UDC 004.35:004.41:004.42 **doi: 10.32620/reks.2023.4.14**

Artem PEREPELITSYN[1], Oleksandr VDOVICHENKO[1], Vitalii MIKHALEVSKYI[2]

[1]*National Aerospace University "Kharkiv Aviation Institute", Kharkiv, Ukraine*
[2]*Khmelnytskyi National University, Khmelnytskyi, Ukraine*

SERVICE FOR COMMUNICATION OF DEVICES WITH INTERNET ACCESS: ANALYSIS OF TECHNOLOGIES AND METHOD OF CREATION

The subject *of study in this article is technologies and services of communication of elements of smart home, as well as the models, methods, and tools for prototyping of interaction of devices for Internet of Things (IoT). The* ***goal*** *is to simplify the process of creation a service for the programmable by an end user device with the ability to communicate over Internet and to simplify the process of choosing the components for such devices.* ***Task****: to perform the analysis of IoT systems requirements and possible issues during the process of creation; to perform the analysis of requirements of safety and cybersecurity for IoT systems; to perform the research and classification of components of IoT systems; to perform the research of communication process between elements of IoT system; to analyze IoT communication protocols; to perform the analysis of services for IoT communication; propose the models of communication of IoT nodes; propose technique for creation of service; and provide practical example of implementation of the research results. According to the tasks, the following* ***results*** *were obtained. The analysis of possible problems and requirements for prototyping elements of smart home systems with Internet access is performed. The analysis of the existing architectures layers of IoT systems is performed. Models of Edge and Fog for IoT system are considered. Analysis of cybersecurity considerations for the devices communication is performed. Five types of components for smart home systems were classified. The process of communication of smart home elements with service at the model level is described. Existing communication protocols for the communication of devices over Internet are analyzed. Commercial and open source communication services for different types of devices are analyzed. Models of IoT node interaction are proposed. Technique of creation of a service for communication of devices and example of use are proposed.* ***Conclusions.*** *The main contribution of this research is the proposed method of creation of own service for communication over Internet for devices implemented on the basis of widely used microcontrollers. Based on the proposed classification of IoT hardware components, it is possible to simplify the decision-making process of selection based on the parameters of price, required efforts from the end customer for installation, and expandability in terms of the possibility of integration of the solution with wider systems. The performed analysis allows to conclude that from existing services ThingSpeak is interesting for use with the simplest chips, and Home Assistant is the preferable solution for home automation.*

Keywords: *communication service; communication protocol; Internet of Things; IoT; smart home components; model of IoT node; ThingSpeak; Home Assistant.*

Introduction

The development of communication capabilities and the increase in the number of ready-made components for the construction of home systems, as well as individual modules for the prototyping of such solutions, actualize the possibility of organizing the interaction of individual elements of such home systems in a convenient way for their use [1].

The possibility of using a smartphone to control modules in a smart home significantly simplifies the communication process. The possibility of using reproducible solutions to organize interactions, both on the user and device sides, becomes relevant [2].

Many individual modules for the creation of such systems are quite popular, well described, and not powerful enough to support complex communication protocols. There are simple methods of communication that do not require implementation of complex algorithms and allow the use of devices based on popular 8-bit chips [3, 4].

Ensuring the required level of cybersecurity is one of the most important requirements for the creation of such systems with Internet access [5].

Popular sets of services for interfacing hardware solutions within smart home elements, across the full range of IoT solutions, in interaction with each other or with a central monitoring device, use the set of protocols of communication that simplify control [6].

There are a large number of IoT solutions and information about them, including their construction. Considering them in a generalized form allows us to find patterns of creation and deployment of such solutions at the higher level of abstraction using

ISSN 18-5 (print)
ISSN 26-22 (online)

model-based approaches to generalize these solutions and simplify the process of interacting with them [7].

These findings are required during the selection of services to create a selection strategy in terms of the applicability of the service for each individual case. Therefore, it is necessary to consider existing solutions, focusing on the ways in which the devices in the solution interact with each other.

The purpose of this work is to simplify the process of creation a service for the programmable by an end user device with the ability to communicate over Internet and to simplify the process of choosing the components for such devices. To achieve this goal, it is necessary to perform the following tasks:

1) to perform the analysis of IoT systems requirements and possible issues during the process of creation;

2) to analyze the requirements of safety and cybersecurity for IoT systems;

3) to perform the research and classification of components of IoT systems;

4) to perform the research of communication process between elements of IoT system;

5) to analyze IoT communication protocols;

6) to perform the analysis of services for IoT communication;

7) propose the models of communication of IoT nodes;

8) propose the technique for creation of service;

9) provide practical example of the implementation of the research results.

The methodology of investigation is based on the following steps: analysis of the theoretical and practical problems of prototyping, reconfiguration, and deployment of smart home systems and devices with Internet access; analysis of practical demands as issues of secure communication of boards based on microcontrollers with limited resources that actualize the investigation of the possibility to find common solution; analysis of components of typical IoT systems to find the criteria for abstract description; analysis of protocols and services to use the well working solutions as the parts during composing the system based on requirements; formulation of the models of IoT of node based on direction of data transferring; proposing a method of service creation and example.

Then, the operational basis can be created. Using this operational basis, it is possible to represent a certain structure for its description. For this operational basis, it is necessary to investigate the completeness and consistency of this operational basis.

Adding operations on the elements of this basis allows to obtain metaalgebra. The use of this obtained metaalgebra makes it possible to solve tasks of higher level of formalization, including prototyping and reconfiguration of such systems.

