A Design of Cyber-physical Production System Prototype Based on an Ethereum Private Network

Maxim Ya. Afanasev, Anastasiya A. Krylova, Sergey A. Shorokhov, Yuri V. Fedosov, Anastasiia S. Sidorenko
School of Computer Technologies and Control
ITMO University
St. Petersburg, Russia
amax@niuitmo.ru, {ananasn94, stratumxspb}@gmail.com, {yf01, olesiia722}@yandex.ru

***Abstract*—The concept of cyber-physical production systems is highly discussed amongst researchers and industry experts, however, the implementation options for these systems rely mainly on obsolete technologies. Despite the fact that the blockchain is most often associated with cryptocurrency, it is fundamentally wrong to deny the universality of this technology and the prospects for its application in other industries. For example, in the insurance sector or in a number of identity verification services. This article discusses the deployment of the CPPS backbone network based on the Ethereum private blockchain system. The structure of the network is described as well as its interaction with the help of smart contracts, based on the consumption of cryptocurrency for various operations.**

I. Introduction

Blockchain is now one of the most actively discussed topics in various fields of human activity. Blockchain became widely known when the Bitcoin cryptocurrency first appeared in 2009. However, specialists started to reflect on the application of this technology not so long ago. Blockchain is a technology for storing and processing data in distributed computer networks and does not relate to any specific domain. All the blocks in blockchain can contain any data, which allows one to think about its use in production.

Development of the cyber-physical production systems (CPPS) and Industrial Internet of Things (IIoT) raises many problems related to the manipulation of data, such as storage, access, security, etc., that need to be solved. Moreover, there is currently a tendency to create distributed systems instead of centralized ones. One of the important properties of the Industrial Internet is the autonomy of its nodes and their ability to interact with each other. Such interaction is often based on the concept of service when certain nodes provide services to other nodes of the network. To ensure similar interaction, some implementations of a blockchain have a special mechanisms—smart contracts.

A smart contract is a self-executing scenario, which is stored in the blockchain as well as other data. Each smart contract has a certain algorithm implemented in a special programming language, which makes it possible to perform any actions automatically without involving third parties. Smart contracts monitor the fulfillment of certain conditions and makes decisions based on them in accordance with the specified algorithm. Since any network participant can sign the contract, interaction is applicable to smart things in the IoT. This approach can provide a trusted environment for interaction in the network and make the service providing mechanism transparent and unified. Furthermore, there is no need to create a separate register of services, since all contracts are already stored in the blockchain.

To date, there are many different implementations of smart contracts, but one of the first widespread implementations put into practice was the implementation of smart contracts in the peer to peer payment system, Ethereum in 2013. This implementation is currently the most advanced and has many features, like private blockchains, different protocols of consensus, etc. However, there have been other attempts such as Hyperledger [1].

These developments call for a more detailed study of smart contracts application in the Industrial Internet of Things. Additionally, it is necessary to consider not only theoretical features but also to carry out practical experiments to create a prototype.

The remainder of this paper is structured as follows. Section II gives the overview of studies dedicated to blockchain application in the context of IIoT and CPPS. Section III describes the possible role and place of blockchain in CPPS and IIoT. In Section IV, the architecture of the backbone network of the CPPS is proposed and, a created prototype of such a network is provided. The main limitations and drawbacks of the proposed approach are discussed in Section V. Finally, Section VI presents conclusions and suggests some future areas of research and development.

II. Related work

Despite the fact that the topic is relatively new, researchers and specialists have already published quite a large number of articles pertaining to the scope of the blockchain and production domain as well as adjacent production areas.

The discussion has mostly touched on the topic of the smart contracts. In the article [2], Nikolay Teslya considers the use of blockchain in IIoT. The article proposes the architecture of the Internet of Things, which is based on the Smart-M3 information sharing platform developed by them and the Hyperledger blockchain platform with smart contracts [3]. The authors note the drawback—restriction in the complex search

for information on the blockchain. As a solution, they created an additional information layer that performs search functions, data representation, additional checks, etc.

One of the neighboring areas to production domains are the supply chains. The place that blockchain can take in this sphere is discussed in the article [4]. The authors note that currently there is a lack of transparency in supply chains. Due to its transparency and immutability, as well as the mechanism of smart contracts, blockchain can help to automatically identify and record events in supply chains. A good example is in the article [5] where a cargo transition between points from the place of production to the desired destination is considered. Using radio tags and the smart contracts registration and tracking of the cargo are carried out automatically. Similar ideas are suggested by the authors of [6]. They propose the architecture of an information platform for the creation of the food supply chain traceability systems.

There are a huge amounts of blockchain implementations, which may be suitable or unsuitable for the IIoT. The author of [7] compared distributed ledgers by the most important characteristics for IIoT, such as the presence of smart contracts, the transaction time and the consensus protocol. In the comparison, the three most well-known distributed ledgers—IOTA, Ethereum, and Hyperledger—are involved. There are also publications where not only a qualitative but also a quantitative assessment of private blockchains is carried out [8], [9]. The authors of [9] created a special framework that allows you to analyze through special API private blockchains based on Ethereum, Parity and Hyperledger Fabric. The tool is open-sourced and can be really useful for blockchain analysis.

III. Place and role of blockchain in the structure of CPPS and IIoT

Blockchain technology is quite universal. To date, there are a significant number of implementations of blockchain used in various areas of human activity. In order to effectively use all the advantages of blockchain technology for building CPPS and IIoT, it is necessary to develop a network structure for the blockchain system that is optimal from the point of view of the solved tasks, and to choose the most suitable tools (software and hardware).

