships and subordinates of the system's goals, which reflect the distribution of the system's goals into sub-goals and tasks or individual actions. This method is based on decomposition and is universal, and is one of the main ones in systems analysis. Decomposition is a way of dividing a system into its components. The goal tree method is used to establish the relationship between the main goal and the ways to achieve it, which are presented in specific actions or tasks. The main principle of building a goal tree is to simplify a complex process to its elementary components. To comply with this rule, follow the following approach:

- a goal located at the highest level is the basis and a kind of reference point for separating goals situated at the lowest level;
- goals that are at the lowest level are means of implementing the task at the highest level and are presented so that their unification provides for the fulfilment of the main goal [30].

The result of building a system goal tree is shown in Fig. 5.

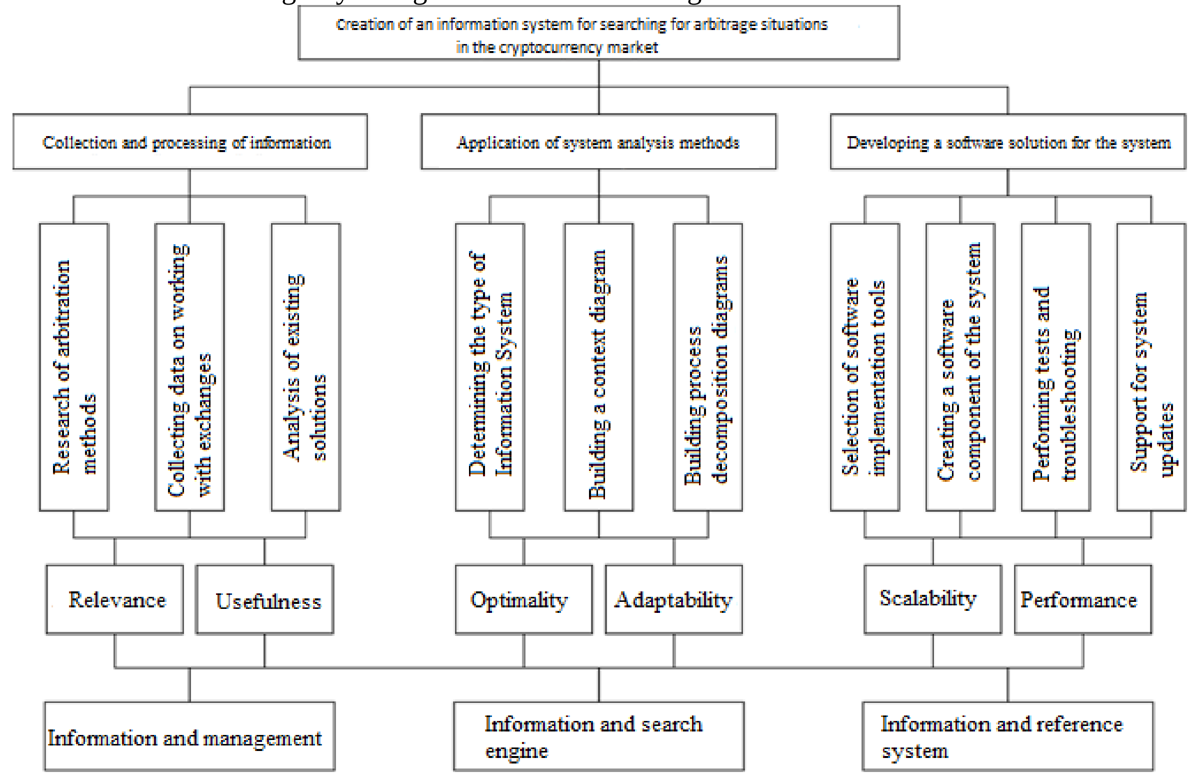

Creation of an information system for searching for arbitrage situations in the cryptocurrency market
Collection and processing of information
Application of system analysis methods
Developing a software solution for the system
Research of arbitration methods
Collecting data on working with exchanges
Analysis of existing solutions
Determining the type of Information System
Building a context diagram
Building process decomposition diagrams
Selection of software implementation tools
Creating a software component of the system
Performing tests and troubleshooting
Support for system updates
Relevance
Usefulness
Optimality
Adaptability
Scalability
Performance
Information and management
Information and search engine
Information and reference system

Figure 5: Goal tree

The main goal specified by the goal tree construction method is to create an information system for searching for arbitrage situations in the cryptocurrency market. This problem can only be solved after all its sub-goals are completed. Therefore, the constructed tree has four levels: the primary goal, sub-goals, tasks that need to be completed, and criteria for evaluating the quality of execution. As a result of creating the goal tree, the following sub-goals were formed:

- Collection and processing of information;
- Application of system analysis methods;
- Developing a software solution for the system.

Consider the first sub-goal - "Collecting and processing of information". At this stage, the analysis of requirements and relationships between the components of the system is carried out, an analytical review of the literature and other sources related to the subject area and the projected plan of direction is carried out. Based on the analysis, several such tasks were identified:

- Research of arbitration methods. To create an information system for searching for arbitration situations, correctness, completeness and accuracy of calculations in such a system, it's needed to familiarize yourself with the very concept of Arbitration, possible methods of conducting it and its features in the cryptocurrency market;
- Collecting data on working with exchanges. This task provides for familiarization with the principle of operation on cryptocurrency exchanges, analysis of the technical documentation of the API of exchange methods for the successful creation of appropriate modules, high-quality typing and implementation in the system;
- Analysis of existing solutions. To create a high-quality and multifunctional product, it's needed to analyze existing software solutions that solve similar problems or are close to the projected system. Highlight their strengths and weaknesses, functionality, and so on.

Create the correct functional model of the projected system, use the analyzed systems' found advantages, solve the found problems, and present their solutions in your system.

For the first sub-goal, we selected such quality assessment parameters as relevance and usefulness, which are the main properties for collecting and analyzing information. The second sub-goal is"Application of system analysis methods". Before creating an information system, you need to select alternative systems, determine the optimal information system among them, and design its central processes, functions, and tasks to solve. The result of this sub-goal is to determine the type of information system that is being projected and a series of diagrams to explain the processes that are taking place. This sub-goal provides for solving several such tasks:

- It is determining the type of Information System. This task involves applying the hierarchy analysis method. By performing paired comparisons of the values of quality ratings among alternative systems, the most optimal version of the system is selected;
- Building a context diagram. To display the primary process or function of the system and its details, it's needed to create a context diagram DFD or IDEF0;
- Building process decomposition diagrams. Based on the context diagram, you need to decompose it to create subsequent charts that specify lower-level processes and details of their implementation.

Optimality and adaptability were chosen as the criteria for evaluating this sub goal.

The last sub-goal for achieving the main goal is "Developing a software solution for the system", which includes performing the following actions and tasks:

- Selection of software implementation tools for the system. Based on the prototype of the system, it's needed to choose the development environment and software implementation tools that are best suited for its development to meet the requirements and tasks that this system should perform;
- Creating a software component of the system. This task consists of writing the code of the necessary modules for the full functioning of the system using pre-selected software implementation tools;
- Performing tests and troubleshooting. This task involves checking and debugging identified problems in the software development process;
- Support for system updates. It is necessary to provide the information system with support for the software product and its updating, which is extremely necessary in cases of changes in the operation of exchanges or the emergence of new arbitration methods, ideas for expanding the functionality to ensure the relevance of the system, and so on. Also, do not forget about the possibility of anomalies during the system's operation, which must be detected and eliminated.

The selected criteria for evaluating the quality of execution of this sub-goal are scalability and performance because these properties play a crucial role during the system's functioning and for its further development in the future. The hierarchy analysis method is chosen to determine the type of Information System. This method is used to solve many practical problems at different stages of system design. The essence of the process is to use paired comparisons of criteria for evaluating the quality of execution of sub-goals to select priority solutions.

Three types of information systems were chosen as alternatives:

- Information and management (A1);
- Information and search engine (A2);
- Information and reference system (A3).

Determining the most optimal type of Information System be projected is carried out by using pre-selected six quality criteria:

- Relevance (C1);
- Usefulness (C2);
- Optimality (C3);
- Adaptability (C4);
- Scalability (C5);
- Performance (C6).

Since the system's goal tree has already been constructed, it is now necessary to create a matrix of paired comparisons of criteria and determine the values of eigenvalues (HF) and eigenvectors (HV) used in the following analysis. Calculating the importance of priorities will be determined by finding the relationships of criteria using scores to evaluate them ranging from 1 to 9, where 1 is the equilibrium criteria, and 9 is the absolute advantage of one standard over another.

The values of eigenvalues are calculated using the Formula 1:

$$\mathrm{HF} = \sqrt[n]{\prod_{j=1}^{n} c_{ij}}, \quad (1)$$

where: n – is several criteria;

c_{ij} – is a relation of specific criteria;

$\prod_{j=1}^{n} c_{ij}$ – is a product of all relations of a specific criterion with other criteria.

The values of eigenvectors of criteria can be calculated using Formula 2:

$$\mathrm{HV} = \frac{s_i}{\sum_{i=1}^{n} s_i}, \quad (2)$$

where: s_i – is an eigenvalue value of a particular criterion;

$\sum_{i=1}^{n} s_i$ – is a sum of the eigenvalues of all criteria.

Table 2 below shows the values of paired comparison matrices for the relations of each of the criteria.

Table 2
Matrix of paired comparisons

Criteria	C1	C2	C3	C4	C5	C6	HF	HV
C1	1.00	0.50	0.25	0.33	0.25	0.14	0.34	0.04
C2	2.00	1.00	0.25	2.00	0.33	0.16	0.61	0.07
C3	4.00	4.00	1.00	3.00	0.50	0.33	1.41	0.17
C4	3.00	0.50	0.33	1.00	0.25	0.20	0.54	0.07
C5	4.00	3.00	2.00	4.00	1.00	0.33	1.78	0.22
C6	7.00	6.00	3.00	5.00	1.00	1.00	3.52	0.43

The next step in the hierarchy analysis method is to construct paired comparison matrices for the selected alternatives A1, A2, and A3, for each criterion and relative to the general goal. The values are calculated using the same principle and formulas as for the matrix of paired comparisons. The calculations for each of the criteria and the general goal are presented in the corresponding tables below.

Table 3
Matrix of paired comparisons of alternatives

relevance	A1	A2	A3	HF	HV
A1	1.00	0.50	0.33	0.55	0.16
A2	2.00	1.00	0.50	1.00	0.30
A3	3.00	2.00	1.00	1.82	0.54
usefulness	A1	A2	A3	HF	HV
A1	1.00	3.00	2.00	1.82	0.54
A2	0.33	1.00	0.50	0.55	0.16
A3	0.50	2.00	1.00	1.00	0.30
optimality	A1	A2	A3	HF	HV
A1	1.00	4.00	6.00	2.88	0.70
A2	0.25	1.00	2.00	0.79	0.19
A3	0.16	0.50	1.00	0.43	0.10
adaptability	A1	A2	A3	HF	HV
A1	1.00	2.00	3.00	1.82	0.54
A2	0.50	1.00	2.00	1.00	0.30
A3	0.33	0.50	1.00	0.55	0.16
scalability	A1	A2	A3	HF	HV
A1	1.00	6.00	5.00	3.11	0.75
A2	0.16	1.00	0.20	0.32	0.08
A3	0.20	2.00	1.00	0.74	0.18
performance	A1	A2	A3	HF	HV

A1	1.00	7.00	8.00	3.83	0.83
A2	0.14	1.00	3.00	0.75	0.16
A3	0.13	0.33	1.00	0.04	0.01
general goal	A1	A2	A3	HF	HV
A1	1.00	0.33	2.00	0.87	0.23
A2	3.00	1.00	5.00	2.47	0.65
A3	0.50	0.20	1.00	0.46	0.12

Based on the obtained results of calculations of eigenvalues and eigenvectors for alternatives, it is necessary to construct the following generalized matrix of their comparisons and determine the most optimal information system for searching for arbitrage situations in the cryptocurrency market. The alternative comparison matrix is shown in Table 4.

Table 4
Generalized alternative comparison matrix

Criteria and alternatives	C1	C2	C3	C4	C5	C6	Priority value
A1	0.16	0.54	0.70	0.54	0.75	0.23	0.49
A2	0.30	0.16	0.19	0.30	0.08	0.65	0.28
A3	0.54	0.30	0.10	0.16	0.18	0.12	0.23

Using the hierarchy analysis method, the most optimal type of Information System with the highest coefficient of 0.49 was determined – information and management, which most corresponds to the type of system being projected.

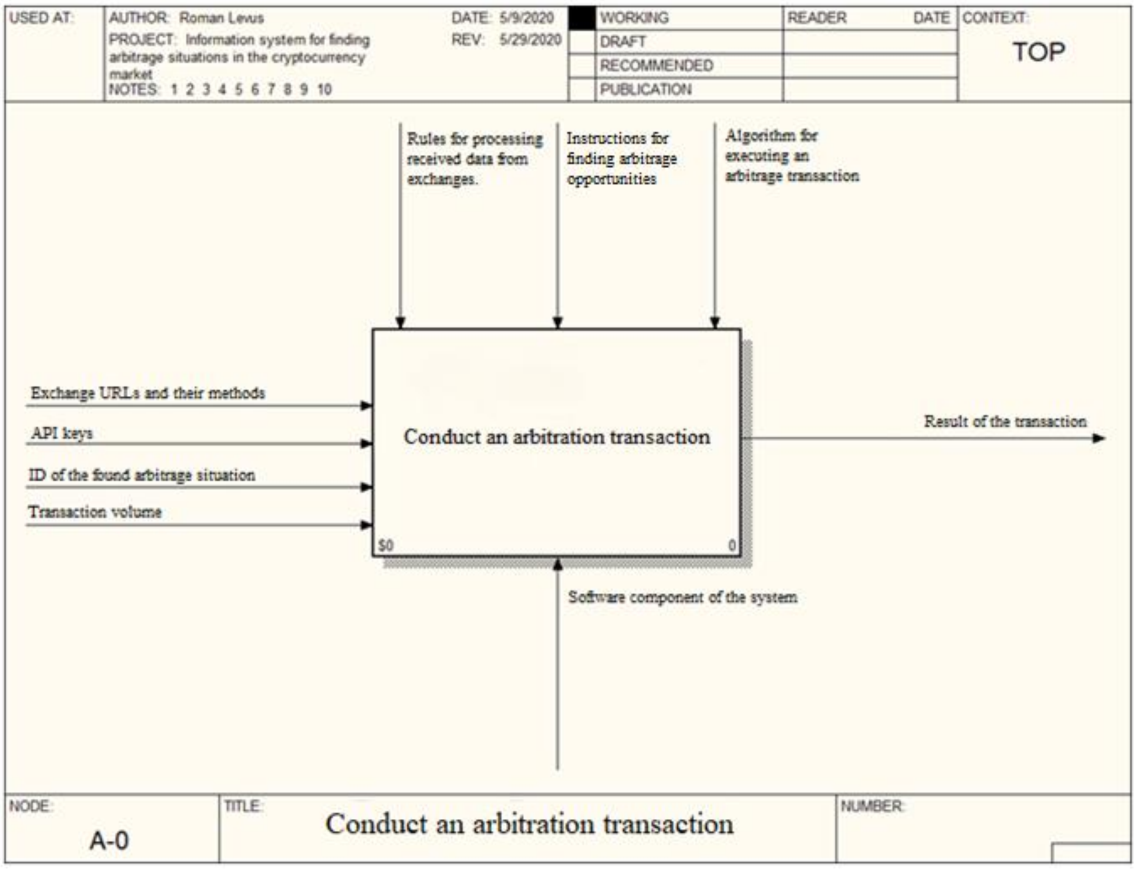
To specify the functioning of the projected system, it's needed to build a contextual diagram. Such a diagram is created according to the IDEF0 standard and is used for the most general description of processes in the system. A characteristic feature of such a diagram model is the emphasis on the hierarchical structure of representation of system objects and the definition of logical relationships between processes, which gives a better understanding of the project being developed [31].