1. Analysis of requirements and issues of creation of IoT systems

The development of embedded solutions involves the consideration of a set of requirements for a particular device. Such requirements include specifics regarding the possible set of hardware components for system realization, communication protocols, dimensions, and other requirements.

In addition, one of the most important requirements is the cost and time of development, together with the technical complexities of construction.

The cost includes the costs of research, design and development, which in turn determines the order and complexity of the solution implementation within the framework of the task to be performed. The cost includes the cost of building the hardware and software components used in the development of software and hardware complexes.

Software components include all libraries, frameworks, and individual elements of existing and available solutions applicable to the project. Hardware components or blocks are commercially available elements, boards, and microcircuits designed to be combined and used in various devices.

The challenge of interface harmonization should also be considered an important issue. In other words, consider the problem of compatibility with popular solutions for managing and deploying IoT solutions.

Interfaces must be compatible within the application, between devices and the user interaction. The existence of common standards allows combining, adding, modifying, and interoperating the components and software bases of different solutions within a single solution. This approach is economically feasible and is considered a good design practice.

The selection of hardware and software components with a long period of development and support from the manufacturer is essential for the design process. This is especially true for open source components. This is important to consider avoiding problems during the deployment phase. For example, the experience of a particular set of devices, their environment, or communication channel may not be considered. This can lead to lack of compatibility.

If the solution has different communication methods, such as SIM cards, GSM modules, or Wi-Fi modules, there is a high probability of incompatibility at the level of frequencies, operators, or authorization data when attempting to connect.

To describe such systems and communication processes at a higher level of abstraction, it is necessary to perform an analysis of architectures of IoT systems, components, and processes of interaction of nodes.

1.1. Analysis of architectures for IoT

To correctly classify the systems under consideration, it is necessary to analyze the existing solutions. One of them is an approach to differentiate system types by architecture. Architectures such as three-tier, service-oriented, and middleware architectures are highlighted as relevant architectures for modern IoT systems.

The three-tier architecture consists of abstract parts called layers. The first application layer provides services that can be requested by clients. The layer provides intelligent services for the client's needs, such as temperature and humidity measurements.

It is used in numerous possible applications such as smart homes, buildings, and healthcare. It can also be used for the implementation of solutions for industrial automation and transportation [8].

The second layer is called the network layer and represents the communications that are responsible for transferring data for the application layer using various protocols and technologies. In addition, this layer provides other functions such as data management and cloud computing processes.

The third layer is called perception and can be used to represent the physical sensors of an IoT system that collect and process information. The layer includes various sensors and actuators that collect, digitize, and transmit data to the network layer through secure channels [9].

A server-oriented architecture is a component-based model that connects different applications and blocks using hardware interfaces and network protocols.

It can be described as being similar to a three-layer architecture constrained by a four-layer model. In addition to the previously described perception, network, and application layers, there is a fourth layer called service [8].

It comprises service discovery, composition, management, and interfaces. Discovery is used to discover service requests. Composition is used to interact with connected objects and integrate them with services to receive requests. Governance provides a trust mechanism for service requests. Service interfaces support the interaction between all the provided services [9].

The middleware architecture of software is a five-layer variation of the three-layer architecture that provides new features such as scalability, interoperability, reliability, and quality of service (QoS) [8].

The middleware layer includes critical functionality such as aggregation and filtering of data received from hardware devices. It also performs information discovery and access control on devices and applications. The business layer acts as a link between applications, data, and users.

It fulfills the need that depends on the technological advancement and design of various new applications and business models [9].

1.2. Fog and Edge architecture models

From the changing perspective of the hardware or software view of IoT systems, data collection, redirection, and processing functions are increasingly common. In this vein, IoT closely borders cloud, Fog and Edge computing through device-to-device or machine-to-machine communication [2].

In this configuration, the Fog and Edge architecture for IoT networks can use a variation of the layered model. In this representation, the network is divided into four layers with vertical redundancy of enabled nodes [10].

The first level described is the hardware or peripherals. It represents the lowest part of the system and includes physical devices such as low-power sensors, systems-on-chip (SoC), and power nodes.

The next layer can be described as middleware but for Fog and Edge deployment. It consists of two sub-layers known as IoT Edge layer gateway and Fog layer nodes. Gateway includes systems such as Edge gateways, security, aggregators, and hub computers. Fog nodes can be represented by actuators, storage devices, computers, and their networks. Fog and Edge middleware also have horizontal connectivity.

The use of low-powered long-range modules with support of 5G infrastructure integration on the gateway sub-level, and Network Functions Virtualization (NFV) on the Fog nodes sub-level, a lot of multi-versed Fog and Edge middleware duplicates can communicate with each other with the data and computing capacities exchanging.

The cloud layer is on top of the multi-tiered Fog and Edge levels architecture. This layer is organized as a global network of services. It provides a wide network of remote servers as the entire ecosystem. In this case, the cloud can be used as a universal tool for providing Edge computing for customers as a service.

2. Analysis of requirements of cybersecurity and safety of IoT systems

According to the above description of widely used IoT system architectures, it will be useful to consider their existing requirements from a cybersecurity perspective.

Information security and cybersecurity are important tasks for any system related to the production, storage, and transmission of information. Both subtasks of securing communication channels, accounts, and differentiating access rights to various elements of IoT infrastructure are considered. The importance of this component of the topic is higher as the project approaches the task of ensuring deployment for mass use. By analyzing the mechanisms of interaction between architecture layers, it is possible to identify the main vulnerabilities caused by their structure and develop requirements for their protection.