Many manufacturers of equipment, both in private conversations and publicly, have declared that they are engaged in the development of components and technologies for the industrial Internet of Things (and, consequently, for cyber-physical production systems) for many years. In their opinion, the only thing that has changed recently is the name of such systems. The essence has stayed unchanged since both the Industrial Internet of Things and cyber-physical production systems are nothing more than a set of industrial controllers, networked and communicating in one of the most common industrial protocols (Modbus, Profibus, CANbus, etc).

To confirm or disprove this statement, it is necessary to refer to the definitions of cyber-physical systems and the Internet of Things. Initially, the concept of “cyber-physical system” will be considered. In accordance with the standard NIST Special Publication 1500-201, “Cyber-physical systems are smart systems that include engineered interacting networks of physical and computational components.” Such a definition itself makes it quite clear that the industrial cyber-physical system differs significantly from the usual industrial network of controllers.

Let’s consider this issue in more detail. According to the definition, the main distinguishing feature of cyber-physical systems is the almost full “transparency” of the connections between the physical and logical components of the system. In the cyber-physical system, the distinction between real and virtual components is erased. In contrast, in traditional industrial networks the main focus is on physical devices—programmable logic controllers (PLCs). In accordance with this concept, only PLCs perform calculations in the network and are also responsible for the proper execution of the technological process and the information interaction of industrial equipment (both with each other and with the external environment). With that, the concepts of “information flow” and “material flow” are clearly separated.

It was the PLC that made it possible to create the first automated and automatic production lines, and it was the development and improvement of these basic nodes of industrial networks that focused the main efforts of manufacturers of such intelligent industrial equipment. Protocols and data transmitted over these protocols received much less attention as each manufacturer sought to create its own industrial protocol, which was incompatible with the protocols of other manufacturers.

In addition, we can add incomplete correspondence of the industrial protocols to the requirements for modern information and telecommunication technologies. For example, the widespread industrial protocol Modbus, despite its openness, uses approaches from the 1970s that manufacturers are trying to improve by using modern transport protocols, in particular TCP, which seems an extremely irrational approach. Also, it should be noted that the concept of “cyber security” in principle is not applicable to many of the currently used industry standards, again because of their strong lag behind modern information and telecommunications technologies.

For example, one implementation of the Modbus protocol—Modbus RTU—uses a physical connection based on the RS-485 standard, which is a slightly modernized UART. These protocols are among the first digital protocols for data transmission, and at the time of their widespread distribution there was no concept of cyber security. Implementing Modbus TCP simply allows you to put Modbus RTU messages in TCP/IP packets. It should be noted that Modbus is not only a protocol for receiving data from some passive devices (sensors, for example), but also a control protocol.

Consequently, this suggests that despite the external similarity, the industrial networks of the PLC (as well as the process control systems based on them) are not industrial cyber-physical systems. Let’s turn to the comparison of industrial networks with the Industrial Internet of Things. Lets start with the general definition of the Internet of Things. Unfortunately, we could not find any industrial or other standards, in the list of terms of which would be a present definition of the concept of “Internet of Things”. However, on the basis of the analyzed literature it is possible to give the following general formulation of the term: “The Internet of Things is a computer network consisting of physical devices equipped

with built-in electronics (and software), sensors, actuators and communication means that allow these devices to communicate with each other and to exchange data."

Such a definition seems to be similar to the definition of the concept of "cyber-physical system", but there is one significant difference between them. The Internet of Things is not designed to reach a certain common goal, that is, it is not a system in its classical meaning. According to the Oxford English Dictionary, a "System is a set of things working together as parts of a mechanism or an interconnecting network; a complex whole."

It should be noted that the cyber-physical system is holonic [10], that is, consisting of a set of "physical entities" and their "digital twins" connected together. "Digital twins" is a computational model of "physical essence", that is, it reproduces the behavior of a physical machine and gives an idea of how this machine reacts when various external influences occur. The connection can be ensured using sensors and actuators.

The holonic nature of cyber-physical systems implies the existence of both hierarchical and heterarchical links, which means that they can be combined into higher-order temporal entities (in turn, again performing one common function), that is, a "System-of-systems" [11]. At the same time, all communications are carried out via common and open to all participants network, which is the Internet of Things or the Industrial Internet of Things. Thus, the cyber-physical system forms the first level, and the Internet of Things is the second level of vertical digital integration.

It is clear that the standards of industrial networks based on PLCs, even on a set of properties, cannot be considered as either cyber-physical industrial systems or the Industrial Internet of Things. The last statement is due to the fact that such networks are not protocol-oriented or service-oriented solutions. They have a clear separation of material and information flows, do not have analogs of "digital twins" in their composition, and also have a rigid hierarchical control system with a single center, not assuming that there are heterarchical links and opportunities for flexible restructuring.

It should be noted that the problem of obsolescence of industrial protocols appeared a long time ago. In 1996, OPC Foundation developed the OPC (Open Platform Communications) family of protocols, which were intended to become the main technology, providing a single interface for managing automation objects and technological processes. The basic principle of this standard is "Open Communications on Open Protocols." However, for the most part, this protocol was based on Microsoft technologies and therefore was not widely used.

The logical evolution of the OPC protocol, called the OPC UA (OPC Unified Architecture) is more useful. This specification defines the transmission of data in industrial networks and the interaction of devices in them. The main advantages of this approach are openness, independence from any particular operating system or technology, scalability, and the implementation of its own system for ensuring cybersecurity. The unified architecture of OPC is a service-oriented architecture (SOA) and is based on various logical levels.