The context diagram for the projected system (shown in Fig. 6) describes the system's most general and essential process – conducting an arbitration transaction.

The user initially provides API keys to their exchange accounts to the system so that a particular component of the system can perform the actions necessary to execute an arbitrage transaction successfully. Also, when selecting an arbitrage situation to fulfil, the user provides its identification code and indicates the desired transaction volume by clicking on the corresponding button.

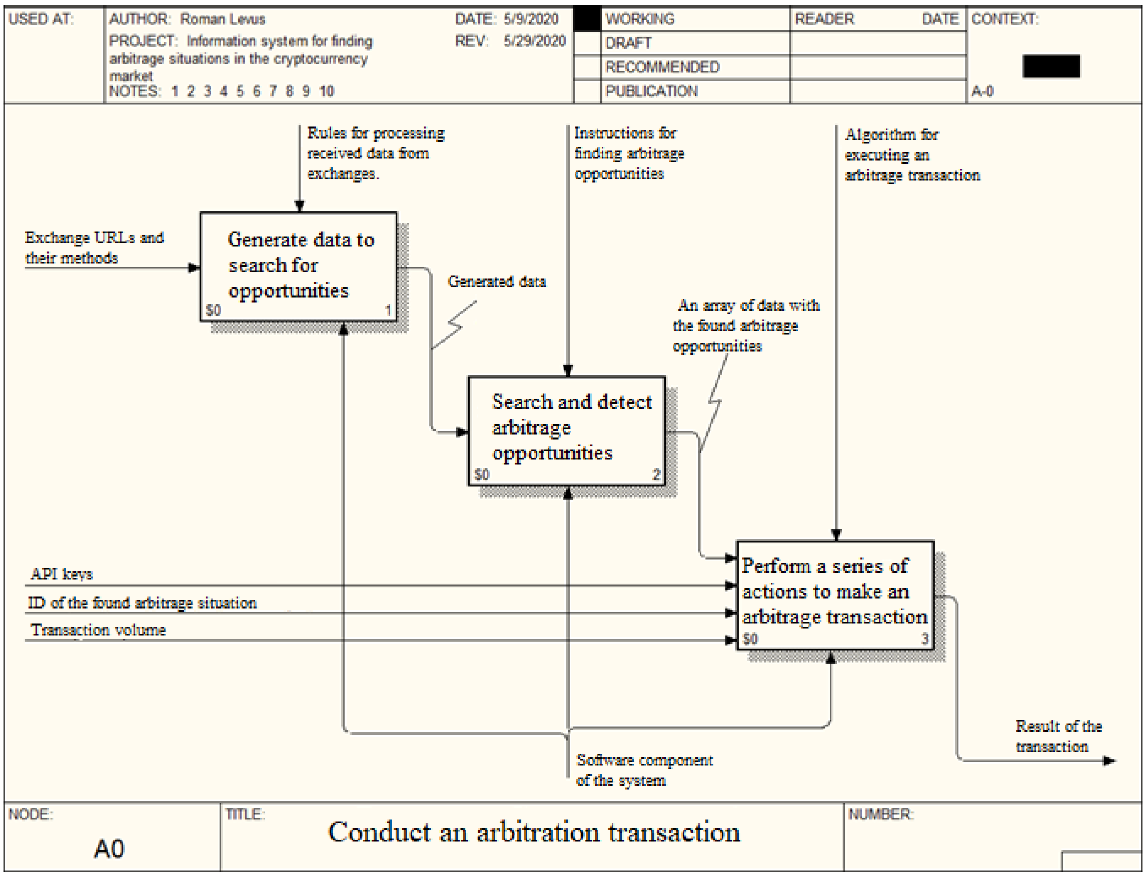
After constructing a context diagram describing the most general process of the system, you need to perform a functional decomposition process to separate the individual components of the system. The result of this process will be the first-level decomposition diagram shown in Fig. 7. According to some rules, the process of "Generate data to search for arbitrage opportunities" involves collecting the necessary data from exchanges and converting it to the desired form in JSON format, according to some rules, for further transmission to the next process "Search and detect arbitrage opportunities". The resulting data is analyzed in this process, resulting in an array of data with the found arbitrage capabilities.

The last process, "Perform a series of actions to make an arbitrage transaction", receives data from the previous method that will be used for the final stage of the primary system process. This process involves several smaller tasks related to performing manipulations on the user's exchange accounts. Again, the user's choice with the identification code and transaction volume is taken into account. This process will be the formation of data on the conduct and completion of an arbitration transaction. After performing the first-level decomposition, each process obtained in this way can be divided into smaller ones, resulting in second-level decomposition diagrams. A diagram of this type for the "Generate data to search for arbitrage opportunities" process is shown in Fig. 8. The second-level decomposition diagram for the "Search and detect arbitrage opportunities" function is shown in Fig. 9.



USED AT:
AUTHOR: Roman Levus
PROJECT: Information system for finding arbitrage situations in the cryptocurrency market
NOTES: 1 2 3 4 5 6 7 8 9 10
DATE: 5/9/2020
REV: 5/29/2020
WORKING
DRAFT
RECOMMENDED
PUBLICATION
READER
DATE
CONTEXT:
TOP
Rules for processing received data from exchanges.
Instructions for finding arbitrage opportunities
Algorithm for executing an arbitrage transaction
Exchange URLs and their methods
API keys
ID of the found arbitrage situation
Transaction volume
Conduct an arbitration transaction
$0
0
Result of the transaction
Software component of the system
NODE:
A-0
TITLE:
Conduct an arbitration transaction
NUMBER:

Figure 6: Context diagram



USED AT:
AUTHOR: Roman Levus
PROJECT: Information system for finding arbitrage situations in the cryptocurrency market
NOTES: 1 2 3 4 5 6 7 8 9 10
DATE: 5/9/2020
REV: 5/29/2020
WORKING
DRAFT
RECOMMENDED
PUBLICATION
READER
DATE
CONTEXT:
A-0
Rules for processing received data from exchanges.
Instructions for finding arbitrage opportunities
Algorithm for executing an arbitrage transaction
Exchange URLs and their methods
Generate data to search for opportunities
$0
1
Generated data
An array of data with the found arbitrage opportunities
Search and detect arbitrage opportunities
$0
2
API keys
ID of the found arbitrage situation
Transaction volume
Perform a series of actions to make an arbitrage transaction
$0
3
Result of the transaction
Software component of the system
NODE:
A0
TITLE:
Conduct an arbitration transaction
NUMBER:

Figure 7: First-level decomposition diagram

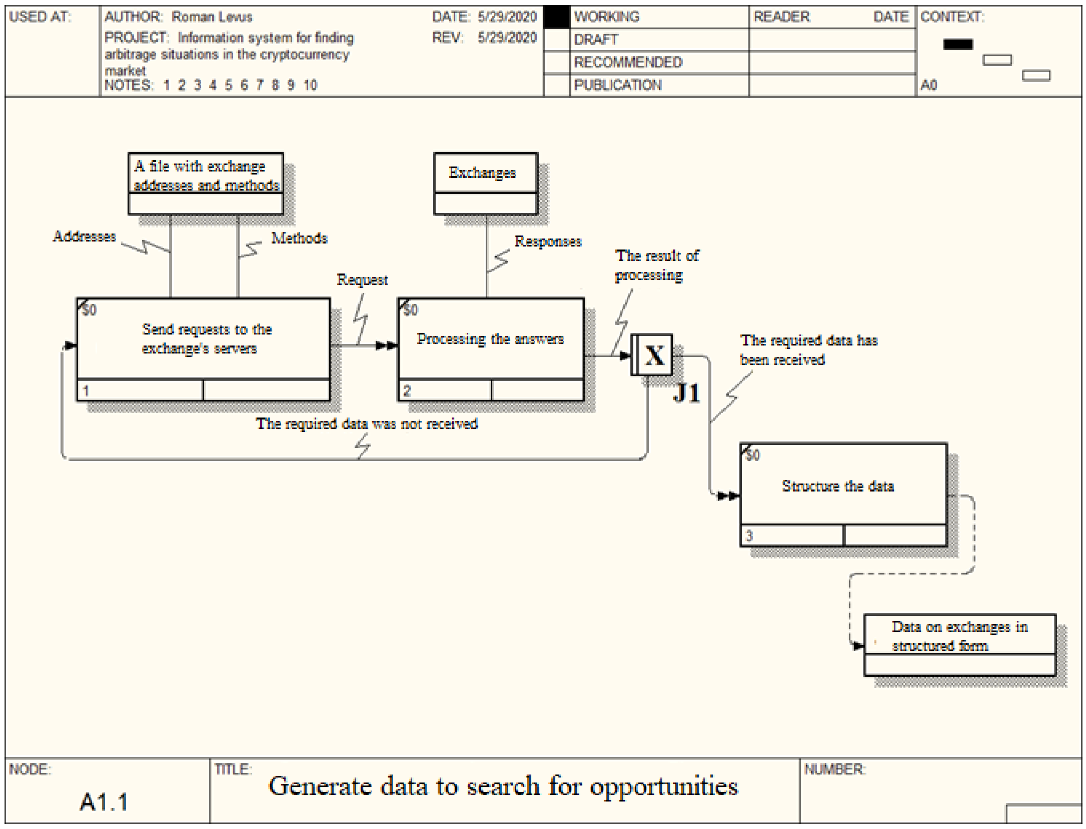

USED AT:
AUTHOR: Roman Levus
DATE: 5/29/2020
REV: 5/29/2020
PROJECT: Information system for finding arbitrage situations in the cryptocurrency market
NOTES: 1 2 3 4 5 6 7 8 9 10
WORKING
DRAFT
RECOMMENDED
PUBLICATION
READER
DATE
CONTEXT:
A0
A file with exchange addresses and methods
Exchanges
Addresses
Methods
Responses
Request
The result of processing
Send requests to the exchange's servers
Processing the answers
X
J1
The required data has been received
The required data was not received
Structure the data
Data on exchanges in structured form
NODE:
A1.1
TITLE:
Generate data to search for opportunities
NUMBER:

Figure 8: The decomposition diagram for "Generate data to search for arbitrage opportunities"

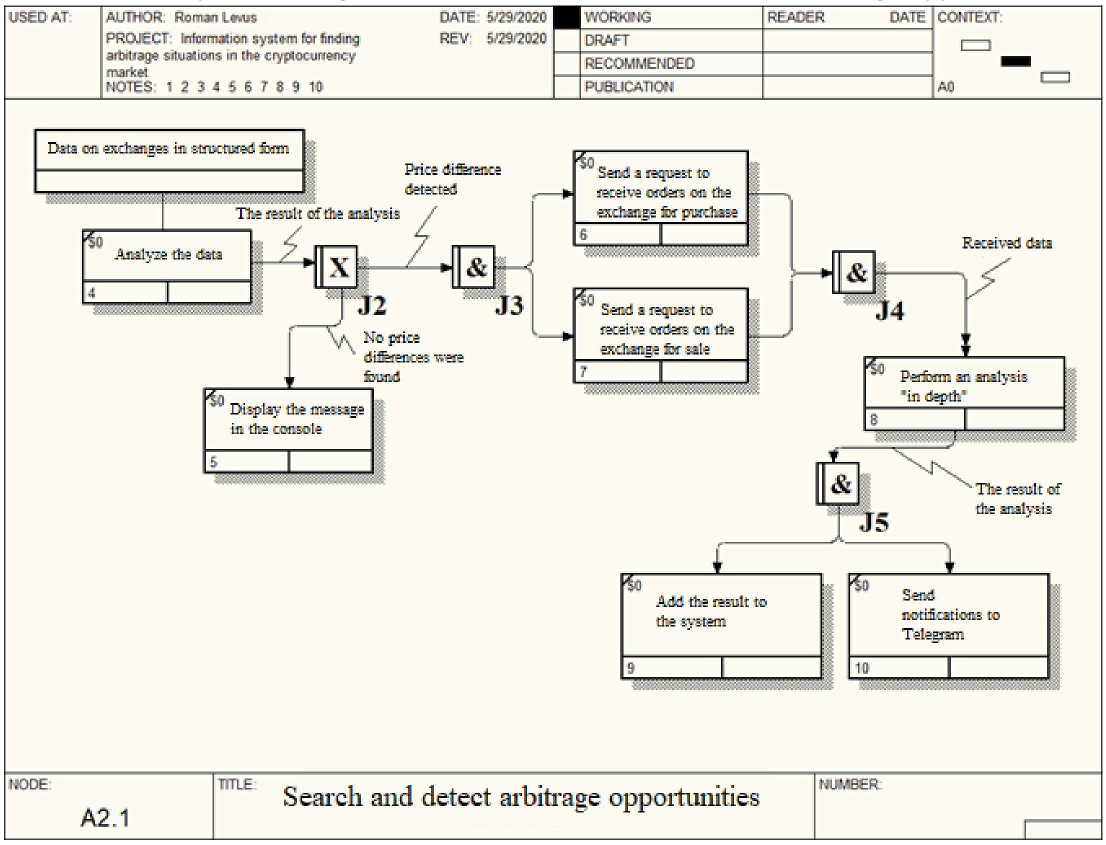

USED AT:
AUTHOR: Roman Levus
DATE: 5/29/2020
REV: 5/29/2020
PROJECT: Information system for finding arbitrage situations in the cryptocurrency market
NOTES: 1 2 3 4 5 6 7 8 9 10
WORKING
DRAFT
RECOMMENDED
PUBLICATION
READER
DATE
CONTEXT:
A0
Data on exchanges in structured form
Analyze the data
The result of the analysis
Price difference detected
X
J2
&
J3
No price differences were found
Display the message in the console
Send a request to receive orders on the exchange for purchase
Send a request to receive orders on the exchange for sale
&
J4
Received data
Perform an analysis "in depth"
&
J5
The result of the analysis
Add the result to the system
Send notifications to Telegram
NODE:
A2.1
TITLE:
Search and detect arbitrage opportunities
NUMBER:

Figure 9: The decomposition diagram for the "Search and detect arbitrage opportunities" process

The process of "Perform a series of actions to make an arbitration transaction" involves several small functions. These processes perform manipulations with the user's account and transmit intermediate data, the results of performing certain operations, among themselves. The input data for the following action is the output data of the previous one. At the end of the process, the arbitration transaction is generated, and a notification is sent to the user. The second-level decomposition diagram for the "Perform a series of actions to perform an arbitrage transaction" process is shown in Fig. 10.