ISSN 18-5 (print)
ISSN 26-22 (online)

The perception layer can identify vulnerabilities related to the low processing power of individual nodes in the system. In this case, an individual node can be bypassed or disabled, thereby allowing unauthorized access to part of the system on its behalf [11].

To avoid this, authentication, data confidentiality, lightweight encryption, and key negotiation mechanisms are implemented in the system deployment phase.

In the application layer, the main source of difficulties is software unification. This makes the layer susceptible to the same types of attacks as standard public software. To decrease the effects of such attacks, information security management and privacy protection mechanisms, in addition to the familiar authentication mechanisms, are implemented in the system.

It should be noted that the problems for the network layer correlate with problems that occurred in both the perception layer and the application layer. Problems such as the vulnerability of individual nodes and congestion due to a large number of requests are complemented by problems of transportation data leakage and traffic disruption.

In addition to authentication, this layer uses security mechanisms such as communication and routing security, intrusion detection, and key management [12].

Safety of IoT systems is also a key feature of such systems. During the creation of devices that are responsible for critical system functions, the task of ensuring functional safety is a constant requirement. An example is the process of preventing emergencies during the operation of a smart home system. These include fire, flooding, and other undesirable accidents. In business-critical decisions, it is important to avoid financial losses. For example, a disruption in the company's logistics due to forklift truck malfunctions.

3. Research and classification of components of IoT systems

An important stage in the construction of IoT systems is the choice of the component base. Depending on the tasks of the system being developed, the set of components may vary [13]. To select the optimal set, it is advisable to study the entire range of ready-made solutions for building hardware systems.

The analysis shows the existence of a wide range of solutions, divided by cost, manufacturing method, and purpose.

Fully exclusive solutions are often used in the stage of creating the first prototype of a device. These include the author's single copies of components and boards that realize a narrow range of functions sufficient for household tasks. They are not mass produced and are often easily reproducible because of the use of serial components, boards, and controllers. Such solutions can test the author's methods and concepts, as well as partial or temporary replacement of factory solutions.

Standardized module sets for rapid prototyping are quite popular and recognizable. They include sets of frequently used boards and devices made in the form of modules. Due to mass production, systems built on their basis significantly win in terms of cost while maintaining the quality of performance. Often, their quality is close to that of factory solutions.

In addition, the modular design and standardization of the enclosures allows for easy and predictable reproducibility of the systems based on them. This, in turn, makes it possible to remotely reproduce and test system instances during the development phase. Due to the prevalence, the development time is also reduced, which, along with the possibility of prebuilding and built-in protection mechanisms of modules, avoids many errors that can cause failures in the final system.

In addition, similar kits have a solid distributed information base on the Internet, capturing user experience and providing recommendations for both users and manufacturers.

Mass-produced solutions represent the bulk of the off-the-shelf IoT solutions. This type of device is the basis for existing IoT solutions. They are optimized and often made in a single housing complex of high-precision subsystems for use in industrial and commercial systems. They are not always suitable for trial builds and test mockups because they are delivered as solutions with pre-installed software and settings, thus minimizing the time required to build and deploy systems.

Mounting solutions are also part of the off-the-shelf IoT market. This type of device is used at the design stage and at the beginning of building construction. These devices are installed in the carcass, facade, and interior and exterior elements.

Mostly, these solutions are used in the tasks of monitoring the state of the premises for fires, unauthorized intrusion, and the monitoring of power supply systems, communication lines, water supply, and drainage. It may be a unit with an autonomous power supply that is in sleep mode until the moment the monitored event occurs.

Exclusive professional devices are the most expensive of all off-the-shelf IoT solutions. This class of systems refers to devices of narrow circulation, designed to order with the help of production facilities. They are characterized by high quality workmanship, materials, and high production costs. For such systems, the manufacturer and supplier provide a support and maintenance program that minimizes risks related to functional and information security.

Component matching using a set of attributes shows the possibility of combining groups of devices depending on the matching category. Some have similar properties in a given category.

The results of the component analysis are presented in Table 1.

Table 1

Comparison of IoT device component categories in terms of cost, customization and modifiability

Category	Price	Required efforts	Support of modification
Single prototypes	Low	Maximal	Any modifications
DIY modules	Minimal	Depends on task	Any modifications
Off-the-shelf devices	Medium	Low	Depends on the device
Integrated devices	High	Minimal	By manufacturer, service center
High end professional devices	Maximal	Minimal	Only by manufacturer

For the selected categories within the classification, the components were compared in terms of price, labor costs required by the user to deploy and configure an instance of such a system, and the possibility of making changes to the software or hardware component of the device.

The results of the analysis allow the selection of components for a particular IoT system based on the specified requirements. It can be concluded that the most rational solutions at the system development stage are ready-made modules. They allow modification and at the same time have the lowest cost and availability in ready form in many regions for parallel development of the solution by a group from different countries.

4. Research of communication processes between elements of IoT system

In the case of the interaction of hardware components, when considering the interaction of elements within the operating systems of a smart home, it is important to define the roles of individual hardware components. In a generalized interaction model, the computing and control solution can act as a coordinator with Internet access.

Regarding the internal structure of the system, the coordinator may be designed as a node or group of nodes within the smart home system, with partial or full access to the periphery.

Peripherals refer to actuators and sensors that communicate with the coordinator via other interfaces. These can be local wired interfaces, infrared channels, optical in other bands, audio, including ultrasound, and radio channels.