However, like any other specification, striving to become an industry standard, the OPC UA is not without certain drawbacks. First, the OPC UA is a fairly complex protocol that attempts to accommodate a wide range of areas of industrial automation. Secondly, OPC UA is also focused on industrial PLCs. In general, this is another protocol for data transfer between industrial devices (machine-to-machine communication protocol), only based on TCP/IP and XML technologies. Nevertheless, OPC UA is the most elaborate protocol today and it was decided to make it the basis of the developed CPPS architecture.

Therefore, having examined in detail the concepts of IIoT and CPPS, we now consider the blockchain technology position in their structure. The first thing that needs to be addressed is the fact that there are two main subclasses of blockchain: global and private. The first are the most developed and are mainly used for solving global problems such as organizing international peer-to-peer payment systems or crowdfunding.

Global peer-to-peer networks are highly stable because of the large number of participants but they are not suitable for creating industrial networks similar to those described earlier. At the same time, the main limitation is the rigid linking of all data exchange operations to the cryptocurrency used in a global cryptocurrency blockchain. In other words, with this technology, any interaction between CPPS nodes will have a certain cost, and it will be directly related to the price of the cryptocurrency used. It is almost impossible to predict the change in the exchange quotations on the market of cryptocurrencies, which makes the predictable cost of ownership of the projected CPPS difficult. Consequently, the basis for the projected architecture will be a *private blockchain*.

Next, it is necessary to determine the main functions of blockchain libraries in the structure of the CPPS, as well as determine the main tools with which these functions can be implemented. The main functions of blockchain in the CPPS should be:

1) Organization of a common information space for the machine-to-machine interaction within CPPS.
2) Ensuring CPPS cyber-security.
3) Ensuring an easy scaling and CPPS restructuration.
4) Ensuring of redundancy of equipment and communication channels.
5) Ensuring common data storage.
6) Implementation of "digital twins" technology through the use of smart contracts.
7) Ensuring implementation of common tasks for CPPS through the use of smart contracts.

The requirements quite severely limit the choice of tools since the implementation of key functions of CPPS implies the possibility of working with smart contracts. At the moment, smart contracts are implemented in several public blockchain distributed computing platform. However, only Ethereum has a well-designed and debugged virtual machine (Ethereum Virtual Machine, EVM) that implements its own Turing-complete programming language, as well as the ability to work with many modern high-level programming languages by compiling them into the EVM bytecode. Thus, the private Ethereum platform will be used as the main platform for building the

described CPPS, and the Solidity programming language as a tool for the implementation of smart contracts.

IV. Proposed architecture

A. Backbone network of the CPPS

As noted in the previous section, the CPPS backbone network will be built on the basis of a private blockchain of Ethereum. The following types of nodes should be included in the network (Fig. 1):

Full nodes—download and store all the network blocks in the internal memory, and are responsible for checking the validity of the blocks and their compliance with the rules of the Ethereum consensus protocol. They can act as miners of new blocks and it is assumed that general purpose PCs running Windows or Linux will be used as full nodes. For example, it can be client machines of CPPS, SCADA server, database server, etc.

The presence of full nodes is extremely important for the full functioning of the network since the stability and uniformity of generating blocks directly depends on the number of nodes. The functions associated with block validation and consensus-building do not require large expenditures of computing resources and RAM. The main requirement is that there is enough disk space to store the full chain of blocks.

Modern general-purpose PCs have a sufficient amount of internal memory, the amount of which is rarely less than 500 GB. At the same time, the volume of the entire Ethereum global network is about 385 GB, while 20–30 GB of free disk space is enough for maintaining a full node. Obviously, for a private network, such numbers are unattainable.

It is also possible to use the resources of general purpose PCs during their idle time for mining new blocks. In this case, the task responsible for mining can be started in the background mode. However, questions of the analysis of congestion and energy consumption of general purpose PCs operating in this mode require additional research. So, at this stage, special mining farms have been created as such farms can act the same general purpose PCs that are working constantly and perform only the mining task.

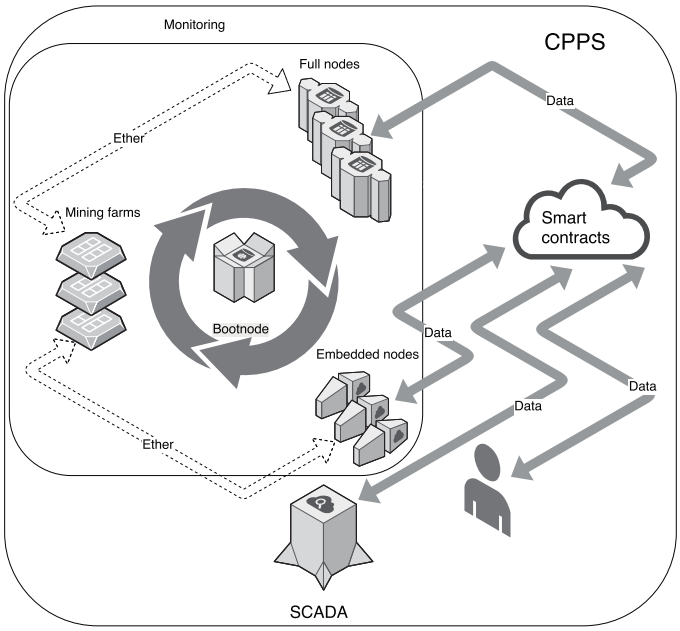

Fig. 1. Blockchain network architecture

It should be noted that in a private network there is the possibility of managing the complexity of mining so the mining farms need not be high-performance, have specialized GPUs or memory with increased bandwidth. Experiments have shown that the minimum number of mining farms is 3. The network of the blockchain system operates even if there is only one mining node, however, to ensure greater reliability and redundancy, the number of miners should be larger.