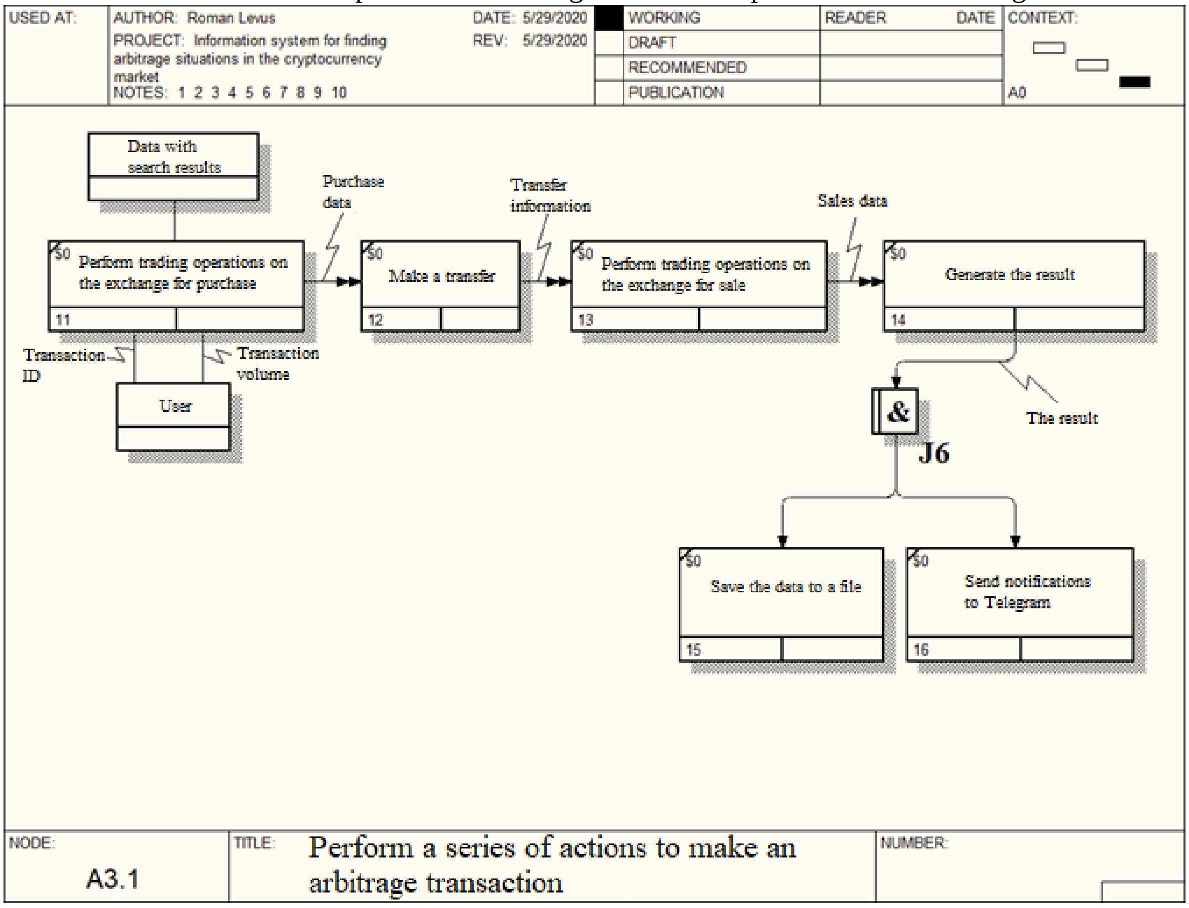

USED AT:
AUTHOR: Roman Levus
PROJECT: Information system for finding arbitrage situations in the cryptocurrency market
DATE: 5/29/2020
REV: 5/29/2020
NOTES: 1 2 3 4 5 6 7 8 9 10
WORKING
DRAFT
RECOMMENDED
PUBLICATION
READER
DATE
CONTEXT:
A0
Data with search results
Perform trading operations on the exchange for purchase
11
Purchase data
Make a transfer
12
Transfer information
Perform trading operations on the exchange for sale
13
Sales data
Generate the result
14
Transaction ID
Transaction volume
User
&
J6
The result
Save the data to a file
15
Send notifications to Telegram
16
NODE: A3.1
TITLE: Perform a series of actions to make an arbitrage transaction
NUMBER:

Figure 10: The decomposition for "Perform a series of actions to make an arbitrage transaction"

By constructing a hierarchy of processes of the projected system, its hierarchical structure is reflected, and various tasks are demonstrated. The process hierarchy diagram shows all the tasks necessary for the full functioning of the system and their relationships with each other.

The main goal in this diagram is the main task of the system "Conduct an arbitration transaction", which is divided into three smaller tasks that are the main components of the system: generate data to search for arbitration opportunities, search for and identify arbitration opportunities, perform several actions to make an arbitration transaction. The tasks listed above apply a certain number of necessary functions for the entire operation of the system. The first task, "Generate data to search for arbitrage opportunities", performs the tasks of collecting data from exchanges, analyzing them, and generating data in the necessary, structured form. The second task, "Search and detect arbitrage opportunities", takes the generated data obtained to solve the previous problem as input and analyses it. The result of this analysis is an array of data with the arbitrage capabilities found.

The third task, "Perform a series of actions for the implementation of an arbitration transaction", involves obtaining data from the previous component and the user, after which, following specific instructions, it manipulates exchange accounts, which ultimately leads to the completion of the arbitration transaction, the result is formed with detailed data. Finally, a notification about the product is sent to the user, which is the designed system's general goal.

In this chapter, we performed system analysis methods for the projected system. A goal tree is built, the general goal and sub-goals are defined, and individual actions and tasks that need to be performed to complete the main goal successfully are defined. Several alternative systems were selected, and the most optimal type of Information System was determined among them, using the analytical hierarchy method, by constructing matrices of paired comparisons of criteria for evaluating sub-goals and alternatives. The best option for the designed system is an information and management system.

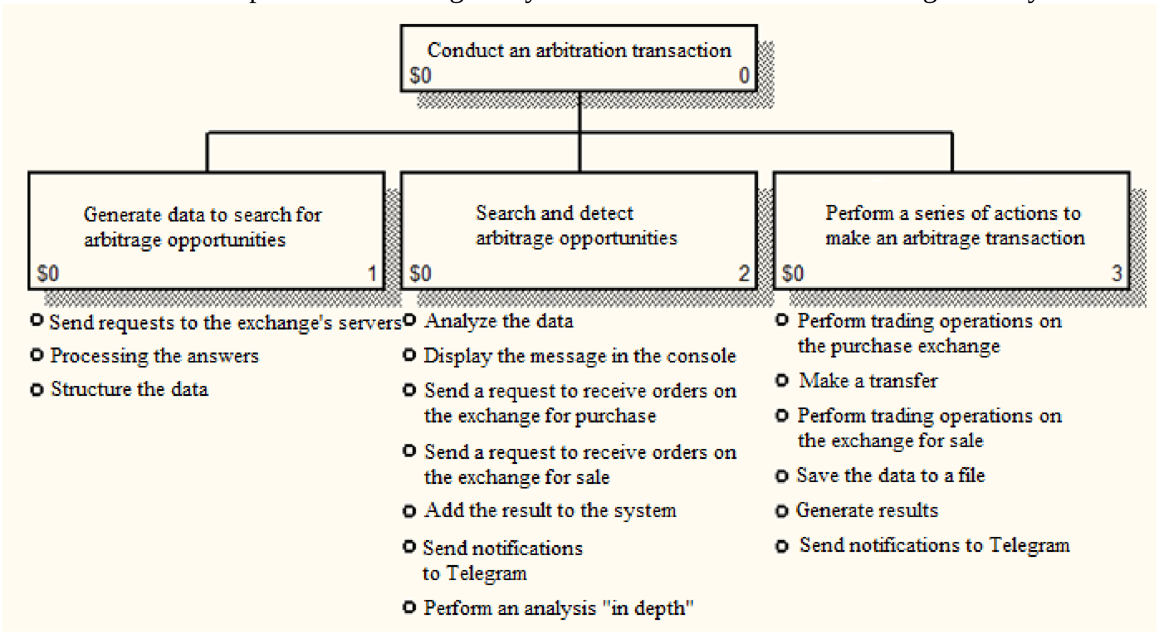

Conduct an arbitration transaction
$0 0
Generate data to search for arbitrage opportunities
$0 1
Search and detect arbitrage opportunities
$0 2
Perform a series of actions to make an arbitrage transaction
$0 3
Send requests to the exchange's servers
Processing the answers
Structure the data
Analyze the data
Display the message in the console
Send a request to receive orders on the exchange for purchase
Send a request to receive orders on the exchange for sale
Add the result to the system
Send notifications to Telegram
Perform an analysis "in depth"
Perform trading operations on the purchase exchange
Make a transfer
Perform trading operations on the exchange for sale
Save the data to a file
Generate results
Send notifications to Telegram

Figure 11: Task hierarchy diagram for the projected system

A context diagram is constructed depicting the primary relationships in the system. Then, its decomposition of two levels is carried out to display more detailed data on small actions and processes occurring in the design, creating a task hierarchy diagram for the projected plan.

3.2. Software tools for solving the problem

One of the most important decisions before starting software development, which should be made as soon as possible, is the choice of software tools that will be used to create the system. Software tools include the programming language itself (or languages, if there are several of them), libraries and frameworks, the operating system, the development environment, testing software, and so on.

The most important factor influencing the choice of programming language is directly the projected system itself, for whom it is created and what tasks it should solve and what to work. The next factor is the type of system in which it should be implemented: mobile application, console application, browser extension, web application, website, etc. An equally important factor to consider when choosing a programming language is the popularity of the selected language. It is closely related to the number of available frameworks and libraries, language support, and future development. Choosing a popular programming language will make it easier to update and develop the projected system. In addition to selecting a programming language, you should choose a set of necessary tools for software construction: the development environment and other software used in the project creation process.

3.2.1. Comparison of Node.js and Python

Before we talk about Node.js, it is necessary to mention the JavaScript programming language itself directly. JavaScript is a programming language created by Netscape Communications in 1995. During these years, there was a need to make web pages dynamic for implementing animations, interacting with the user, etc. So, there was a need to create a scripting programming language for solving such problems. The similar syntax and rapid growth in the popularity of Java have given the programming language an appropriate name. But JavaScript was very slow compared to traditional compiled programming languages. Years of research and development have taken the programming language to a new level and made it possible to create large-scale applications such as Google Maps.

For a long time, JavaScript was executed only in the browser. However, in 2009, a framework called Node.js appeared, which allowed you to play JavaScript code on the server. Node.js is a platform written in C++ that will enable you to execute JavaScript files. It allowed JavaScript to use this platform to perform several actions, such as performing activities on system files that are not available in the browser version. Node.js is based on the V8 engine developed by Google. It is a fast and progressive virtual machine with a built-in interpreter that compiles JavaScript to native code. Unlike programming languages such as C++, C#, Java, Python, etc., JavaScript on the Node.js platform is an event-driven programming language that provides asynchronous input and output capabilities. Each process is called when a specific event occurs, which means that none of the methods blocks the flow. Because JavaScript code is executed outside of the browser on the Node.js platform, the performance of this programming language becomes significantly higher, as the V8 engine allows you to use resources more efficiently. In addition, the event-driven, flow-free platform architecture will enable you to process multiple requests simultaneously, which positively affects efficiency and speeds up code execution.

Syntax in Node.js is almost no different from browser-based JavaScript. Also, Node.js provides the ability to use heavily typed TypeScript. In Node.js, when developing an application, it is preferable to create a set of modules and microservices that communicate with each other using a specific mechanism and perform their processes separately. It makes the Node.js platform's development process quite flexible since there is no need to create a monolithic system, and you can always easily add a new module. An active and numerical community creates a significant advantage for Node.js - the availability of tens of thousands of modules and libraries to facilitate work and solve various publicly available tasks. In Node.js, modules are managed using the npm package manager, where instructions are executed by entering specific commands in the terminal.

Scope of application of Node.js is basically the development of the server-side of web applications. However, the platform provides all the opportunities for creating mobile applications, desktop applications, cloud services, solutions for the Internet of Things segment, etc. Also, a significant advantage of Node.js is cross-platform, meaning the created application will run on such popular operating systems as Windows, Linux, Mac, etc. [32].

The development of the Python programming language began at the end of 1989, when Guido van Rossum, a Dutch programmer from the Center for mathematics and computer science (CWI) in Amsterdam, started working on the project. Initially, development was considered a hobby, and the software solution itself was intended to solve existing problems and become a successor to the ABC Language. The first version, 0.9.0, of the Python programming language was published in 1991. It differs significantly from languages such as Java or C++. The main task during its development was to create a language whose code is straightforward to read. In 2000, Python 2.0 was released. With the release of this version, the functionality of the programming language was quite expanded: a garbage collector appeared, which automatically performed operations on Ram, Unicode support, and so on.

Today, Python is a high-level open-source programming language. The latest version of the language at the time of writing is Python 3.8. Python is a reasonably popular language and has an active audience. Because of this, there are also many open-access libraries for solving various tasks. The installation and management of additional packages and libraries in Python are managed by a system that runs on the command line and is called pip. Applications created in Python will run on major popular operating systems, as the language is cross-platform. It can be used for both server-side and client-side tasks.

One of the most important advantages and features of the Python programming language is its syntax. The language is relatively concise and straightforward and does not require square brackets but instead correct tabs or spaces. Because of these features in the language's syntax, it is elementary for specialists of various levels to read and understand. The design of the Python language involves the use of a monolithic model in development. Of course, by using specific libraries and packages, you can build applications that allow you to write asynchronous code and manage events in the program. But this is not a native solution for the language, and debugging, in this case, takes quite a long time. The Python programming language is an interpreted programming language and is slower than traditional languages whose code compiles. This language does not support multithreading, but only works with one thread, due to the technical features of the language. However, again, using a certain number of libraries, you can reproduce certain multithreading functionality. Still, the interpreter – Gil mechanism that uses Python does not allow you to perform specific tasks simultaneously but will enable you to perform operations on only one thread at a time [33].