The exception is the Internet, because there is a need to differentiate the channels of information exchange with the coordinator through the use of different interfaces. In this case, the task of communication on the Internet is performed entirely by the home system coordinator.

In addition to the coordinator, the IoT solution interaction service is also present in the component interaction scheme with the Internet. It provides and maintains the communication of the entire system. The consideration of these services is of great interest.

In the case of interaction between the IoT system and service elements, to control and display the system controls, an appropriate element within the system model is required. In practice, such an element can be a computer or telephone used for monitoring and controlling the system. This element is connected via the Internet through program interfaces with both the interaction services and the coordinator for data exchange and management.

Interaction within the system is understood as vertical interfacing with the coordinator through the IoT interaction service. In this case, when serving multiple systems, the service acts as a centralized monitor, while the monitoring device takes a secondary place.

In such a case, the monitoring device is designed to collect statistics or secondary information about the system operation.

In the classical case, the service acts as an intermediary between the control and monitoring device and the coordinator.

In both cases, the service, in addition to computational offloading, allows more flexible management of connections within the system by offloading the coordinator device. This significantly reduces the power consumption of the nodes.

In addition, in contrast to the non-service option, there is no need for a dedicated address for each coordinating node to provide connectivity. Instead, it is sufficient to provide support for a set of protocols for interacting with the service. The use of protocols greatly simplifies the structure of the device and the procedure of interaction with it.

When the device interacts with the service through the protocol and complies with the requirements of the complexity of the implementation of communication with the user, which is performed programmatically, a competent communication pro-

ISSN 18-5 (print)
ISSN 26-22 (online)

cess is performed, bypassing most of the problems associated with hardware implementation and power consumption issues.

The service interacts with a large number of instances of such solutions. This reduces the power consumption per unit of such instance compared with the solutions that realize the same functions without using the service. A significant portion of services are already designed for interaction with smart home elements and IoT solutions and provide direct support for such solutions.

The general structure of the interaction of the elements of the described IoT system is presented in Figure 1.

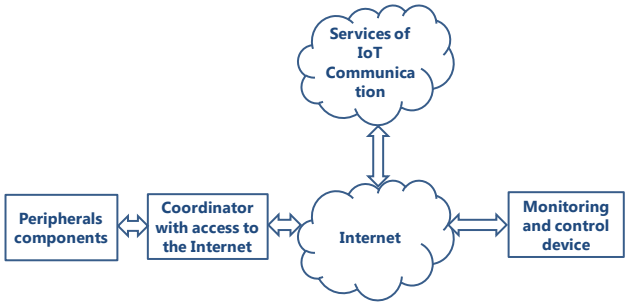

Fig. 1. Communication of elements of IoT system over Internet

5. Analysis of IoT communication protocols

Interaction protocols allow the realization of the process of communication with the considered type of devices. Some of them are designed for interaction with smart home elements and IoT solutions that provide direct access. A common example is services, which provide the ability to communicate using a stateful protocol.

REST (Representational State Transfer) is one of the possible protocols. Often, there are services that interact via REST. Transfers the representative status or state of an information resource. The same name can also refer to the architectural style of the interaction of distributed components in a network. Among such REST protocols, XMPP is a protocol for the fast exchange of messages and information that the device is present. Here, presence information is used at the protocol level to terminate exchanges in the absence of final information that the exchange is complete.

This saves system resources because of the ability to pause communication for long periods of time and still retain the presence information of the device. It stays in communication until an emergency channel breakdown occurs, which eliminates the possibility of communication.

XMPP (Extensible Messaging and Presence Protocol) is also commonly used as a communication protocol. In specialized systems, it is used to quickly exchange messages and, importantly, presence information.

This greatly saves resources when communicating with devices and makes it easier to verify their presence and activity on the network with the use of small boards and implementation in limited resources.

The leader among specialized interfaces after HTTP is MQTT. This protocol allows interaction and avoids problems related to data exchange. During the exchange process, a queue of messages is built both inside the device and on the service side. This protocol is supported by most of the services discussed below.

MQTT (Message Queue Telemetry Transport) is the leader in terms of frequency of use among specialized systems, after HTTP (Hypertext Transfer Protocol).

This protocol is a simplified network protocol based on TCP/IP request queues in a publisher-subscriber format. This communication protocol is supported by most IoT device communication services [14].

AMQP (Advanced Message Queuing Protocol) is the next protocol to consider. In its current form, it is an open standard focused on message processing.

It is found in only one-third of the total number of solutions in practice because of the low prevalence of support for it.

It is used to build a solution and select a possible service if the target hardware contains a library supporting this protocol.

CoAP (Constrained Application Protocol) is an important and frequently used protocol. It is a simplified HTTP for the Internet of Things. It is found in half of the solutions and services used. It establishes an exchange channel between the user and the end device.

This is a simplified HTTP protocol for this type of solution. It is quite popular and is found in at least half of the services that allow exchange between the end device and its user.

Different protocols of interaction, including HTTP, HTTPS, FTP, and Web Sockets, are also widely used in the construction of services for the interaction of elements of distributed hardware systems.

Other communication methods such as TCP, HTTP, WebSockets, direct socket connection, and other protocols should not be forgotten.

6. Analysis of services for communication

The power consumption of this part of the task is rather low. Within the framework of the task, the service meets the requirements of complexity of realizing interaction with the user due to its software implementation. With regard to the hardware part and energy consumption issues, the service combines a huge number of instances of low-energy-consumption solutions. Consequently, the power consumption per unit of such an instance is practically negligible compared to the comparable power consumption of an end device that can realize the same functions.