For the architecture under consideration, there is no need to try to optimize the process of searching for new blocks, for example, by running multithreaded mining, it is much more important to have a sufficient number of "slow" miners. This allows for having greater stability in the appearance of new blocks (generating blocks too fast leads to conflicts and branching), as well as the stability of the system due to redundancy. Also important is that, in the absence of miners in the network, the generation of new transactions will not stop but transactions will stop receiving confirmation, and therefore spreading across the network. In addition to all the foregoing, the mining farms will be charged with the distribution of the Ether between the nodes as we discuss in more detail below.

Bootnodes—the special nodes that perform the initial network boot. Each new node that connects to the network first accesses this node to get a network peers list. Without this node, it is necessary to implement the manual procedure of adding peers. It consumes very few resources, so it can be installed on a microcomputer, for example, a Raspberry Pi. However, for more stability and network redundancy, one needs to install several more bootnodes in parallel with other services.

Nodestat and monitoring node—a special node that monitors the network, allows for obtaining statistics: number of nodes, their load, the frequency of new blocks, the number of blocks, transactions, the performance of miners, the difficulty of mining new blocks, etc. Availability and working capacity of this node does not affect the network as a whole, so this node can be either dedicated or installed as a service to any suitable network node.

Embedded nodes—nodes, working on a light client protocol, specially designed for embedded smart property environments. In the considering architecture, they are the main type of nodes connected directly to equipment, controllers, sensors, etc. Although full security is only possible for a complete node, the light client protocol allows the light nodes to receive about 1 kB of data every 2 min from the network. These data allow it to perform a partial check on the state of the network in the part that interests them, as well as monitor compliance with the consensus.

B. Hardware and software

To implement the network of test blockchain units the following types of equipment were used:

General purpose PC—a low-performance computer used to solve a wide range of CPPS tasks. Specifications: Intel Core i5 CPU 2.4 GHz, 4 GB RAM, AR9285 Wireless Network Adapter 802.11 b/g/n, Yukon Optima 88E8059 Gigabit Ethernet. Operating system: Ubuntu 16.10, GNU/Linux 4.8.0-41 x86-64.

Laptop—a portable computer that can be used by the CPPS operator as a terminal, or for remote management. Specifications: MacBook Pro, Intel Core i5 CPU 2.6 GHz, 8 GB RAM, Gigabit Network. Operating system: OS X 10.10.5, Darwin 14.5.0.

Server—a high-performance computer used for the most resource-intensive software components of CPPS. Specifications: 2x Intel Xeon E5620 2.4 GHz, 32 GB RAM, Intel 82575EB Gigabit Network. Operating system: Ubuntu 16.04.2 LTS, GNU/Linux 4.4.0-64 x86-64.

Embedded system—system-on-chip, used to implement low-level algorithms in real time. Specifications: Amlogic S905 Quad Core Cortex-A53 1.5 GHz 64bit ARMv8 CPU with Mali-450 GPU, 2 GB, Realtek RTL8211F Gigabit Network. Operating system: Ubuntu 16.04 LTS.

Virtual private server (VPS)—a cloud-based virtual machine used to optimize the processing power of CPPS by transferring part of the software components to dedicated servers. Specifications: Intel Xeon CPU E5645 2.4 GHz, 512 MB RAM. Operating system: Ubuntu 16.04.2 LTS, GNU/Linux 2.6.32-042stab120.18 x86-64.

As the Ethereum client, the console client "geth" version 1.8.1 [12] was used. With its help, all complete nodes, mining farms, a monitoring node and bootnodes were implemented. The GUI of the monitoring system was implemented on the platform "eth-netstats" version 0.0.1 [13]. To test the network, as well as the deployment of smart contracts, the browser "Mist" (version 0.9.3) was used [14]. IDE "Remix" [15] was used for writing and debugging smart contracts for "Solidity" [16]. For the initial network configuration, the "puppet" manager, which is part of the "geth" distribution, was used.

For embedded systems on the ARM platform, the "EthEmbedded" client [17] was used-a Ubuntu OS special assembly for Raspberry Pi, Odroid microcomputers, etc. It should be noted that the implementation of the Ethereum client on the basis of PLC is possible. However, to date, the authors have not found a single mention of such projects.

C. Network deployment

The kernel of the private Ethereum network was deployed on the basis of the local network of the laboratory CPPS of Instrumentation Technologies Department. Also, several nodes located in the global network were connected to the network (several territorially separated cloud VPSs were used). At the first stage of the deployment of a private Ethereum network, bootstrap nodes were created. As the main bootnode, a Raspberry Pi microcomputer was used, which has a permanent connection to the local network and Internet access. To configure a bootnode, the key must be generated with the command:

```
bootnode -genkey nodekeyfile
```

After that, the bootnode can be started by:

```
bootnode -nodekey nodekeyfile
```

Then the bootnode address with a form like `enode://<node-name>@[::]:30301` will be displayed. In the future, this address will be used to configure other nodes, and the expression in square brackets is replaced by the IP address of the bootnode. To organize access from the external network, the address was additionally translated to the "white" address, as well as port forwarding 30301.