Table 5
Comparative characteristics of Node.js and Python

Criteria	Node.js	Python
Architecture	Event-driven doesn't block the flow	Synchronous code execution blocks the flow
Performance	Multithreading support	Single-threading
Syntax	Similar to browser-based JavaScript	Concise, tab-based, and easy to read
Scalability	Easily scalable, microservices model	Hard-to-scale, monolithic model
Libraries	Large number	Large number
Versatility	Ability to develop web applications, desktop and mobile applications	Ability to build web applications, desktop and mobile applications
Popularity and community	High	High
Application	Complex single-page web applications, Internet of things, real-time operation	Big data, 3D modelling, artificial intelligence

When choosing the programming language that will be used to develop the software solution of the projected system, preference was given to JavaScript.

Thanks to the features of the Node.js architecture, this platform, because it is event-based, is best suited for developing applications that will frequently perform numerous parallel queries and communicate between the client and the server in real-time. Although the JavaScript and Python languages are dynamically typed, however, using the microservices approach in Node.js, unlike the monolithic model in Python, is a more appropriate option for the scalability of the developed system. Furthermore, this approach will be better for developing extensible projects, making it easier to work, and giving you a better understanding of the system when it is maintained. The Node.js platform, unlike Python, is multithreaded, and requests are processed much faster, which makes this platform, due to better performance, a favourite choice for creating a web system [34].

3.2.2. Using auxiliary modules

Express.js. Express.js is one of the most popular server frameworks for the Node.js platform used in web application development. It is a kind of "backbone" in development when creating the server part of a web system. Provides a wide range of features for creating single-and multi-page web applications. This framework is flexible and does not impose any specific correct way to perform a particular task. This module allows you to quickly develop an application software interface (API) and create mechanisms for processing various query methods, such as GET, POST, PUT, DELETE, and others [35].

Request. Request module for Node.js is in demand when working with HTTP requests. Although it is a wrapper around the native HTTP module Node.js, it is much easier to operate. The module provides the ability to send different types of requests, use proxies, other network adapters when sending requests, TSL/SSL, and choose different configurations. Given its functionality, this module is a good solution for implementing data collection methods by the server and creating an API for the client part of the system [36].

Fs. The fs module (from the file system) is one of the most basic and necessary Node.js modules. This library of functions is used when working with operating system files. The set of methods in this module allows you to perform various manipulations with files and with the computer's file system:

read, write, supplement the File, Create, Edit and delete directories, and so on. Each method has two versions to work with: synchronous and asynchronous, depending on the need. With even such a concise set of plans, this module provides the program with the ability to perform all basic operations on operating system files or their directories [37].

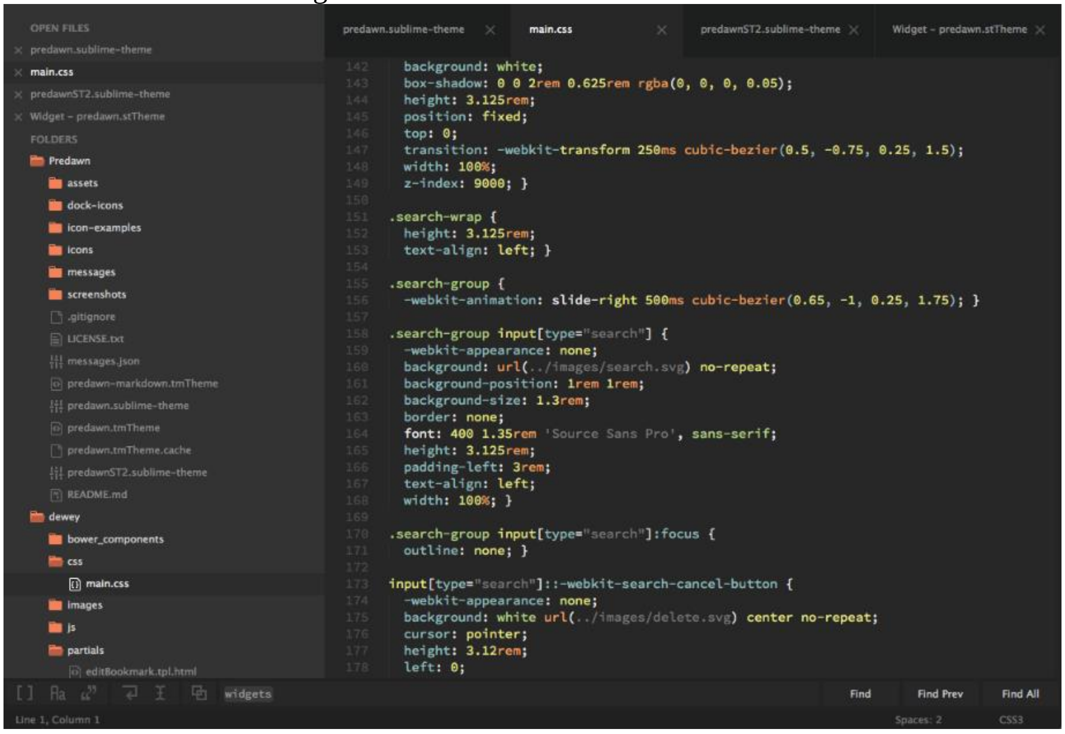
3.2.3. Atom development environment

Creating and editing code- Atom, the development environment for creating the projected information system, was chosen as a text editor. The first version of this software appeared in 2014 and has undergone several updates, and has gained considerable popularity since then. The program is developed by one of the largest software development communities – GitHub. This software product is completely free and can be used on operating systems such as Windows, Linux, Mac and others.

The main advantage of Atom is the ability to install third-party packages that significantly modify the editor, making it a full-fledged development environment. Today, there are about 9,000 modules for Atom in the public domain, helping developers solve various tasks and software development. In addition, the editor can also directly modify any developer settings because the program is open-source and written using technologies such as JavaScript, CSS, and HTML.

This program is suitable for developing a graphical user interface and creating the server part of web applications. Atom already has a basic wide range of valuable tools and functions and the ability to create your functionality or add existing ones by installing the necessary package.

Atom provides the ability to easily and quickly navigate through project files, use auto-completion technology (also known as IntelliSense) for popular programming languages, quickly edit HTML and CSS files and view changes on the page in real-time, edit many files at the same time, execute console commands and run programs without leaving the program environment, etc. [38]. For example, the program interface is shown in Fig. 12.

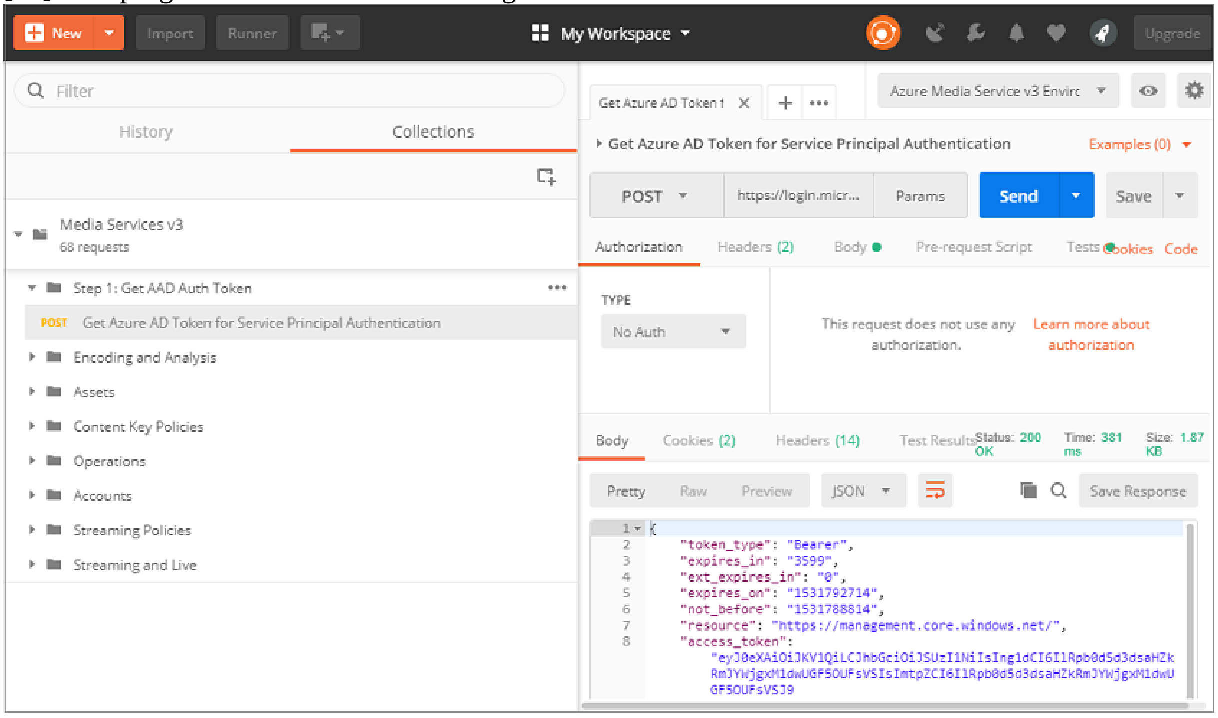


```
  background: white;
  box-shadow: 0 0 2rem 0.625rem rgba(0, 0, 0, 0.05);
  height: 3.125rem;
  position: fixed;
  top: 0;
  transition: -webkit-transform 250ms cubic-bezier(0.5, -0.75, 0.25, 1.5);
  width: 100%;
  z-index: 9000; }

.search-wrap {
  height: 3.125rem;
  text-align: left; }

.search-group {
  -webkit-animation: slide-right 500ms cubic-bezier(0.65, -1, 0.25, 1.75); }

.search-group input[type="search"] {
  -webkit-appearance: none;
  background: url(../images/search.svg) no-repeat;
  background-position: 1rem 1rem;
  background-size: 1.3rem;
  border: none;
  font: 400 1.35rem 'Source Sans Pro', sans-serif;
  height: 3.125rem;
  padding-left: 3rem;
  text-align: left;
  width: 100%; }

.search-group input[type="search"]:focus {
  outline: none; }

input[type="search"]::-webkit-search-cancel-button {
  -webkit-appearance: none;
  background: white url(../images/delete.svg) center no-repeat;
  cursor: pointer;
  height: 3.12rem;
  left: 0;
```

Figure 12: Atom development environment interface

3.2.4. Postman testing software

Since the software solution of the projected system provides for the availability and development of an application software interface API, sending many requests, there is a need to use software for testing for such tasks. Postman was chosen as such software – a program that is a good tool for testing ready-made or personally created APIs. The main number of necessary functions in this software is free of charge. The program provides a user-friendly interface with extensive functionality that allows you to conduct testing over the API, namely, send HTTP requests without first writing program code. The principle of operation in Postman is quite simple: type the link to which the request will be sent, select a method, fill out the form or headers if necessary, and send a proposal that will return the status code, response time, and response from the server with detailed data. It is also possible to install monitoring [39]. The program interface is shown in Fig. 13.



```
{
    "token_type": "Bearer",
    "expires_in": "3599",
    "ext_expires_in": "0",
    "expires_on": "1531792714",
    "not_before": "1531788814",
    "resource": "https://management.core.windows.net/",
    "access_token":
        "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6IlRpb0d5d3dsaHZk
        RmJYWjgxM1dwUGF5OUFsVSIsImtpZCI6IlRpb0d5d3dsaHZkRmJYWjgxM1dwU
        GF5OUFsVSJ9
```

Figure 13: Postman program interface

3.2.5. Technical characteristics of the selected development software tools

The JavaScript programming language and Node.js platform were chosen to develop a software solution for the arbitrage search information system [40-42], which is the most optimal choice based on the analysis performed. The system will be implemented as a web application with a client-server architecture. Asynchronous code execution capability, event-driven platform environment, high performance in real-time client-server communication tasks, and support for node platform multithreading.js make it the best choice for software implementation of the system. A significant advantage is also the popularity of the language and the size of its community, which ensures further updates and the availability of a large number of ready-made solutions in the public domain [43-47]. The following main modules were selected to work with [48-55]:

- express – to create an application software interface and provide communication between the client and server;
- request – for sending and processing submissions by the server part of the system;
- fs – for working and manipulating operating system files.

Atom, a free and open-source text editor from GitHub with the ability to install additional packages, was chosen as the software development environment. This program provides all the necessary functionality for developing and partially debugging the server component of the system, the ability to auto-complete code, easy navigation through the project, creating web pages and viewing their changes in real-time, and so on. To conduct testing on personally designed application interfaces of the program and check sending requests to existing third-party APIs, The Postman program was chosen, the free functionality of which fully copes with the tasks set. In this chapter, comparative analysis and chapter of the programming language for its further application in creating a software solution of the system were carried out. The type of architecture of the projected system was selected – a client-server web application. The necessary modules and frameworks for the chosen programming language that will be used during development are established. Finally, the development environment is selected, and its primary functions and advantages are described.

4. Experiments, results and discussion

4.1. Description of the created software application

The software solution of the information system for searching for arbitrage situations in the cryptocurrency market is called "arbitrage". The source code of the developed software is contained in a folder with the appropriate name. The project is written in JavaScript and runs on the Node.js platform v8. 11. 3. the operating system used in the test operation is Microsoft Windows [Version 10.0.17134.407]. The main tasks of the developed software are to collect, analyze and process data from a wide range of exchanges, identify arbitrage opportunities based on the study of pre-generated data, and conduct arbitrage transactions. Thus, the created system is both informational and managerial: the existing functionality provides tools for searching and selecting arbitrage situations in the cryptocurrency market for further analysis and can automate such transactions with the user's approval.

The software solution is presented in a web system written in JavaScript and running on the Node.js platform using several frameworks and libraries. Partial functionality and system management are also available via the Telegram messenger, with the appropriate user settings.