There is a rather powerful, wide range of services and manufacturing companies that provide Cloud resources that interact with IoT, including as a service. It is necessary to consider a group of well-known companies that allow building services for processing and interaction with IoT solutions, including Microsoft, Amazon, Google, IBM, and Cisco.

Azure IoT is one of the most secure services in terms of channel organization for working with IoT devices. It has one of the highest degrees of protection because of the implementation of high-level security mechanisms. It is also scalable and easy to deploy. It provides full-value support and comprehensive documentation.

AWS IoT is a service that provides cloud-based Amazon solutions and cloud resources, including FPGA as a Service solutions. This is a separate area of hardware implementation and access to hardware solutions at the cloud level. Within the framework of interaction with IoT solution it should be noted that AWS IoT supports an unlimited number of devices. Simultaneously, it has multi-level protection of the data itself and the ability to manage specific devices and their interaction. It also provides elements of artificial intelligence that allow to work with devices and perform the analysis of data from such devices.

Google Cloud Platform is the solution that works on security issues, and this is ensured at the level of the interaction itself and the platform as a whole. Google also stores data securely, has a mechanism related to the storage of data that comes from the device, and is protected at the level of this company. One of the important features of such a solution is the enormous amount of usable information.

IBM Watson IoT, unlike solutions from Google, is a fully paid solution and, in fact, that is why it is supported. Support ensures continuity of service and a high level of security. An additional feature of this solution is a set of tools for detailed analysis of the data that comes from the target device as part of the interaction.

Cisco IoT is a solution whose customers are communications companies, including cellular networks. Providers use separate solutions for remote access and interaction. In this case, a secure organization of interactions for target solutions is provided. A sufficiently resilient infrastructure provided in real time is used. Also to consider, smaller vendor companies such as Oracle and Eclipse IoT and products such as Saleforce IoT solutions, IRI solutions, Particle, and ThingWorx, supported by the community and developing board-based solutions mass-produced in the lower price segment.

ThingSpeak is definitely worth highlighting because it is one of the most common and well-known services because of its simple way of working and interaction. It provides MathLab tools for data analysis and visualization, namely, building graphs and various charts based on data from IoT solutions, sensors, and devices connected to this service.

This service also provides the possibility of registering individual nodes for the user and the possibility of tracking the location of some events and informing about these events. Data analysis and the ability to work with various solutions, including solutions with 8-bit microcontrollers, are the distinctive features of this service, and as a result, the service has become popular among the current range of services used for Arduino solutions. The resources of such chips are limited for some protocols [4].

CyberVision offers **KAA** service and the global infrastructure with global solutions for building completely different services. It provides an opportunity to implement protocols such as REST and provides a set of languages Java, C and C++ for the realization of client-server connections in the form of ToolKIT.

Zetta is a server-oriented service on NodeJS. It is implemented using REST and WebSockets protocols. It considers device as a REST interface and allows interaction with it. It supports all Raspberry PI series devices and their modifications. ARM solutions compatible with Arduino are also supported.

DSA is a model concept that represents Distributed Service Architecture. It represents the interaction of IoT components as a set of fully connected networks. It is a rather complex system to implement and use, but it is interesting from the point of view of interaction management.

DeviceHive is a collection of devices that analyze big data. It requires a device with the Ubuntu kernel. It performs interactions by creating an intermediate gateway for node-to-service communication.

GE Predix service realizes platform as a service (PaaS) for building solutions based on industrial IoT solutions. It is a pay-as-you-go service, has a long

ISSN 18-5 (print)
ISSN 26-22 (online)

history, is the IoT platform, and offers capabilities related to real-time data support and monitoring.

Home Assistant is a mass-market platform written in Python that provides a set of tools for deploying and building the service [15].

This solution combines the possibility of installing an open source server and the wide support of secure communication with popular sensors and components for home automation. It is also well supported by the detailed description of use experience by the community.

Open Connectivity Foundation, also called **IoTivity**, is a development of Intel and Samsung. The service claims to be a leader in organizing data exchange with target devices using a set of common protocols, including REST and Constrained Application protocol.

OpenHAB is a service for building and interoperating solutions as elements of an open source smart home. The requirement for working with such a solution is a JVM. A Java virtual machine is required to run solutions based on such a service.

OpenIoT is also a Java-based solution that interacts with endpoint devices with the ability to build a sensor network, interact, and combine the sensors.

Open Remote is a service intended for automation and is used to connect multiple interfaces. It is of interest for its support of quite complex protocols and the interaction with end sensors. These are OneWare, EnOcean, xPL, Insteon, and X10.

OpenThread is a service focused on devices based on ARM architecture that are connected via IPv6 to a shared network. Microchip, Qualcomm, and AVR solutions are among the supported solutions.

PlatformIO is a Python-based solution that includes a development environment for generating various project building blocks and a set of libraries. The service provides interaction and support with more than two hundred different hardware solutions in the form of separate boards based on Arduino compatible and ARM solutions and allows integration with Eclipse, Qt Creator, and other development tools.

The results of the competitive analysis are presented in Table 2. The comparison is based on the possibility to use the service without payment on a long-term basis and to install an instance of an open source server. It simplifies the decision making about the suitable service based on the specified requirements of service pricing and self-hosting.

The last two solutions are not the services bat the platforms for IoT. The most interesting for practical use are ThingSpeak for simple boards and communication over HTTP and Home Assistant for automation of start home with open source server.