Next, the genesis block is set up the zero block of the blockchain. All nodes belonging to the same private network must have the same genesis. Genesis is a plain text file in JSON format. The file contents are as follows (Listing 1):

```
{
    "config": {
        "chainId": 73655,
        "homesteadBlock": 0,
        "eip155Block": 0,
        "eip158Block": 0
    },
    "alloc"      : {},
    "coinbase"   : "0x0...0",
    "difficulty" : "0x400",
    "extraData"  : "",
    "gasLimit"   : "0x2fefd8",
    "nonce"      : "0x0000000000000076",
    "mixhash"    : "0x0...0",
    "parentHash" : "0x0...0",
    "timestamp"  : "0x00"
}
```

Listing 1. Genesis file

The most important parameters in this file are:

`chainId` is a random number that is a unique identifier of the chain of blocks. It is necessary to prevent relay attacks when an attempt is made to record and then reproduce previously sent correct messages or their parts.

`nonce` is a "number that can only be used once", a one-time code selected in a random or pseudo-random manner, which is used for secure data transmission, and is also used to prevent relay attacks.

`difficulty` is a scalar value corresponding to the difficulty level of finding a new block. It determines the purpose of the mining, which can be calculated based on the complexity and time of generation of the previous block. The higher the complexity, the more calculations must be made by the miner to discover a new valid block. This value is used to control the block generation time in the blockchain, keeping the block generation frequency in a predetermined interval. For a private network, this value should be small in order to minimize the load on the mining farms.

`gasLimit` is a coefficient that is used to calculate the final cost of a transaction. At the initial stage, this value should be large enough so that smart contracts can be tested without hindrance. In the future, it can be changed.

After the genesis of the block is created, the first full node on the network is initialized with the following command:

```
geth
    --datadir .
    --bootnodes
        "enode://<name>@<ip>:30301, ..."
    --networkid 10777
init genesis.json
```

As one can see from the presented record, the start flags followed after the command, and then the client mode. If the mode is not specified, the client runs in background mode and waits for connections. The `--datadir` flag specifies the full path to the directory where the blockchain will be stored, the `--bootnodes` flag specifies the addresses of the bootstrap nodes, and the `--networkid` flag specifies the unique identifier of the private network, which must not coincide with the known global block IDs. The init mode starts the procedure for initializing the block library structure in the directory specified by the `--datadir` flag according to the genesis of the block specified in the `genesis.json` file, which should be located in the same directory.

To start the monitoring node, it is necessary to specify the `--rpc` flag, which allows access to the geth client API via the JSON-RPC protocol (port 8545). Geth uses console (accesses the node and the console at the same time) and attach (connects to the already-running node) modes to access the geth client console. At least one account must be created at each node through which it can receive funds from other sites and access smart contracts.

Mining is started by the command:

```
geth --mine --minerthreads=1
```

or from the console using the command:

```
miner.start(1)
```

As mentioned above, multithreaded mining is not applied to a private network, because it does not give any advantages, but only increases the load on the mining farms.

Initialization of embedded nodes does not differ from the initialization of the full nodes. The launch of embedded node occurs using the same command, only with the additional flag `--light`. The rest of the nodes are configured in the same way. It is important that, for the correct operation of the private network, all nodes must be synchronized in time.

The “Mist” browser is installed only on the nodes where it will be used. The setup of the statistics collection and monitoring node is well described in the documentation and is of no interest since it is reduced to a simple installation of the “Node.js” framework and deploying application.

D. Smart contracts

The first thing that is needed to pay attention to when considering smart contracts is the lack of the possibility of changing the smart contract after deployment. This is due to the special abilities of blockchain technology—it is only allowed to add data to the blockchain as removing something from the database, without violating the structure of the whole chain of blocks, is denied. Therefore, all contracts are deployed by the CPPS operators as necessary, and each contract has a self-destruct function, which can be started only from the address of the operator that carried out the deployment. Obviously, this creates a certain vulnerability, but it will not be possible to block a function that disrupts the operation of all CPPS without this function, and this is much more dangerous for its operability.

At the current stage of implementation of the CPPS under consideration, smart contracts are used to solve the following tasks:

- Implementing the “digital twins” of CPPS’ physical components
- Provide failsafe and redundancy of CPPS
- Providing a persistent store of process data
- Redistribution of funds between nodes.

The last task is the first thing that is needed to be discussed. Since the developed CPPS uses the Proof-of-Work consensus protocol, network nodes require a cryptocurrency (Ether), which they can convert into so-called “gas”, which is necessary for providing work in the network. In the Ethereum blockchain, “gas” is used to execute smart contracts, is consumed during the passage of transactions, and when data is stored in the blockchain.

As previously mentioned, mining on embedded nodes does not seem to be advisable because of their limited computing abilities. Because of this, there are mining farms in the network, which constantly generate new blocks, for which, according to the Proof-of-Work protocol, they get a reward—Ether. This reward is distributed amongst the CPPS nodes according to the following principle: all the nodes of the CPPS are ranked by their importance. The significance is determined by the influence of the node on the technological process realized by the CPPS.

For example, machine tools and industrial robots are considered “rich” customers and they are allocated the maximum amount of cryptocurrency, which gives them instant and unhindered access to all CPPS’ resources. Also the “middle class” is highlighted—CPPS’ components related to the planning and logistics of the production process such as automated ground vehicles. Units that are not directly related to the production process but are necessary for general observing and monitoring are considered “poor”, and they are given limited access to CPPS resources. For example, temperature and humidity sensors in the workshop are “poor”, because there is no need to receive data from them too often. Accordingly, a small amount of cryptocurrency allocated to such devices automatically will not allow them to use the resources of the network and distributed storage of CPPS too aggressively.