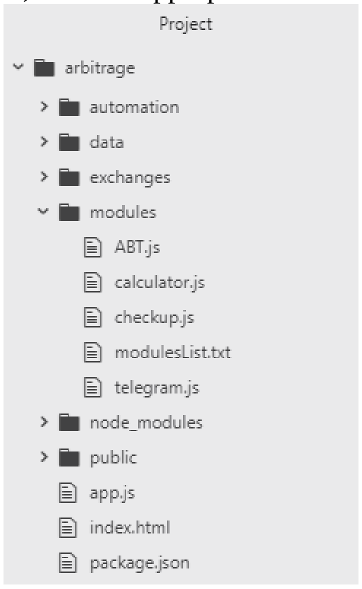


Figure 14: Project structure

The software structure looks like this:

- arbitrage – the project's root folder contains all the necessary components of the developed software;
- automation – a folder containing all the files required to ensure the operation of the module for automating arbitrage transactions (methods for working with private APIs of exchanges, program instructions for conducting arbitrage transactions, a file with the values of API keys from the user's exchange accounts);
- data – folder with saved data (results of specific processes), which are output, in JSON format;
- exchanges – this folder contains definitions of exchange objects and their methods necessary for data collection, processing, analysis, and generation;
- modules – folder containing self-written modules for system operation;
- node_modules – folder containing libraries and frameworks for system operation;
- public – a folder containing Cascading Style Sheet files, HyperText Markup files, fonts, and JavaScript scripts to enable client-server interaction;
- app.js – the main file that is the entry point of the system, which is used to launch it;

- index.html – HyperText Markup file for the system home page;
- package.json – a file containing metadata about the project: application name, description, version, list of dependent modules, and so on.

The developed system can be divided into four main components::

- Data collection and processing component;
- Analytical and computational part;
- Arbitrage transaction component;
- GUI and user interaction component.

The first component is responsible for receiving, analyzing, and processing data coming from exchanges. The result of this component is filling exchange objects with data. This process for each sale starts with executing the Exchange.build() method.

This method sends a certain number of requests (on average from 2 to 5) for each object (exchange) to the servers of the related businesses to receive data. This data includes the values of the exchange's trading commissions, available markets for trading, a list of available cryptocurrency pairs, prices, technical data on the specifics of depositing and withdrawing cryptocurrencies from exchanges, and so on. After successful receipt of the response, i.e. arrays or objects of the required data, they are converted to JSON format (if necessary) and converted to a general structured form, which is the same for all exchanges. Example of the object structure of one of the exchanges:

```
{  "name": "Bittrex",
   "url": "https://bittrex.com/",
   "apiPublicURL": "https://bittrex.com/api/v1.1/public/",
   "marketURL": "https://bittrex.com/Market/Index?MarketName=",
   "tradeFee": 0.25,
   "minTradeSize": 0.0005,
   "quotes": [ "BTC", "ETH", "USDT", "USD"],
   "assets": [],
   "exceptions": { "BITS": "Bitswift", "MOC": "Mossland", "SLT": "Smartlands", "GTO": "GTO_ERC-20" },
   "disabledWallets": [ "EMC2", "GRIN", "IOTA" ],
   "disabledMarkets": [ "BTC-GRIN", "USDT-GRIN", "BTC-CPT*" ],
   "timeStamp": "2020-03-07T16:02:00.870Z",
   "readinessStatus": true,
   "requests": 3,
   "responses": 3,
   "responsesData": {}
}
```

The "assets" property, exchange objects, contains an array of things filled with data about the technical features of a particular cryptocurrency on the corresponding exchange. Namely: ticker, the full name of the currency, the number of confirmations in the network for crediting the deposit, the value of the withdrawal Commission, minimum deposit, the minimum and maximum amount for withdrawal, data on the operability of the withdrawal or deposit of cryptocurrency, particular information messages, a list of pairs against which the cryptocurrency is traded, the purchase and sale price and the last operation performed, links to trades, etc. The structure of such objects looks like this:

```
{  "id": "LTC",
   "ticker": "LTC",
   "name": "Litecoin",
   "depositConfirmations": 4,
   "txFee": 0.002,
   "minDeposit": 0,
   "withdrawalPrecision": 8,
   "minWithdrawal": 0.004,
   "maxWithdrawal": 150,
   "depositsEnabled": true,
   "withdrawalsEnabled": true,
   "markets": {
      "BTC": {"isTraded": true, "marketName": "LTC-BTC",
            "marketURL": "https://crex24.com/exchange/LTC-BTC",  "isActive": true,
            "timeStamp": "2020-03-07T16:02:04Z", "ask": 0.00688, "bid": 0.00687, "last": 0.00688,
            "orderBook": { "buy": [],  "sell": [] }, "minTradeSizeBase": 0.001 },
      "USD": { "isTraded": false, "orderBook": { "buy": [], "sell": [] }
      }
    }
  }
```

The second analytical and computational component is necessary for analyzing the data obtained due to the operation of the first component. Analysis of this data involves searching for similar cryptocurrencies and their trading pairs on different exchanges. After this search, a particular cryptocurrency pair's purchase and sale prices are compared on the deals where matches were found. If a higher purchase price and a lower sale price are detected on different exchanges, the system performs the following action – the so-called "in-depth" analysis. This method sends two requests for detailed data on buy and sells orders on the two corresponding exchanges, containing the price and volume of transactions. After receiving and processing this data, the calculation process is carried out – the result is information about the maximum amount of cryptocurrency that can be purchased on the exchange, how much quota currency is needed, potential profit, and so on. The results are displayed on the main page of the web system. In addition, they can be sent to the Telegram messenger immediately after the analysis, provided that the system user has the appropriate settings. An object that contains data on the results of calculations and performing several studies has the following structure:

```
{  "id": 1588999027620
   "done": true,
   "success": true,
   "responses": 2,
   "currentResult": {
      "changePercents": "2.82",
      "exchangeToBuy": {
         "name": "Bittrex",
         "asset": { "ticker": "SOLVE", "name": "Solve.Care", "status": "Online", "txFee": 1.25 },
         "market": { "URL": "https://bittrex.com/Market/Index?MarketName=BTC-SOLVE",
                     "ask": "0.00000888", "base": "BTC", "status": "Online" }
      },
      "exchangeToSell": {
         "name": "HitBTC",
         "asset": { "ticker": "SOLVE", "name": "Solve", "status": "Online", "depositConfirmations": 20 },
         "market": { "URL": "https://hitbtc.com/exchange/SOLVE-to-BTC",
                     "bid": "0.00000917", "base": "BTC", "status": "Unknown" }
      }
   },
   "profitByVolume": [  {
         "quoteQuantity": 0.09519300000007454, "quoteQuantityInUSD": 685.3896000005367,
         "assetQuantity": 10689.786877642566,    "assetQuantityReceived": 10688.536877642566,
         "quoteQuantityReceived": 0.0978830878942926,   "profitQuote": 0.0026900878942180684,
         "profitUSD": 19.368632838370093,  "profitPercentage": 2.825930366955518,
         "firstPrice": 0.00000888, "lastPrice": 0.00000892
      },
      {
         "quoteQuantity": 0.07037200000004971,  "quoteQuantityInUSD": 506.67840000035795,
         "assetQuantity": 7908.925225230812,  "assetQuantityReceived": 7907.675225230812,
         "quoteQuantityReceived": 0.07242185464070117,  "profitQuote": 0.0020498546406514523,
         "profitUSD": 14.758953412690456, "profitPercentage": 2.912883875191838,
         "firstPrice": 0.00000888,  "lastPrice": 0.00000888
      }
   ]
}
```

The arbitrage transaction component automates the sequence of actions required to eliminate the price difference between two exchanges. The input data of this component is the results of calculations and analysis obtained as a result of the operation of the previous element. This data provides instructions on how to perform the arbitrage transaction module. The process of conducting an arbitration transaction is started exclusively by the system user. On the main page of the web system, in the section with the found arbitrage capabilities, there is an interface for conducting a transaction. The user "gives the go-ahead" to complete the selected transaction and specifies the desired volume. Starting this process, with the appropriate settings, is also available via the Telegram messenger: the user receives a notification about finding an arbitration situation with an interface that provides the ability to "respond": start the execution process, or they ignore a specific arbitration situation for a selected period. This component performs several actions that are necessary for a full-fledged arbitration transaction, namely: checking balances on exchange accounts, obtaining an address for depositing cryptocurrency on an exchange with a higher price, creating and cancelling purchase and sale orders, sending cryptocurrencies, transferring funds between accounts of one exchange account, conducting interim settlements, and so on. Upon completion of an arbitration transaction, the object with all data on its conduct has the following structure:

```
{   "id": 1589147511273,
    "code": "STEX:PART/BTC-HitBTC:PART/BTC",
    "stage": "SELL_ORDER_CALCULATED",
    "time": { "start": "2020-05-10T21:56:27.074Z",  "end": "2020-05-10T22:07:44.356Z" },
    "exchangeToBuy": {  "name": "STEX",   "balances": {
        "quote": { "ticker": "BTC", "name": null,  "beforeBuy": 0.09863381, "afterBuy": 0.09243302,
                  "spent": 0.006200789999999998  },
        "base":{"ticker":"PART","name":"Particl","beforeBuy":0,"afterBuy":154.01681657,
                "bought": 154.01681657, }
      },
      "order": { "id": 239086323,  "filled": true, "canceled": false },
      "withdrawal": { "id": 1303657, "success": true }
    },
    "exchangeToSell": {  "name": "HitBTC",  "balances": {  "trade": {
          "quote": { "beforeSell": 0.09769798088, "afterSell": 0.10413836945 },
          "base": { "beforeTransfer": 0, "afterTransfer": 154.00681657, "transferred": 154.00681657,
                  "beforeSell": 154.00681657, "afterSell": 0 }
        },
        "main": { "base": { "beforeDeposit": 0, "afterDeposit": 154.00681657 } },
        "quote": { "ticker": "BTC", "name": null, "beforeSell": 0.09769798088,
                  "afterSell": 0.10413836945, "received": 0.006440388570000011 },
        "base": { "ticker": "PART", "name": "Particl", "depositAddress": {
            "address": "Pe1747PokyVHULxz7Tecy29Kog5KtsESmg", "paymentIdRequired": false,
            "addressGenerationRequired": false, "generationRequestSent": false, "addressGenerated": false},
          "beforeDeposit": 0, "afterDeposit": 154.00681657, "received": 154.00681657,
          "transferred": 154.00681657, "afterSell": 0.00681657, "sold": 154.00681657 }
      },
      "transfer": { "id": "50ee9ed5-660e-4654-843c-7ee862bf8b9b", "success": true },
      "order": { "id": "fda3563905d917f5ebbf9a7b5bda02ca", "filled": true, "canceled": false }
    },
}
```

The last component concerns the GUI and user interaction. It contains instructions on how the client part interacts with the server part. Implemented functionality is for data transfer and system management using created APIs. The developed software product is intended for use on servers or personal computers. The software will run on all operating systems that support the Node platform.js (Linux, Windows, Mac, and others). Since the software tool sends requests for information at very short intervals and is faced with processing a large amount of data, the system's speed on which the software will run plays a significant role. Recommended technical requirements:

- Processor: 4 GHz;
- Amount of RAM: 16 GB;

- The presence or absence of a graphics module does not affect the operation of the system;
- Free disk space: 32 GB;
- Internet connection: 50 Mbps and low latency.

The source files of the system software are located in the "arbitrage" folder. Before launching the software solution of the information system for searching for arbitrage situations in the cryptocurrency market, you must install the Nod.js platform version 8.11.3 or later. To start the system, go to the project's root folder and in the console and run the command " node app.js». At the first launch, all the necessary modules will be installed to work (if required). After that, it will be launched.

There are no system inputs because the entire sequence of actions is set in the system's program code, and other processes are automated. Therefore, only API keys from exchanges and the created bot in the Telegram Messenger are input data. The user must generate them on the corresponding sales in the Settings section to get these keys, granting certain access rights necessary to execute an arbitrage transaction successfully. Also, the input data can be System Settings-Filters for searching for arbitrage situations, which the user sets. The source data in the program are messages received in the Telegram messenger and located on the main page of the system, exchange objects, copies of which are stored in a file containing the results of calculations and the results of an arbitration transaction. In addition, files are stored in the data folder with specific system settings.

4.2. User manual

The developed software product is designed to find and perform arbitrage opportunities in the cryptocurrency market among several popular exchanges.

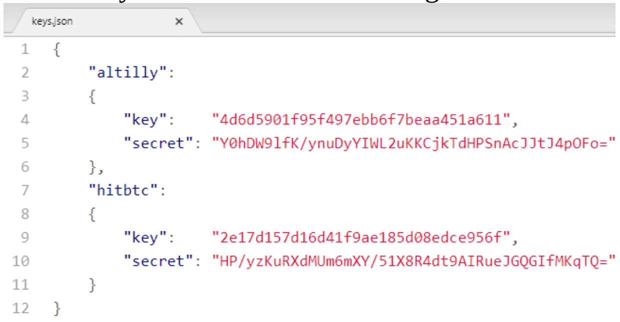
The created application was called "arbitrage" and written in JavaScript running on the Node.js platform. Several popular frameworks are used, including request, express, fs, and others. The system was created using the Atom development environment. The developed software implementation of the program can be used as an information system for searching and analyzing arbitrage opportunities in the cryptocurrency market among a wide range of exchanges and as a management system – for automating and monitoring or reporting on arbitrage transactions. To use this system, namely the component for automating arbitrage transactions, you must provide API keys from exchanges with permission to conduct trading, check balances, generate and receive addresses for depositing cryptocurrency, and consent to withdraw funds from the sale. You can also manage the system using third – party software-The Telegram messenger, after creating a bot in it and providing the system with its API key. To understand the ongoing processes and manage the system, you need basic knowledge about the specifics of the principles of functioning of cryptocurrencies and a minimal understanding of the operation of trading markets in general. The software runs on such operating systems as Windows, Linux, Mac and all others that support installing the Node.js platform.

The created software solves several tasks, resulting in identifying arbitrage situations in the cryptocurrency market among a certain number of exchanges, and it is also possible to automatically conduct such transactions. After the user starts the system, the program makes a series of requests to the servers of exchanges to collect the necessary information, which will be used in the formation of a structured data set that will contain information about price changes in the cryptocurrency market on each of the exchanges presented in the system, technical data on the features of cryptocurrencies on these exchanges, and so on. Then, based on the previously described data, they are analyzed by one of the system components, which is the identification of arbitrage opportunities. When the user agrees, the system executes the found arbitrage transactions and notifies the user about their progress, from start to finish, in a detailed form by sending informational notifications. Special features of the program are that it can work in two modes:

- As an information system;
- As an information and management system.

In the first case, the system works purely as an information system when the user does not provide API keys from exchanges. Then the software only provides the user with the ability to view the found arbitrage opportunities and other data that can be used for any purpose. In the second case, the system provides the same functions as in the first case. Still, if you provide API keys from exchanges, the user can automatically conduct arbitrage transactions, thus managing the process and performing operations on their exchange accounts. Furthermore, the system checks whether actions are performed correctly by the system itself. If an error occurs, the system user will receive a detailed information message with its description. The user himself can practically not influence the program's correct operation in any way since the input data and certain functionality do not affect the principles of operation of the system.