Table 2

Comparison of existing solutions and services for communication of components of IoT system

Service	Is free plan available	Is host an open source
Azure IoT	yes	no
AWS IoT	yes	no
Google Cloud Platform	ended	no
IBM Watson IoT	no	no
Cisco IoT	no	no
ThingSpeak	yes	exists
KAA	yes	exists
Zetta	yes	yes
DSA	yes	yes
DeviceHive	yes	yes
GE Predix	no	no
Home Assistant	yes	yes
IoTivity (OCF)	yes	yes
OpenHAB	yes	yes
OpenIoT	yes	yes
Open Remote	yes	exists
OpenThread	open client	–
PlatformIO	open client	–

7. Proposed models for communication of IoT nodes

At the abstract model level, when considering solutions from the point of view of their interaction in the system, it is possible to distinguish several categories of devices connected to the Internet.

Monitoring and telemetry devices are described by a model in which the data flow is directed more from the device to external nodes. These are devices from manufacturers that monitor and make telemetry decisions via the Internet, including smart thermometers, sensors, and alarms. In this case, the model of interaction is presented as a vertical, directed toward the cloud, due to the presence of a hierarchy in the interaction of related blocks, namely the receiving side and the producing side.

Control devices and actuators are described by a model that assumes bidirectional data flow between combinations of devices. These devices imply the availability of action sets to control actuators and peripherals. In the case of device control, with an emphasis on environmental data, a symmetric channel is provided for bidirectional flow for back-and-forth communication to avoid delays in data exchange. For physical mechanical devices, some critical actions may rely on operator response. Therefore, delays should be minimized. Also included in this category are stand-alone devices with options for selectable functions, operating modes, and remote

startup. An example could be a device for controlling components of a smart home, such as turning on the kettle, starting the laundry, or the engine of a single car. In this case, the interaction model is represented as a bidirectional vertical because of the presence of a hierarchy in the interaction of related blocks, namely the setting side and the executing side.

The active node is described by a new model, which is simultaneously considered as a source and an element of intelligent interaction. This model should not be confused with the smart home coordinator model. Often, the device has elements of artificial intelligence and allows for active connections.

These are devices capable of accessing the Internet on their own initiative, making individual and group calls without the operator's initiation. The autonomy of actions of such devices is high enough to form requests for a human instead of a human. In this case, the model of interaction determines the presence of both controlling and peer-to-peer nodes in relation to the device under consideration. This, in turn, removes the system from the strict vertical hierarchy of interaction, providing for the activity of devices at the same level.

Self-organization of a set of nodes is the model that assumes a model of interaction that takes on the character of swarm interaction and moves from a vertical hierarchy to a horizontal hierarchy, peer-to-peer exchanging data with each other. The modes of interaction are variable and include all systems in which devices can interact horizontally.

The presence of a master node is not denied in this model. This is realized when there is a control device interacting with the others, and the other elements are equivalent in their interactions with respect to each other.

In this case, the presence of a master node is optional. A system consisting solely of peer-to-peer devices is an applicable case and is also used.

8. Proposed technique of creation of service

Among all the services, it is necessary to consider one more important set of solutions. This set of solutions can be quite non-standard. It is necessary to remember that there is such a possibility of building some quite exotic solutions, but at the same time these solutions can find their embodiment for quite different categories of the considered components.

Own server means not only the own instance of server but also the own implementation. This may not be a dedicated server, but a Web server, which will have some generated domain name, not necessarily maintained or readable.

It may be provided through the account of any of the hosting services, including free. In this case, there is no need to access from such a device using an IP address.

In this case, it is accessed by a fixed domain name, which can be long, composite, with subdomains, and unreadable. It does not matter because if this information is placed as a key in such a device, the device will be able to access this server.

Because this is a proprietary solution, it can be implemented for a set of devices as part of the entire infrastructure that will be deployed and may contain elements of internal identification that will be agreed upon as part of the project. This solution solves a complex problem related to the centralization of access.

Popular mail services provide the possibility of using online solutions that can be considered as services. Popular mail services, such as Outlook or Google, include the possibility of using spreadsheet tables, which provide an opportunity with the help of App Script to automate many processes and interaction to build some serverless solutions for processing.

In this case, security issues are left entirely to the postal provider that provides such services and resources. Perhaps, in terms of solutions, this can be justified if it is provided by some corporate properties of such solutions and there is an advantage in this in terms of resources or in terms of labor costs, if in this case they will be minimal due to some solutions that have already been deployed before.

Messenger bots are another interesting way to organize interaction using bots or even clients of popular messengers. In contrast to bots, the use of clients is a full-fledged implementation that can perform authorization as an individual customer and perform the interaction as a normal user of a messenger.

However, in the case of the organization of bots, there is a possibility to interact with such a device directly from the internal implementation of a set of libraries that will interact directly with the services of a particular messenger, and due to the keys and identification elements embedded, there will be the ability to interact with a particular user, which is important.

The advantage of such solutions may be the ease of use when it comes to the use of bed solutions by the end user. In this case, the person who is engaged in building such a solution is eliminated and the user is the user and puts into operation some device from mass production or from a lower price solution, which can be simply connected directly for direct information about some parameters or control possibilities.

ISSN 18-5 (print)
ISSN 26-22 (online)

Combining several methods assumes that such solutions can be combined if there are difficulties with hardware realization of support. If the libraries for such bots for messengers are part of solutions and at the same time are part of a hardware solution, there is a need to implement an approach that provides the possibility of combining individual elements of these ideas and using them together.