The Ether extracted by the mining farms is transferred to the address of the smart contract that carries out the redistribution procedure. The ranking algorithm determines the time intervals and automatically transfers funds from the mining farm wallet to the corresponding node.

The next task of smart contracts is the “digital twins” equipment implementation. At the current stage, it is possible to collect data from sensors used in CPPS. An example of such a contract with comments is presented in Listing 2.

```
pragma solidity ^0.4.18;

contract TempSensor {
    int256 temp = 0;
    address holder = msg.sender;
    event tempChanged(int256 temp);

    function
    setTemp(int256 newTemp)
    public {
        temp = newTemp;
        tempChanged(temp);
    }

    function
    getTemp()
    public constant returns (int256) {
        return temp;
    }

    function
    kill()
    public {
        if (msg.sender == holder) {
            selfdestruct(holder);
        }
    }
}
```

Listing 2. Smart contract for temperature sensor.

What is important is that this small and, at first sight, very simple smart contract also solves the task of creating a "digital twin". It is necessary to understand that the smart contract is a separate independent entity that is located in the blockchain. All data that the smart contract receives is automatically stored in the database of the blockchain, for example, technological process data, states, addresses of the senders of the transaction, transaction time, etc. That is, blockchain in general, and smart contracts in particular are excellent tools for collecting and analyzing statistics about the technological process, which is implemented in CPPS. At the same time, *these data cannot be changed or destroyed without completely removing all the nodes of the blockchain*. This behavior opens up new prospects for using blockchain technology such as to investigate emergencies, or the integration of CPPS on the basis of a blockchain system with a SCADA system for monitoring the technological process in real time. In addition, all the same contracts can provide both redundancy and fault tolerance of CPPS. Suppose that there are several identical temperature sensors in the workshop. Each of them is a smart thing, that is, it has built-in computing resources, a control program, and is connected to the blockchain network. The received data is sent to the blockchain unit via a function call of the smart contract, while the smart contract is a "digital-twin" of this array of sensors, thereby providing redundancy. The failure of one of the sensors will not disrupt the operation of the system. since the smart contract will continue to receive data.

The only problem that can arise when implementing such clusters of physical devices is their performance monitoring, because there may be a situation when all the sensors fail. But even this can be envisaged. It should be noted that the sensors are nodes of the block and receive a cryptocurrency from the mining farms. In accordance with the algorithm, they should spend it on access to the resources of the blockchain system, and if it turns out that one of the nodes ceases to create transactions and to work with smart contracts, it means, perhaps, this node has failed.

E. Test results

The network was tested for three weeks. The network had three mining farms and, in total, 161,000 blocks were found, with each miner receiving approximately 265,000 ETH at a gas price of 18 gwei. On average, a new block appeared on the network every 15.13 s. With a target value defined in the "geth" client code of 15 s. The time interval distribution corresponds to the classical Pareto distribution, which can be shown in Fig. 2. It should also be noted that in spite of the fact that the interval for the appearance of a new block in the blockchain is 15 seconds, the time of the block propagation through the network is 250 ms, that is, the process of distribution of new transactions has practically no effect on the performance of the network of blockchain. During the testing, more than 10,000 transactions were performed, related to the redistribution of the cryptocurrency and the implementation of smart contracts. The total database volume of the block database for the full node was 139 MB.

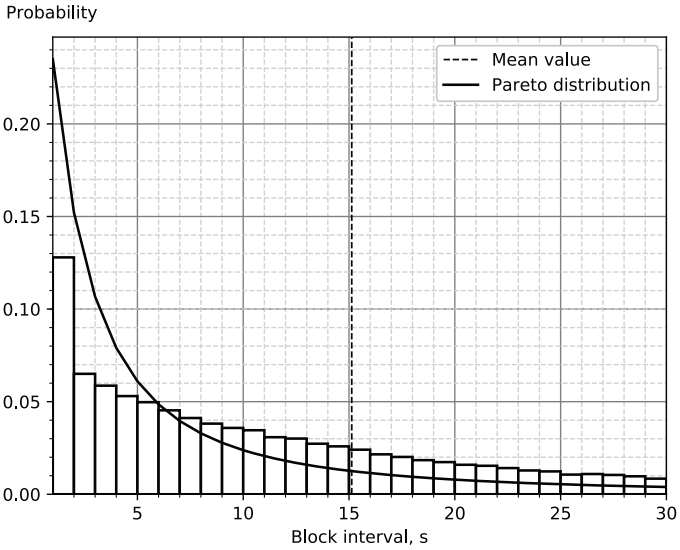

Fig. 2. The time interval distribution

V. Discussion

A. Performance issues

According to the authors, the main restriction of using blockchain to create CPPS for today is the low latency of the network, that is, the transaction does not pass until a new block appears. Since the generation of blocks depends on the complexity and complexity changes by the total network hashrate, the periodicity of the changes in each block, nevertheless, the average time for the appearance of a new block remains practically unchanged. This is especially noticeable if only one miner is working on the network.

Nevertheless, a cyber-physical production system based on a blockchain network can be referred to real-time systems, because it generally meets the requirements of such systems. Experiments have shown that it is able to react to events in an external environment, or to influence the environment within the required time constraints, and to process information for a certain finite period of time in order to maintain a constant and timely interaction with this environment.