To use this software, it's needed a device running on Windows, Linux, Mac, or any other operating system that supports the Node.js platform. As for the technical requirements, for the productive functioning of the system, it is necessary to have a device with a sufficient amount of RAM and a modern processor to ensure fast calculation of operations, processing and analysis of information. In addition, it is essential to have high-speed internet access, without which the system will not receive data and find solutions to tasks and will generally be useless. Before starting the system itself, if you use a module that automates arbitrage transactions, the user must provide the system with API keys to their exchange accounts with permissions to conduct a certain number of operations (the ability to conduct trades, receive account information, withdraw cryptocurrencies) and an API key from the previous created bot in the Telegram messenger, for the possibility of notifications and remote management of the system. API keys are stored in the arbitrage/automation/keys file.json (Fig. 15).

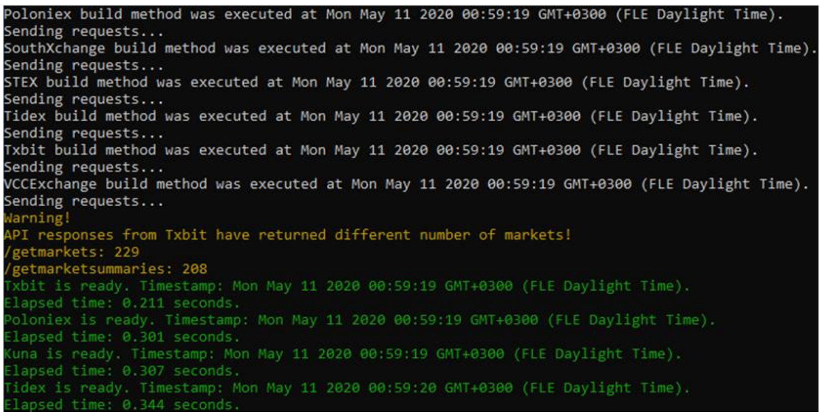


```
{
    "altilly":
    {
        "key":    "4d6d5901f95f497ebb6f7beaa451a611",
        "secret": "Y0hDW9lfK/ynuDyYIWL2uKKCjkTdHPSnAcJJtJ4pOFo="
    },
    "hitbtc":
    {
        "key":    "2e17d157d16d41f9ae185d08edce956f",
        "secret": "HP/yzKuRXdMUm6mXY/51X8R4dt9AIRueJGQGIfMKqTQ="
    }
}
```

Figure 15: Structure of the file containing API keys

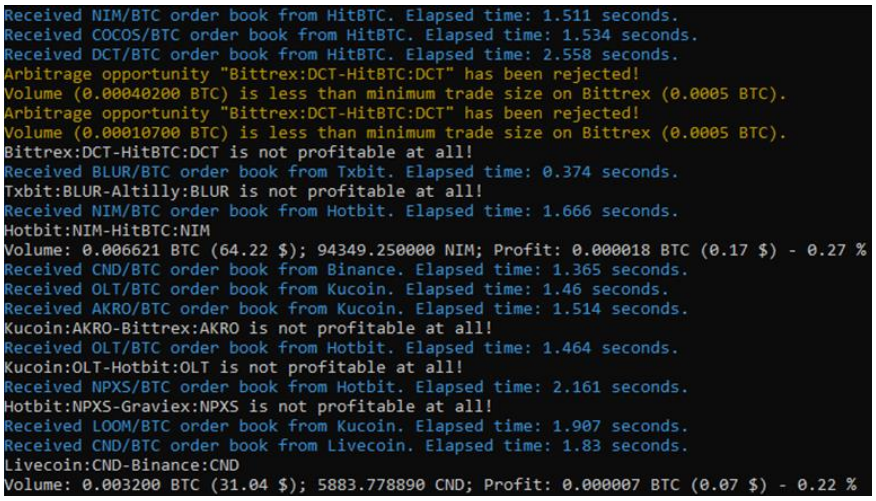
Next, it's needed to install the Node.js platform version 8.11.3 or later. Then, the system itself is started using the console. It's required to go to the project's root folder (arbitrage) and run the command "node app.js". If necessary, the system will load all the necessary modules for its functioning at the first launch. After their successful installation, the system itself starts working directly.

Immediately after its launch, the system sends requests (Fig. 16) to the exchanges' servers to obtain the necessary data and further analysis and processing. The corresponding objects of sales will be generated. After that, the data contained in the exchange objects are analyzed, and price differences are searched. The next step is to run the analysis "in-depth" and generate the results of the work. The output of messages about the application of this analysis is shown in Fig. 17.



```
Poloniex build method was executed at Mon May 11 2020 00:59:19 GMT+0300 (FLE Daylight Time).
Sending requests...
SouthXchange build method was executed at Mon May 11 2020 00:59:19 GMT+0300 (FLE Daylight Time).
Sending requests...
STEX build method was executed at Mon May 11 2020 00:59:19 GMT+0300 (FLE Daylight Time).
Sending requests...
Tidex build method was executed at Mon May 11 2020 00:59:19 GMT+0300 (FLE Daylight Time).
Sending requests...
Txbit build method was executed at Mon May 11 2020 00:59:19 GMT+0300 (FLE Daylight Time).
Sending requests...
VCCExchange build method was executed at Mon May 11 2020 00:59:19 GMT+0300 (FLE Daylight Time).
Sending requests...
Warning!
API responses from Txbit have returned different number of markets!
/getmarkets: 229
/getmarketsummaries: 208
Txbit is ready. Timestamp: Mon May 11 2020 00:59:19 GMT+0300 (FLE Daylight Time).
Elapsed time: 0.211 seconds.
Poloniex is ready. Timestamp: Mon May 11 2020 00:59:19 GMT+0300 (FLE Daylight Time).
Elapsed time: 0.301 seconds.
Kuna is ready. Timestamp: Mon May 11 2020 00:59:19 GMT+0300 (FLE Daylight Time).
Elapsed time: 0.307 seconds.
Tidex is ready. Timestamp: Mon May 11 2020 00:59:20 GMT+0300 (FLE Daylight Time).
Elapsed time: 0.344 seconds.
```

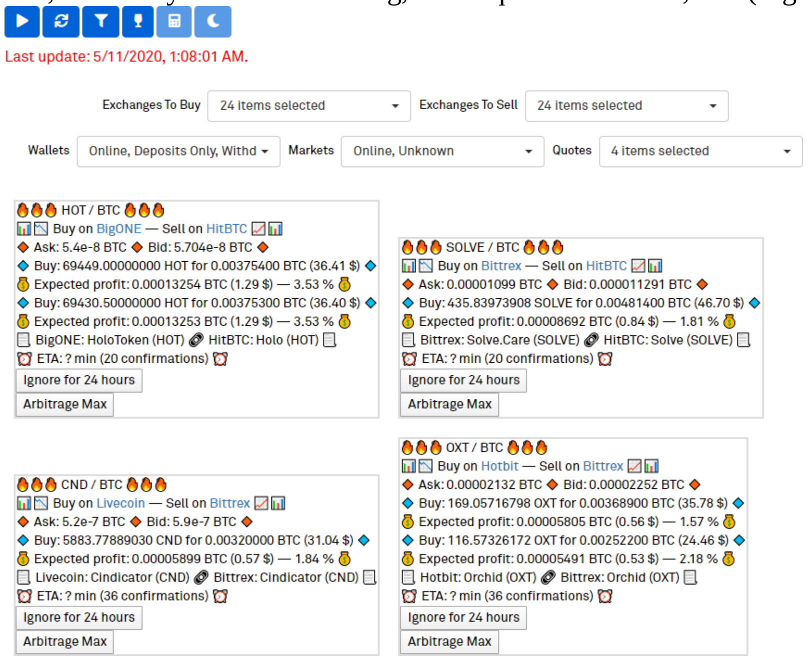
Figure 16: Sending requests by the system



```
Received NIM/BTC order book from HitBTC. Elapsed time: 1.511 seconds.
Received COCOS/BTC order book from HitBTC. Elapsed time: 1.534 seconds.
Received DCT/BTC order book from HitBTC. Elapsed time: 2.558 seconds.
Arbitrage opportunity "Bittrex:DCT-HitBTC:DCT" has been rejected!
Volume (0.00040200 BTC) is less than minimum trade size on Bittrex (0.0005 BTC).
Arbitrage opportunity "Bittrex:DCT-HitBTC:DCT" has been rejected!
Volume (0.00010700 BTC) is less than minimum trade size on Bittrex (0.0005 BTC).
Bittrex:DCT-HitBTC:DCT is not profitable at all!
Received BLUR/BTC order book from Txbit. Elapsed time: 0.374 seconds.
Txbit:BLUR-Altilly:BLUR is not profitable at all!
Received NIM/BTC order book from Hotbit. Elapsed time: 1.666 seconds.
Hotbit:NIM-HitBTC:NIM
Volume: 0.006621 BTC (64.22 $); 94349.250000 NIM; Profit: 0.000018 BTC (0.17 $) - 0.27 %
Received CND/BTC order book from Binance. Elapsed time: 1.365 seconds.
Received OLT/BTC order book from Kucoin. Elapsed time: 1.46 seconds.
Received AKRO/BTC order book from Kucoin. Elapsed time: 1.514 seconds.
Kucoin:AKRO-Bittrex:AKRO is not profitable at all!
Received OLT/BTC order book from Hotbit. Elapsed time: 1.464 seconds.
Kucoin:OLT-Hotbit:OLT is not profitable at all!
Received NPXS/BTC order book from Hotbit. Elapsed time: 2.161 seconds.
Hotbit:NPXS-Graviex:NPXS is not profitable at all!
Received LOOM/BTC order book from Kucoin. Elapsed time: 1.907 seconds.
Received CND/BTC order book from Livecoin. Elapsed time: 1.83 seconds.
Livecoin:CND-Binance:CND
Volume: 0.003200 BTC (31.04 $); 5883.778890 CND; Profit: 0.000007 BTC (0.07 $) - 0.22 %
```

Figure 17: Performing deep analysis by the system

The results are presented on the main page of the system, located at 127.0.0.1:3000 (Fig. 18). On the main page, the user can disable or enable periodic data updates, hide the result of calculations "in-depth", and set filters: select a list of exchanges for buying and selling, set the status of cryptocurrency wallets of businesses, the ability to conduct trading, select quota currencies, etc. (Fig. 19).



Last update: 5/11/2020, 1:08:01 AM.

Exchanges To Buy: 24 items selected
Exchanges To Sell: 24 items selected
Wallets: Online, Deposits Only, Withd
Markets: Online, Unknown
Quotes: 4 items selected

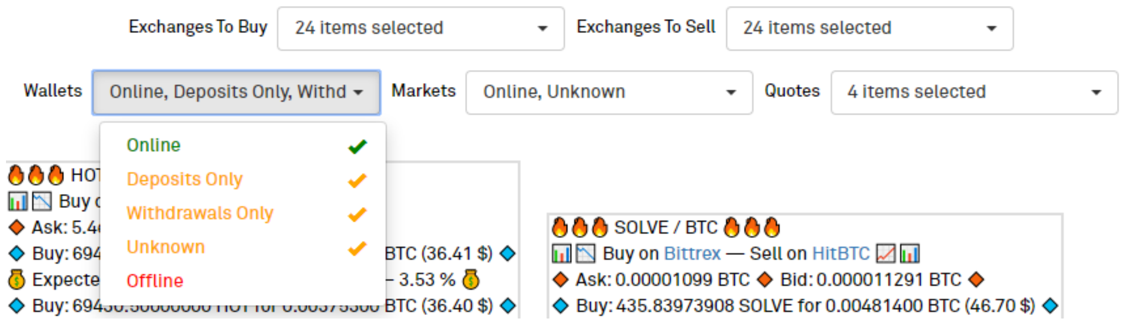
🔥🔥🔥 HOT / BTC 🔥🔥🔥
Buy on BigONE — Sell on HitBTC
◆ Ask: 5.4e-8 BTC ◆ Bid: 5.704e-8 BTC ◆
◆ Buy: 69449.00000000 HOT for 0.00375400 BTC (36.41 $) ◆
💰 Expected profit: 0.00013254 BTC (1.29 $) — 3.53 % 💰
◆ Buy: 69430.50000000 HOT for 0.00375300 BTC (36.40 $) ◆
💰 Expected profit: 0.00013253 BTC (1.29 $) — 3.53 % 💰
BigONE: HoloToken (HOT) 🔗 HitBTC: Holo (HOT)
⏰ ETA: ? min (20 confirmations) ⏰
Ignore for 24 hours
Arbitrage Max

🔥🔥🔥 SOLVE / BTC 🔥🔥🔥
Buy on Bittrex — Sell on HitBTC
◆ Ask: 0.00001099 BTC ◆ Bid: 0.000011291 BTC ◆
◆ Buy: 435.83973908 SOLVE for 0.00481400 BTC (46.70 $) ◆
💰 Expected profit: 0.00008692 BTC (0.84 $) — 1.81 % 💰
Bittrex: Solve.Care (SOLVE) 🔗 HitBTC: Solve (SOLVE)
⏰ ETA: ? min (20 confirmations) ⏰
Ignore for 24 hours
Arbitrage Max

🔥🔥🔥 CND / BTC 🔥🔥🔥
Buy on Livecoin — Sell on Bittrex
◆ Ask: 5.2e-7 BTC ◆ Bid: 5.9e-7 BTC ◆
◆ Buy: 5883.77889030 CND for 0.00320000 BTC (31.04 $) ◆
💰 Expected profit: 0.00005899 BTC (0.57 $) — 1.84 % 💰
Livecoin: Cindicator (CND) 🔗 Bittrex: Cindicator (CND)
⏰ ETA: ? min (36 confirmations) ⏰
Ignore for 24 hours
Arbitrage Max

🔥🔥🔥 OXT / BTC 🔥🔥🔥
Buy on Hotbit — Sell on Bittrex
◆ Ask: 0.00002132 BTC ◆ Bid: 0.00002252 BTC ◆
◆ Buy: 169.05716798 OXT for 0.00368900 BTC (35.78 $) ◆
💰 Expected profit: 0.00005805 BTC (0.56 $) — 1.57 % 💰
◆ Buy: 116.57326172 OXT for 0.00252200 BTC (24.46 $) ◆
💰 Expected profit: 0.00005491 BTC (0.53 $) — 2.18 % 💰
Hotbit: Orchid (OXT) 🔗 Bittrex: Orchid (OXT)
⏰ ETA: ? min (36 confirmations) ⏰
Ignore for 24 hours
Arbitrage Max