It should be concluded that these are completely exclusive and exotic architectures. It is possible to find such solutions in the discussion of the control of individual nodes of IoT systems in the community.

Proposed development steps include the steps of taking and selecting a specific service as part of the solution, some that will be developed, that will then be deployed, and then used. First, it is necessary to determine the ranges of specific hardware solutions, their resources what components will be used as part of such a solution.

It is necessary to understand the scale of such a solution and the number of instances of such a system. For example, how much it will require further scaling after the first steps of its implementation and deployment. It should be determined at the stage of selecting the service that will be used for interaction.

The level of security defines the requirements that can be different for some critical, business-critical, financial solutions and solutions that can be created with some risk of condition changes when buying a ready-made solution, which provides the ability to independently configure and run.

It is definitely important to understand whether there is a possibility to switch to a paid basis, to some kind of paid interaction plan, to be able to scale such a solution, to protect the data, to get much better and more efficient support, and to eliminate some problems that definitely arise when scaling such solutions.

This possibility should be considered on the basis of the specifics of the required solution. It is defined by the model of interaction of such solutions and the set of activities and services of the IoT system. The requirements define choosing a specific system and a specific service that can be used for such interaction, and it is necessary to emphasize this.

Therefore, if for some reason the services discussed in the previous sections are not being used, and if there is a need to organize the solution in this way, then HTTPS can be used to provide the security components, but in this case, it is also necessary to ensure that the endpoint device supports HTTPS.

It is possible to use additional protection with block ciphers for data manipulation, both on the device and not on the receiving side. However, the labor costs of such solutions increase significantly.

9. Practical example of implementation of research results

The first model of the monitoring and telemetry device was implemented using Arduino pro mini based on Atmega328p with the connected peripheral module for network connection. The correctness of the work of the circuit was tested using the considered service ThingSpeak. While the device is the data producer in this model, for such a simple device, the use of HTTP is a possible solution. It helps simplify the process of communication with the service.

The proposed sequence of creation of own service was implemented with the use of a free plan of web service. The communication is implemented as GET requests to auto generated domain name. It helps to use the textual representation of the address of the service to add it as a hardcoded constant to the device. This approach allows the use of existing open source web libraries for the creation of the required processing logic of the service with free web hosting. The web application is capable of interfacing with APIs of massagers and popular communication platforms to simplify the process of receiving information from the simplest device in a convenient user form.

This result is reproducible because there are web hosting with free planes and there is a demand from the community to perform enhanced processing of the information from Arduino or similar simple devices.

10. Discussion

This study considers the possibility of using cheap and popular microcontrollers and free services for the creation of reproducible smart home solutions based on open source projects and convenient ways of interaction. It considers sets of services for the interoperability of hardware solutions as part of smart home elements, as part of the entire IoT spectrum when interacting with each other or with a central controlling device that performs interaction and control.

The categories of devices considered according to the way they interact with each other, as well as the interaction models allocated for them, can be used in the future to decide on the applicability of the interaction service for each individual case and a particular IoT solution.

While ThingSpeak service allows the implementation of communication with simple boards using HTTP and GET requests, for home automation with open source server instances and wide support of communication protocols, Home Assistant is the best choice.

Thus, the selection of the service and components depends on the system requirements and a possibility of maintenance and support of communication protocols.

Conclusions

The practical significance of this research is in the possibility of making a decision about the most suitable set of components for the exact distributed hardware system, selecting the model of communication and the protocol, and finding or creating a service to implement the communication. As part of this work, a study of problems was carried out and a solution was proposed for the remote interaction of nodes, as well as individual small-sized components of built-in IoT systems. Existing methods and problems of communication of devices connected to the Internet are analyzed, as well as possible technical solutions and components in IoT systems.

The proposed classification of the set of components of such systems allows the process of selecting components for the set of initial conditions, considering the requirements of price, the required effort to deploy the system, and the possibility of making changes to the system, including solutions with access to the Internet using low-cost microcontrollers with a small amount of program memory and considering the requirement of limited resources. Models for describing the types of interaction of individual nodes are proposed.

Thus, based on this research, it is possible to decide the most suitable set of components for a distributed hardware system, choose a communication model, a protocol, and prepare the service for communication.

Further research might consider improving the protection mechanisms for IoT services. As part of further research, it is necessary to consider the problems associated with the construction of distributed solutions. For a detailed understanding, it is necessary to consider the entire set of requirements for interaction services, which in turn requires defining the list of the main problems in their construction.

Contribution of authors: goal and tasks formulation, proposing the classification of IoT components, analysis of communication services, description of models, description of method – **Artem Perepelitsyn**; analysis of IoT architectures and protocols of communication, analysis of cybersecurity for IoT, analysis of publications, translation – **Oleksandr Vdovichenko**; project administration, supervision, validation of results, formulation of problem, finding and preparation of publications for analysis – **Vitalii Mikhalevskyi**.

All authors have read and agreed to the published version of this manuscript.

References

1. Esposito, M., Belli, A., Palma, L., & Pierleoni, P. Design and Implementation of a Framework for Smart Home Automation Based on Cellular IoT, MQTT, and Serverless Functions. *Sensors* 2023, vol. 23, no. 9, article no. 4459, pp. 1-16. DOI: 10.3390/s23094459.

2. Plakhteyev, A., Perepelitsyn, A., & Frolov, V. Edge computing for IoT: An educational case study. *Proceedings of 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies, DESSERT 2018*, 2018, pp. 130-133. DOI: 10.1109/DESSERT.2018.8409113.