Thus, such a system can at least fulfill the role of a real-time database, storing information about the technological process. At the same time from the point of view of data analysis, the frequency of collecting the readings from the sensors is not limited by the speed of appearance of the blocks in the network. In blockchain, the block is a storage unit, not a data transfer unit. Each block can have an unlimited number of transactions, each of which contains data from sensors arriving at the speed with which they are polled. Before the new block appears, all of this data is stored on the sending node, and in the block it is recorded and distributed over the network.

The uncontrolled frequency of the occurrence of blocks can make it difficult to transmit signals for logical control of equipment, that is, using smart contracts as "digital twins". However, according to the authors, such a control is still possible. It should be noted that the implementation of the algorithm for automatically modifying the complexity of block generation implies that the average frequency tends to some pre-programmed value. Testing the blockchain network showed that with a sufficient number of nodes, the average generation time of a new block stays unchanged within a few hundredths of a second, which is quite acceptable for many real-time production networks.

It should be noted that in any industrial networks there are delays in the transmission of control signals. These delays are caused by many factors, in particular obsolete physical layer protocols, which just are not able to provide an acceptable data transfer speed at large distances between network nodes. For example, the still used RS-485 protocol for a segment length of 1200 m provides a transfer rate of only 62.5 kbps. And this is only a physical level, delays can be on the channel, and even on the application layers.

Nevertheless, analyzing many years of experience in the use of industrial networks, it can be concluded that with proper networking, as well as the correct implementation of the management process, these delays do not affect the operation of industrial equipment. Also, it should not be forgotten that devices that generate emergency stop signals ("mushroom" head pushbuttons, automatic fuses, limit switches, etc.) are never connected to the industrial network by the safety rules, only directly to the equipment. Also, it must not be forgotten that the blockchain technology in general and the Ethereum network in particular are developing very rapidly. The latest versions of the "geth" client implement the consensus protocol Proof-of-Authority, which firstly allows one to almost completely abandon the mining, and secondly, it allows one to specify the exact interval for the appearance of a new block during the initial configuration of the block.

Also worth mentioning the Raiden technology [18] is a lightweight network within the global Ethereum network. It allows to execute transactions between predefined nodes without having to write to the main blockchain, while using all the advantages of the core network associated with security and anonymity. The main advantage of using a lightweight protocol is a reduction in commission and a shorter transaction time. Can be used for micropayments, in particular in applications of Internet of Things. As for mining, first, there is a clear tendency to abandon mining and the transition from the Proof-of-Work consensus model to Proof-of-Stake, and secondly, unused resources can be used for mining. In particular, the previously considered microcomputers such as Odroid or Raspberry Pi have built-in video processors that are not used to solve CPPS tasks, so they can be used for blocks mining. For example, GPU Raspberry Pi (Broadcom Videocore IV) supports OpenCL technology. The same applies to Mali video processors from T6xx version too (ODROID-XU4 microcomputer) and PowerVR SGX544 (microcomputer CubieBoard6). Support of this technology means that on these microcomputers it is possible to run the "ethminer" program—Ethereum miner with OpenCL, CUDA and stratum support, which allows to mine blocks with a hashrate comparable to the general purpose PCs, with only a much lower energy consumption.

B. Security issues

Undoubtedly, like any other open network, a blockchain may be subject to a hacker attack. The most dangerous type of attack for blockchains is a "51 % attack", in which the attacker can block all transactions on the network [19]. However, in order to carry out such an attack on private blockchain, the attacker will not only need a physical connection to the network, but also the availability of the genesis-block for registering the node, as well as the resource costs for mining. Of course, with low computational complexity, there is a potential threat of hacking the network by temporarily connecting a high-performance cluster or even an ordinary high-performance PC.

Thus an interesting situation arises. The fact is that in order to carry out any malicious activity in a private detachment, a hacker needs an ether. The only way to get it is mining, that is, the generation of new blocks. Thus, even at the moment of infiltration, an attacker is forced to perform work to ensure the operability of a private network. From this, we can also conclude that the denial-of-service attack is extremely difficult to implement, because any transaction on the network entails the need to pay a commission, so simply "flood" the network with transactions will not work.

The reason for using blockchain in corporate networks on the basis of which CPPS can be built is the need to ensure security and openness simultaneously. The classical approach used to ensure the security of geographically distributed networks is the organization of virtual private networks (VPNs). Geographically distributed VPN nodes are combined in some kind of local network, all traffic is encrypted and passes through a public network along the tunnel organized in it. This approach proved its reliability and effectiveness, but it has a number of significant drawbacks. First, truly reliable VPN solutions can be implemented only with the use of specialized software and hardware systems that have embedded encrypting tools. Deployment of such complexes is a non-trivial task requiring the participation of highly qualified

specialists, and the cost of such solutions is quite high. It is also quite a challenge to change any structure and topology of the network. Even a simple extension by adding a new node (for example, when organizing a new unit) requires the purchase and configuration of additional equipment. At the same time, the reliability of a VPN is determined by the reliability of its most vulnerable node, because hacking one segment of the network theoretically can compromise the entire network.

Blockchain allows to simplify work with global and territorially distributed networks, as well as the possibility of simple integration with cloud solutions, which can be very useful in the organization of CPPS. Blockchain allows you to work your sites in an unfriendly environment without the need to "fence off" from the global network with tunnels, firewalls, etc. A potential attacker can connect to the network while remaining unnoticed. However, this connection will give the attacker not so many opportunities to intercept data, as well as interference in the network. The structure of the blockchain system is such that it is only allowed to add data to the decentralized database, the attacker will never be able to delete or change the data already stored in the blockchain. At the same time, any actions of the attacker aimed at destabilizing the work of the blockchain network require an active connection, that is, they will also remain in the database. This will not allow you to "clean up tracks" to hide the fact of infiltration.