Figure 18: System home page

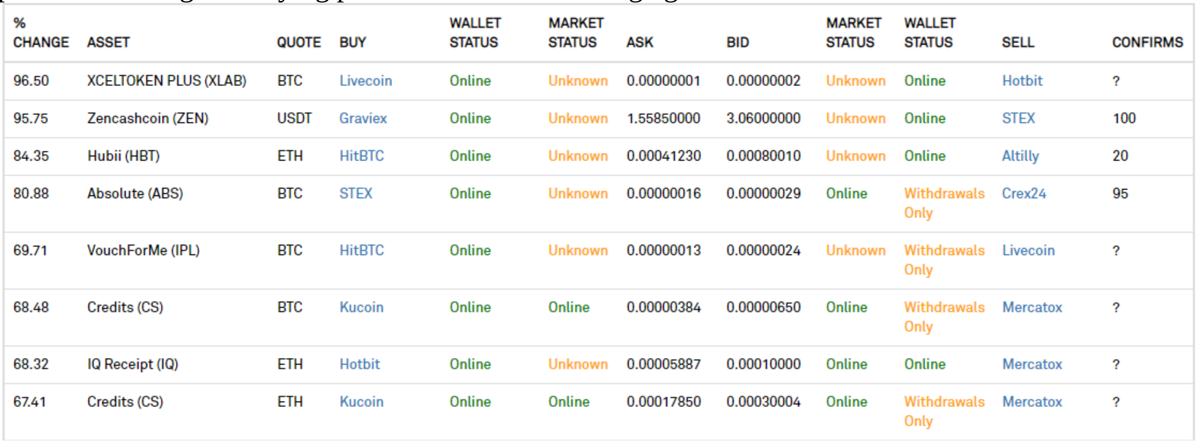


Exchanges To Buy: 24 items selected
Exchanges To Sell: 24 items selected
Wallets: Online, Deposits Only, Withd
Markets: Online, Unknown
Quotes: 4 items selected

🔥🔥🔥 SOLVE / BTC 🔥🔥🔥
Buy on Bittrex — Sell on HitBTC
◆ Ask: 0.00001099 BTC ◆ Bid: 0.000011291 BTC ◆
◆ Buy: 435.83973908 SOLVE for 0.00481400 BTC (46.70 $) ◆

Figure 19: Filters for selecting results (the wallet status filter is open)

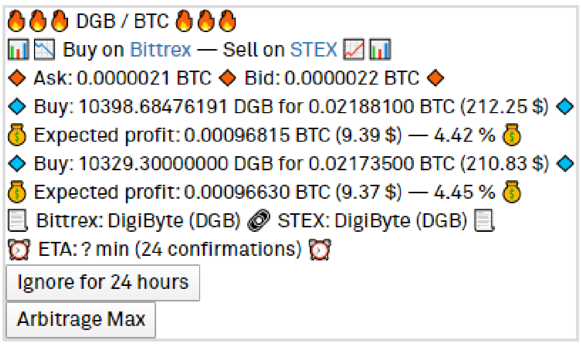
Below on the page, there is an arbitration table (Fig. 20). It provides data on the found arbitrage situations, without analyzing "in-depth", without considering the volumes, but only the lowest selling price and the highest buying price in the stock exchange glass.



% CHANGE	ASSET	QUOTE	BUY	WALLET STATUS	MARKET STATUS	ASK	BID	MARKET STATUS	WALLET STATUS	SELL	CONFIRMS
96.50	XCELTOKEN PLUS (XLAB)	BTC	Livecoin	Online	Unknown	0.00000001	0.00000002	Unknown	Online	Hotbit	?
95.75	Zencashcoin (ZEN)	USDT	Graviex	Online	Unknown	1.55850000	3.06000000	Unknown	Online	STEX	100
84.35	Hubii (HBT)	ETH	HitBTC	Online	Unknown	0.00041230	0.00080010	Unknown	Online	Altilly	20
80.88	Absolute (ABS)	BTC	STEX	Online	Unknown	0.00000016	0.00000029	Online	Withdrawals Only	Crex24	95
69.71	VouchForMe (IPL)	BTC	HitBTC	Online	Unknown	0.00000013	0.00000024	Unknown	Withdrawals Only	Livecoin	?
68.48	Credits (CS)	BTC	Kucoin	Online	Online	0.00000384	0.00000650	Online	Withdrawals Only	Mercatox	?
68.32	IQ Receipt (IQ)	ETH	Hotbit	Online	Unknown	0.00005887	0.00010000	Online	Online	Mercatox	?
67.41	Credits (CS)	ETH	Kucoin	Online	Online	0.00017850	0.00030004	Online	Withdrawals Only	Mercatox	?

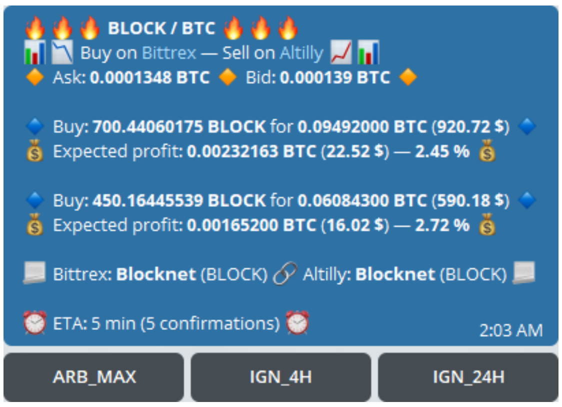
Figure 20: Arbitration table

The type of presentation of data on the found arbitration situations with "in-depth" analysis is shown in Fig. 21.



🔥🔥🔥 DGB / BTC 🔥🔥🔥
📊📉 Buy on Bittrex — Sell on STEX 📈📊
🔶 Ask: 0.0000021 BTC 🔶 Bid: 0.0000022 BTC 🔶
🔷 Buy: 10398.68476191 DGB for 0.02188100 BTC (212.25 $) 🔷
💰 Expected profit: 0.00096815 BTC (9.39 $) — 4.42 % 💰
🔷 Buy: 10329.30000000 DGB for 0.02173500 BTC (210.83 $) 🔷
💰 Expected profit: 0.00096630 BTC (9.37 $) — 4.45 % 💰
📃 Bittrex: DigiByte (DGB) 🔗 STEX: DigiByte (DGB) 📃
⏰ ETA: ? min (24 confirmations) ⏰

Figure 21: Search result



🔥🔥🔥 **BLOCK / BTC** 🔥🔥🔥
📊📉 Buy on Bittrex — Sell on Altilly 📈📊
🔶 Ask: **0.0001348 BTC** 🔶 Bid: **0.000139 BTC** 🔶

🔷 Buy: **700.44060175 BLOCK** for **0.09492000 BTC (920.72 $)** 🔷
💰 Expected profit: **0.00232163 BTC (22.52 $) — 2.45 %** 💰

🔷 Buy: **450.16445539 BLOCK** for **0.06084300 BTC (590.18 $)** 🔷
💰 Expected profit: **0.00165200 BTC (16.02 $) — 2.72 %** 💰

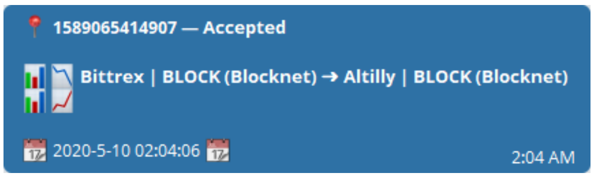
📃 Bittrex: **Blocknet** (BLOCK) 🔗 Altilly: **Blocknet** (BLOCK) 📃

⏰ ETA: 5 min (5 confirmations) ⏰ 2:03 AM

Figure 22: Telegram Messenger interface

Such a message consists of: the name of the cryptocurrency pair, the names of exchanges for buying and selling, the terms of cryptocurrencies on various exchanges and their tickers, a direct link to trading on these exchanges, the volume that can be purchased, the amount of purchase and sale, potential profit, the approximate time of the transaction and the number of necessary confirmations in the cryptocurrency network required by the exchange, where it is transferred, and so on. When you provide an API key to the bot in the Telegram messenger, the user immediately receives notifications about finding an arbitration situation based on pre-set criteria. The message being sent (Fig. 22) is identical to the information provided on the system's main page. Sending notifications to the messenger eliminates the need for the user to constantly be at the monitor screen to wait for the arbitrage transaction to be found. Both on the system's web page and with the help of a bot in the Telegram messenger, the user can conduct an arbitration transaction. It is done by clicking the corresponding button located under the message. You can also ignore a specific transaction. For one reason or another, an arbitrage opportunity with the specified parameters will be missed by searching for the selected period.

Provided that the user has decided to conduct an arbitration transaction and clicked on the corresponding button, the messenger receives a message about its successful acceptance for processing by the system (Fig. 23). The following notification to the user is a notification about the volume of purchased cryptocurrency and the quota currency spent (Fig. 24).

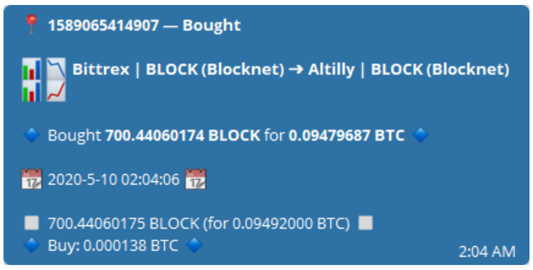

1589065414907 — Accepted

Bittrex | BLOCK (Blocknet) → Altilly | BLOCK (Blocknet)

2020-5-10 02:04:06

2:04 AM

Figure 23: Notification of acceptance


1589065414907 — Bought

Bittrex | BLOCK (Blocknet) → Altilly | BLOCK (Blocknet)

Bought **700.44060174 BLOCK** for **0.09479687 BTC**

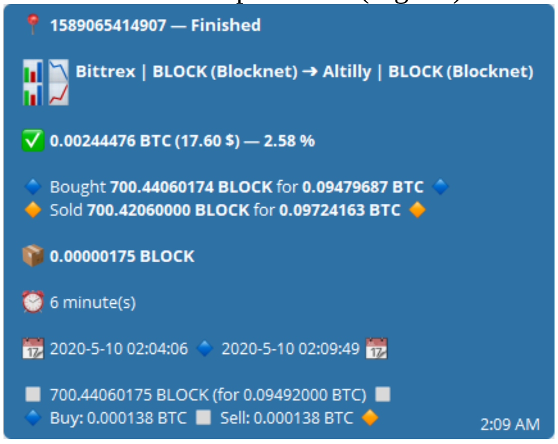
2020-5-10 02:04:06

700.44060175 BLOCK (for 0.09492000 BTC)
Buy: 0.000138 BTC

2:04 AM

Figure 24: Notification of purchased cryptocurrency

The last notification is the result of an arbitrage transaction (Fig. 25). The message consists of the following data: transaction ID, name of the cryptocurrency, its ticker, terms of exchanges for buying and selling, realized profit, conversion of realized profit to the dollar (at the current rate or rate specified in the system code), the percentage change in balance, amount of purchased cryptocurrency, amount of received quota, amount of cryptocurrency that was not sold, transaction time, initial parameters for buying and selling, etc. If an error occurs at any stage of the transaction, the user receives a message with a detailed description of it (Fig. 26).


1589065414907 — Finished

Bittrex | BLOCK (Blocknet) → Altilly | BLOCK (Blocknet)

✅ **0.00244476 BTC (17.60 $) — 2.58 %**

Bought **700.44060174 BLOCK** for **0.09479687 BTC**
Sold **700.42060000 BLOCK** for **0.09724163 BTC**

0.00000175 BLOCK

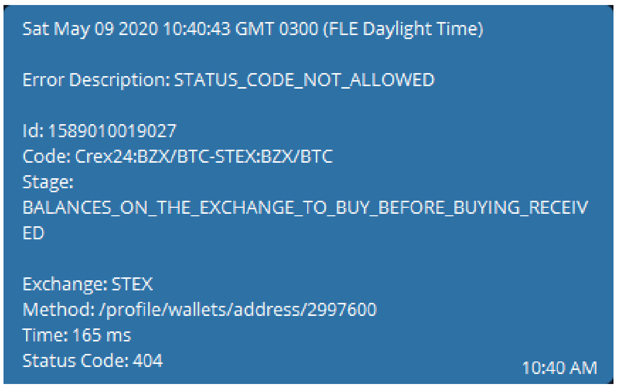
6 minute(s)

2020-5-10 02:04:06 2020-5-10 02:09:49

700.44060175 BLOCK (for 0.09492000 BTC)
Buy: 0.000138 BTC Sell: 0.000138 BTC

2:09 AM

Figure 25: Notification of the completion


Sat May 09 2020 10:40:43 GMT 0300 (FLE Daylight Time)

Error Description: STATUS_CODE_NOT_ALLOWED

Id: 1589010019027
Code: Crex24:BZX/BTC-STEX:BZX/BTC
Stage:
BALANCES_ON_THE_EXCHANGE_TO_BUY_BEFORE_BUYING_RECEIVED

Exchange: STEX
Method: /profile/wallets/address/2997600
Time: 165 ms
Status Code: 404

10:40 AM

Figure 26: Notification when an error occurs

You can terminate the program by pressing Ctrl + C in the console with the system running.

This chapter describes the developed software solution of the information system for searching for arbitrage situations in the cryptocurrency market. According to the requirements, the description of the software implementation is carried out according to the standard (GOST 19.402-78). The programming language used in system development, frameworks, development environment, and other technologies used in system development are specified. Technical requirements for the installation and proper functioning of the developed system are defined. A user manual has been created that contains the necessary information about the operation features of this software. The functionality of the developed system, principles and elements of operation, and its purpose are considered. The input and output files, contents, and the structure of intermediate data used by the system in the main processes are described. A control example of using the program is given, and the process of starting and installing the system is considered. The work aims to create an information system for finding arbitrage situations in the cryptocurrency market. The system's main task is to collect, process, and analyze data from a wide range of cryptocurrency exchanges. The software product under development aims to find arbitrage opportunities in the cryptocurrency market and access this data conveniently to the user. In addition to the search function, the system can perform a management function with the user's consent: conduct arbitrage transactions automatically and notify about their execution results. During its history, money has undergone many stages of change: once its role was played by random, today almost priceless items, and now we can observe how even cash is gradually disappearing from circulation, as contactless payments are gaining more and more popularity.

The rapid development of Information Technologies, which gave an impetus to drastic changes in the global economy, also showed its impact on the financial sector. A significant event, in particular, is the emergence of electronic money and cryptocurrencies.

Cryptocurrency is usually called a virtual currency that works using cryptography methods and blockchain technology. A characteristic feature of most cryptocurrencies is decentralization. To conduct transactions on the network, you do not need to contact a third party or a central management body, such as a bank, since the system ensures equal rights for all its participants. Due to the lack of a legal framework and the low degree of settlement by the state, cryptocurrency cannot be attributed to the classic concept of money in this sense. Although for almost ten years, cryptocurrency as an alternative payment method has been accepted by hundreds of institutions and services worldwide.

The first cryptocurrency was Bitcoin, which appeared in early 2010. Soon there were many other projects related to the development of cryptocurrencies. Today there are more than several thousand of them. This number of projects has also led to the rapid growth of unique trading platforms – crypto exchanges. The growing popularity of the market, a large number of businesses and projects, the lack of a legal component for proper settlement, the phenomenon of decentralization of cryptocurrencies, high volatility, etc. – all these reasons contribute to the emergence of Arbitration situations in the cryptocurrency market. Arbitrage refers to performing a sequence of actions to eliminate profit when one asset is traded on two different platforms at different prices. Arbitration in the cryptocurrency market, compared with traditional markets, does not differ in the method of conducting and is even more attractive in the emerging market. So, this topic is relevant today. Therefore, there is a need to create a system for finding arbitrage opportunities in the cryptocurrency market. The software solution will be helpful for both experienced traders and beginners in this field.

5. Conclusions

In this paper, research, project and development of a software solution for the information system for finding arbitrage situations in the cryptocurrency market were conducted.

The analysis of literature and other sources closely related to the subject area of the projected system was carried out. The concept of cryptocurrencies, features of their operation and aspects of functioning are revealed. The factors influencing pricing in the cryptocurrency market are described and evaluated. The idea of Arbitration and the basis of its methodology are considered, and examples are given. The features of conducting arbitration in the cryptocurrency market are also described. A comparative analysis of existing trading systems is carried out, their weaknesses and strengths are highlighted. A number of problems were found, the solution of which was presented in the designed system and the useful functionality of existing systems implemented during development was determined.

An information system for searching for arbitrage situations in the cryptocurrency market was designed by conducting a system analysis. One of the most critical methods of system analysis was used – the goal tree. Using the constructed tree, the system's primary goal, its sub-goals, and specific actions that must be performed to solve all tasks and implement the system in full successfully are defined. Criteria for assessing the system's quality are highlighted, with the help of which, and by applying the hierarchy analysis method, a comparison was made, and the most optimal type of information system was selected among the previously proposed alternatives. A contextual diagram has been created to specify the functioning of the projected system and the most general description of processes. The first and second levels of the designed system are decomposed to divide it into separate components and specify the functions. A hierarchy of tasks is constructed to structure the process of implementing the projected system and determine the main functionality that should be created in the software solution. The system's leading tools for software implementation are identified: the programming language, auxiliary libraries, the development environment, and other software used during development. A comparative analysis was carried out, based on which the previously proposed software tools for implementing the development of the projected system were selected. The technical characteristics of the selected software tools and their features are described, and their application in specific cases is considered. The client-server architecture is preferred. The developed software is described in accordance with the standard. General information about the program and its purpose is provided. The logical structure, input and output data are described. A user manual has been created to help you operate the system and understand the features and flow of the designed software product. The features of the developed software solution are considered, and an example of application is given with an analysis of the results, which proves the legal capacity of the system. The costs of developing a design solution for an information system for searching for arbitrage situations in the cryptocurrency market are calculated, and its economic feasibility is proved.

6. References

[1] V. Lytvyn, V. Vysotska, V. Kuchkovskiy, I. Bobyk, O. Malanchuk, Y. Ryshkovets, I. Pelekh, O. Brodyak, V. Bobrivetc, V. Panasyuk, Development of the system to integrate and generate content considering the cryptocurrent needs of users, volume 1(2-97) of Eastern-European Journal of Enterprise Technologies, 2019, pp. 18-39. DOI: 10.15587/1729-4061.2019.154709

[2] J. Frankenfield. Cryptocurrency. Investopedia: Sharper insight, better investing. 2019. URL: https://www.investopedia.com/terms/c/cryptocurrency.asp.

[3] N. Reiff. Blockchain Explained. Investopedia: Sharper insight, better investing. 2020. URL: https://www.investopedia.com/terms/b/blockchain.asp.

[4] A. Berko, M. Bublyk, L. Chyrun, Y. Matseliukh, R. Levus, V. Panasyuk, O. Brodyak, L. Dzyubyk, O. Garbich-Moshora, Models and Methods for E-Commerce Systems Designing in the Global Economy Development Conditions Based on Mealy and Moore Machines, volume Vol-2870of CEUR Workshop Proceedings, 2021, pp. 1574-1593.

[5] A. Antonopoulos, Mastering Bitcoin: Unlocking Digital Cryptocurrencies. O'Reilly Media, 2014.

[6] A. M. Antonopoulos, G. Wood, Mastering Ethereum: Building Smart Contracts and DApps. O'Reilly Media, 2018.

[7] YU. V. Fedorova Kryptovalyuty ta yikh mistse u finansoviy systemi. URL: http://www.economyandsociety.in.ua/journal/15_ukr/116.pdf.

[8] Lee A. Smales, Bitcoin as a Safe Haven: Is It Even Worth Considering? URL: http://dx.doi.org/10.2139/ssrn.3204237.

[9] W. Suberg, Bitcoin Pizza Day 2018: Community Celebrates a Takeout Order Now Worth $82 mln. URL: https://cointelegraph.com/news/bitcoin-pizza-day-2018-community-celebrates-a-takeout-order-now-worth-82-mln.

[10] YE. O. Halushka, O. D. Pakon, Sutnistʹ kryptovalyut ta perspektyvy yikh rozvytku. URL: http://nbuv.gov.ua/UJRN/molv_2017_4_149.

[11] J. Edwards, Bitcoin's Price History. Investopedia: Sharper insight, better investing. 2020. URL: https://www.investopedia.com/articles/forex/121815/bitcoins-price-history.asp.

[12] Top 100 Cryptocurrencies by Market Capitalization. URL: https://coinmarketcap.com/.

[13] V. D. Bazylevych, et al., Sutʹ ta osoblyvosti provedennya arbitrazhu. Kyiv: Znannya, 2015.

[14] R. Veysveller Arbitrazh. Vozmozhnosti i tekhnika operatsiy na finansovykh i tovarnykh rynkakh. Moscwa: Tserikh-PE, 1995.

[15] S. Krückeberg, P. Scholz, Decentralized Efficiency? Arbitrage in Bitcoin Markets, in: Financial Analysts Journal, 2020. URL: https://doi.org/10.1080/0015198X.2020.1733902.

[16] Matt Levine, Crypto Bots Feast on Your Typos. Bloomberg. 2019. URL: https://www.bloomberg.com/opinion/articles/2019-04-16/crypto-bots-feast-on-your-typos.

[17] K. Egorova, Crypto Exchanges, Explained. Cointelegraph Bitcoin & Ethereum Blockchain News. 2018. URL: https://cointelegraph.com/explained/crypto-exchanges-explained.

[18] J. Chen. Currency Pair Definition. Investopedia: Sharper insight, better investing. 2019. URL: https://www.investopedia.com/terms/c/currencypair.asp#what-is-a-currency-pair.

[19] W. Kenton. Order Book. Investopedia: Sharper insight, better investing. 2020. URL: https://www.investopedia.com/terms/o/order-book.asp.

[20] C. Blenkinsop, Arbitrage Trading in Crypto, Explained. Cointelegraph Bitcoin & Ethereum Blockchain News. 2019. URL: https://cointelegraph.com/explained/arbitrage-trading-in-crypto-explained.

[21] Triangular arbitrage. URL: https://en.wikipedia.org/wiki/Triangular_arbitrage.

[22] T. G. Fischer, C. Krauss, A. Deinert, Statistical Arbitrage in Cryptocurrency Markets, volume 12(1) of Journal of Risk and Financial Management, MDPI AG, 2019. URL: https://www.mdpi.com/1911-8074/12/1/31.

[23] A Brief Look At Crypto Arbitrage Trading. Steemit. 2016. URL: https://steemit.com/cryptocurrency/@scrawl/a-brief-look-at-crypto-arbitrage-trading.

[24] Sangeet Moy Das, Algo Trading 101 for Dummies like Me. Towards Data Science. 2019. URL: https://towardsdatascience.com/algo-trading-101-for-dummies-like-me-b3938725d184.

[25] C. Pauna, Arbitrage Trading Systems for Cryptocurrencies. Design Principles and Server Architecture, volume 22(2) of Informatică economică. Inforec Association, 2018, pp. 35-42. URL: https://ideas.repec.org/a/aes/infoec/v22y2018i2p35-42.html.

[26] Arby About. 2020. URL: https://arby.trade/about-project.

[27] Blackbird. The world's leading software development platform GitHub. 2018. URL: https://github.com/butor/blackbird.

[28] About Gekko. Open source bitcoin trading bot platform. 2020. URL: https://gekko.wizb.it/docs/introduction/about_gekko.html.

[29] Zenbot. The world's leading software development platform GitHub. 2020. URL: https://github.com/DeviaVir/zenbot.

[30] Y. Matseliukh, V. Vysotska, M. Bublyk, T. Kopach, O. Korolenko, Network Modelling of Resource Consumption Intensities in Human Capital Management in Digital Business Enterprises by the Critical Path Method, volume Vol-2851 of CEUR Workshop Proceedings, 2021, pp. 366-380.

[31] V. Vysotska, A. Berko, M. Bublyk, L. Chyrun, A. Vysotsky, K. Doroshkevych, Methods and tools for web resources processing in e-commercial content systems, in: IEEE 15th International Scientific and Technical Conference on Computer Sciences and Information Technologies, CSIT 2020, pp. 114-118. DOI: 10.1109/CSIT49958.2020.9321950

[32] S. Peyrott, A Brief History of JavaScript. Auth0: Identity is Complex. Deal with it. 2017. URL: https://auth0.com/blog/a-brief-history-of-javascript/.

[33] J. Wolfe, A Brief History of Python. Medium – Get smarter about what matters to you. 2018. URL: https://medium.com/@johnwolfe820/a-brief-history-of-python-ca2fa1f2e99e.

[34] O. Romanyuk, NodeJS vs Python: How to Choose the Best Technology to Develop Your Web App's Back End. freeCodeCamp.org: Learn to code at home. 2020. URL: https://www.freecodecamp.org/news/nodejs-vs-python-choosing-the-best-technology-to-develop-back-end-of-your-web-app/.

[35] Express/Node introduction. MDN Web Docs. 2020. URL: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction.

[36] S. Robinson, The Node.js Request Module. Stack Abuse. 2019. URL: https://stackabuse.com/the-node-js-request-module/.

[37] J. Hall. Introduction to the fs Module in Node.js. DigitalOcean – The developer cloud. 2019. URL: https://www.digitalocean.com/community/tutorials/nodejs-intro-to-fs-module.

[38] Sejal Jaiswal, Overview of Atom IDE. DataCamp. 2018. URL: https://www.datacamp.com/community/tutorials/ide-atom.

[39] Ankit Muchhala, UI Testing with Postman. Medium – Get smarter about what matters to you. 2019. URL: https://medium.com/better-practices/ui-testing-with-postman-df713eb9788c.

[40] V. Vysotska, R. Hasko, V. Kuchkovskiy, Process analysis in electronic content commerce system, in: Proceedings of the International Conference on Computer Sciences and Information Technologies, CSIT, 2015, pp. 120-123. DOI: 10.1109/STC-CSIT.2015.7325447

[41] I. Moskovchenko, A. Kuznetsov, S. Kavun, B. Akhmetov, I. Bilozertsev, S. Smirnov, Heuristic methods for the design of cryptographic Boolean functions, volume 18(3) of International Journal of Computing, 2019, pp. 265-277.

[42] L. Chyrun, P. Kravets, O. Garasym, A. Gozhyj, I. Kalinina, Cryptographic information protection algorithm selection optimization for electronic governance IT project management by the analytic hierarchy process based on nonlinear conclusion criteria, volume 2565 of CEUR Workshop Proceedings, 2020, pp. 205-220.

[43] M. Bublyk, V. Vysotska, L. Chyrun, V. Panasyuk, O. Brodyak, Assessing Security Risks Method in E-Commerce System for IT Portfolio Management, volume Vol-2853 of CEUR Workshop Proceedings, 2021, pp. 462-479.

[44] V. Lytvyn, V. Vysotska, Designing architecture of electronic content commerce system. In: Computer Science and Information Technologies, in: Proceedings of the International Conference on Computer Sciences and Information Technologies, CSIT, 2015, pp. 115-119. DOI: 10.1109/STC-CSIT.2015.7325446

[45] V. Vysotska, I. Rishnyak, L. Chyrun, Analysis and evaluation of risks in electronic commerce, in: Proceedings of the 9th International Conference on CAD Systems in Microelectronics, 2007, pp. 332-333. DOI: 10.1109/CADSM.2007.4297570
[46] V. Vysotska, L. Chyrun, L. Chyrun, Information Technology of Processing Information Resources in Electronic Content Commerce Systems, in: Proceedings of the International Conference on Computer Sciences and Information Technologies, CSIT, 2016, pp. 212-222. DOI: 10.1109/STC-CSIT.2016.7589909
[47] V. Vysotska, L. Chyrun, Methods of information resources processing in electronic content commerce systems, in: Proceedings of 13th International Conference: The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM, 2015.
[48] V. Andrunyk, L. Chyrun, V. Vysotska, Electronic content commerce system development, in: Proceedings of 13th International Conference: The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM, 2015.
[49] K. Alieksieieva, A. Berko, V. Vysotska, Technology of commercial web-resource processing, in: Proceedings of 13th International Conference: The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM, 2015.
[50] A. Gozhyj, I. Kalinina, V. Vysotska, S. Sachenko, R. Kovalchuk, Qualitative and Quantitative Characteristics Analysis for Information Security Risk Assessment in E-Commerce Systems, volume Vol-2762 of CEUR Workshop Proceedings, 2020, pp. 177-190.
[51] V. Vysotska, A. Berko, M. Bublyk, L. Chyrun, A. Vysotsky, K. Doroshkevych, Methods and tools for web resources processing in e-commercial content systems, in: IEEE 15th International Scientific and Technical Conference on Computer Sciences and Information Technologies, CSIT, 2020, pp. 114-118. DOI: 10.1109/CSIT49958.2020.9321950
[52] L. Chyrun, The E-Commerce Systems Modelling Based on Petri Networks, volume Vol-2870 of CEUR Workshop Proceedings, 2021, pp. 1604-163.
[53] A. Berko, V. Andrunyk, L. Chyrun, M. Sorokovskyy, O. Oborska, O. Oryshchyn, M. Luchkevych, O. Brodovska, The Content Analysis Method for the Information Resources Formation in Electronic Content Commerce Systems, volume Vol-2870 of CEUR Workshop Proceedings, 2021, pp. 1632-1651.
[54] V. Kuchkovskiy, V. Andrunyk, M. Krylyshyn, L. Chyrun, A. Vysotskyi, S. Chyrun, N. Sokulska, I. Brodovska, Application of Online Marketing Methods and SEO Technologies for Web Resources Analysis within the Region, volume Vol-2870 of CEUR Workshop Proceedings, 2021, pp. 1652-1693.
[55] A.Demchuk, et al., Commercial content distribution system based on neural network and machine learning, volume Vol-2516 of CEUR Workshop Proceedings, 2019, pp. 40–57.