VI. Conclusion

To implement the CPPS core network, it is not enough to design a protocol for exchanging data between nodes and sensors; it is necessary to take into account network security, data access mechanisms, and performance of hardware components. Considering the fact that many implementations of CPPS on production lines rely on the technologies of the 70s and 80s, an important task is to find an alternative solution. The authors suggest that the actively developing technology of the blockchain network can serve as a platform for a distributed decentralized network, not least thanks to a special method of communication between the network nodes of the blockchain system—smart contracts.

Based on the tests conducted by the private Ethereum network as a CPPS backbone network, it can be concluded that this implementation is possible, but has a number of problems that need to be resolved. First, the time of block generation directly depends on the complexity of the network algorithms, and without the appearance of a new block, the transaction cannot be passed. Secondly, the susceptibility of the network of blockers to a number of hacker attacks, which can block the passage of transactions in the network. Finally, the frequency of the occurrence of blocks is uncontrollable, which can cause undesirable delays in the network, however, these delays are rather small in time and are not critical. Nevertheless, the Ethereum blockchain is a good alternative to the current CPPS technologie, and given that the blockchain technology is currently actively developing and being finalized, many of the shortcomings can be eliminated in future work.

VII. Acknowledgements

This work was carried out under project no. 617026 "Technologies of cyber-physical systems: management, computing, security."

References

[1] D. T. T. Anh, M. Zhang, B. C. Ooi, and G. Chen, "Untangling blockchain: A data processing view of blockchain systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. PP, no. 99, pp. 1–20, 2018.

[2] N. Teslya and I. Ryabchikov, "Blockchain-based platform architecture for industrial IoT," in *Proceedings of the 21st Conference of Open Innovations Association FRUCT*, ser. FRUCT'21. Helsinki, Finland, Finland: FRUCT Oy, 2017, pp. 42:321–42:329. [Online]. Available: http://dl.acm.org/citation.cfm?id=3176190.3176232

[3] J. Honkola, H. Laine, R. Brown, and O. Tyrkk, "Smart-M3 information sharing platform," in *The IEEE symposium on Computers and Communications*, June 2010, pp. 1041–1046.

[4] J. H. Lee and M. Pilkington, "How the blockchain revolution will reshape the consumer electronics industry [future directions]," *IEEE Consumer Electronics Magazine*, vol. 6, no. 3, pp. 19–23, July 2017.

[5] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the Internet of Things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[6] F. Tian, "A supply chain traceability system for food safety based on HACCP, blockchain and Internet of Things," in *2017 International Conference on Service Systems and Service Management*, June 2017, pp. 1–6.

[7] V. A. Red, "Practical comparison of distributed ledger technologies for IoT," pp. 10 206–10 206–6, 2017. [Online]. Available: https://doi.org/10.1117/12.2262793

[8] S. Pongnumkul, C. Siripanpornchana, and S. Thajchayapong, "Performance analysis of private blockchain platforms in varying workloads," in *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, July 2017, pp. 1–6.

[9] T. T. A. Dinh, J. Wang, G. Chen, R. Liu, B. C. Ooi, and K.-L. Tan, "BLOCKBENCH: A framework for analyzing private blockchains," in *Proceedings of the 2017 ACM International Conference on Management of Data*, ser. SIGMOD '17. New York, NY, USA: ACM, 2017, pp. 1085–1100. [Online]. Available: http://doi.acm.org/10.1145/3035918.3064033

[10] M. Y. Afanasev, Y. V. Fedosov, A. A. Krylova, and S. A. Shorokhov, "Modular industrial equipment in cyber-physical production system: Architecture and integration," in *2017 21st Conference of Open Innovations Association (FRUCT)*, Nov 2017, pp. 1–9.

[11] A. W. Colombo, T. Bangemann, and S. Karnouskos, "A system of systems view on collaborative industrial automation," in *2013 IEEE International Conference on Industrial Technology (ICIT)*, Feb 2013, pp. 1968–1975.

[12] A. Beregszaszi, F. Vogelsteller, A. Maiboroda, I. Matias, J. Pitts, and J. Steiner. Go ethereum official repository. [Online]. Available: https://github.com/ethereum/go-ethereum/releases

[13] M. OANCΞA. Ethereum network stats official repository. [Online]. Available: https://github.com/cubedro/eth-netstats/release

[14] A. Beregszaszi, F. Vogelsteller, A. Maiboroda, I. Matias, J. Pitts, and J. Steiner. Mist browser official repository. [Online]. Available: https://github.com/ethereum/mist/releases

[15] ——. Remix IDE official repository. [Online]. Available: https://github.com/ethereum/remix

[16] ——. The Solidity contract-oriented programming language official repository. [Online]. Available: https://github.com/ethereum/solidity

[17] J. Gerryts. The EthEmbedded official repository. [Online]. Available: https://github.com/EthEmbedded

[18] Raiden network–fast, cheap, scalable token transfers for ethereum. [Online]. Available: https://github.com/EthEmbedded

[19] C. Natoli and V. Gramoli, "The blockchain anomaly," in *2016 IEEE 15th International Symposium on Network Computing and Applications (NCA)*, Oct 2016, pp. 310–317.