3. Perepelitsyn, A., Duzhyi, V., Vdovichenko, O., & Zheltukhin, O. Technologies of Embedded Systems Prototyping using Reconfigurable Nodes: Technical Solutions. *Proceedings 2022 IEEE 12th International Conference on Dependable Systems, Services and Technologies, DESSERT 2022*, 2022. 6 p. DOI: 10.1109/DESSERT58054.2022.10018581.

4. *ATmega328P Data Sheet*. Microchip Technology Inc., 2015. 294 p. Available at: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf. (accessed 19.10.2023).

5. Bobrovnikova, K., Lysenko, S., Savenko, B., Gaj, P., & Savenko, O. Technique for IoT malware detection based on control flow graph analysis. *Radioelectronic and Computer Systems*, 2022, no. 1, pp. 141-153. DOI:10.32620/reks.2022.1.11.

6. Kolisnyk, M. Vulnerability analysis and method of selection of communication protocols for information transfer in internet of things systems. *Radioelectronic and Computer Systems*, 2021, no. 1, pp. 133-149. DOI: 10.32620/reks.2021.1.12.

7. Perepelitsyn, A., & Vdovichenko, O. Technologies and Services of Communication for Embedded Systems over Internet. *Proceedings 2023 IEEE 13th International Conference on Dependable Systems, Services and Technologies, DESSERT 2023*, 2023. 6 p. accepted.

8. Lombardi, M., Pascale, F., & Santaniello, D. Internet of Things: A General Overview between Architectures, Protocols and Applications. *Information*, 2021, vol. 12, no. 2, article no. 87, pp. 1-20. DOI: 10.3390/info12020087.

9. Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. *in IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347-2376, Fourthquarter 2015, DOI: 10.1109/COMST.2015.2444095.

10. Omer, A., Khairi, M. Kamran, M. Khan, I. & Kim, K. A Comprehensive Review of Internet of Things: Technology Stack, Middlewares, and Fog/Edge Computing Interface. *Sensors* 2022, vol. 22, no. 3, article no. 995, pp. 1-43. DOI: 10.3390/s22030995.

11. Tetskyi, A., Kharchenko, V., & Uzun, D. Analysis of the Possibilities of Unauthorized Access in Content Management Systems Using Attack Trees. *Proc. PhD Symposium at ICTERI 2018*, Kyiv, Ukraine, May 14-17, 2018, CEUR-WS, vol. 2122, pp. 16-25.

12. Wencheng, Y., Song, W., Masri, S., Nickson, M., Mohiuddin, A., & Craig, V. Biometrics for Internet-of-Things Security: A Review. *Sensors* 2021, vol. 21, no. 18, article no. 6163, pp. 1-36. DOI: 10.3390/s21186163.

13. Stolojescu-Crisan, C., Crisan, C., & Butunoi, B.-P. An IoT-Based Smart Home Automation System. *Sen-*

ISSN 18-5 (print)
ISSN 26-22 (online)

sors 2021, vol. 21, no. 11, article no. 3784, pp. 1-23. DOI: 10.3390/s21113784.

14. Munshi, A. Improved MQTT Secure Transmission Flags in Smart Homes. *Sensors* 2022, vol. 22, no. 6, article no. 2174, pp. 1-15. DOI: 10.3390/s22062174.

15. de la Puente-Gil, Á., de Simón-Martín, M., González-Martínez, A., Diez-Suárez, A.-M., & Blanes-Peiró, J.-J. The Internet of Things for the Intelligent Management of the Heating of a Swimming Pool by Means of Smart Sensors. *Sensors* 2023, vol. 23, no. 5, article no. 2533, pp. 1-14. DOI: 10.3390/s23052533.

Надійшла до редакції 19.10.2023, розглянута на редколегії 30.11.2023

СЕРВІС ДЛЯ КОМУНІКАЦІЇ ПРИСТРОЇВ З ДОСТУПОМ В ІНТЕРНЕТ: АНАЛІЗ ТЕХНОЛОГІЙ ТА МЕТОД СТВОРЕННЯ

Артем Перепелицин, Олександр Вдовіченко, Віталій Міхалевський

Предметом дослідження в даній статті є технології і сервіси комунікації елементів розумного будинку, а також моделі, методи та інструментальні засоби організації взаємодії пристроїв у складі Інтернету речей. **Метою** роботи є спрощення процесу створення сервісу для програмованого кінцевим користувачем пристрою з можливістю зв'язку через Інтернет і спрощення процесу вибору компонентів для таких пристроїв. **Завдання:** провести аналіз вимог до систем IoT та можливих проблем у процесі їх створення; виконувати аналіз вимог безпеки та кібербезпеки для систем IoT; провести дослідження та класифікацію компонентів систем IoT; провести дослідження процесу взаємодії між елементами системи IoT; проаналізувати існуючі протоколи взаємодії для систем IoT; провести аналіз сервісів для організації комунікації для систем IoT; запропонувати моделі взаємодії вузлів системи IoT; запропонувати послідовності створення сервісу; навести приклад практичного застосування результатів. Відповідно до поставлених завдань, були отримані наступні **результати**. Проведено аналіз можливих проблем і вимог процесу створення елементів систем розумного будинку з доступом в Інтернет. Проведено аналіз існуючих архітектур систем Інтернету речей. Розглянуто Edge і Fog моделі побудови систем IoT. Проаналізовано питання кібербезпеки відносно взаємодії пристроїв. За результатами класифікації виділено п’ять типів компонентів для систем розумного будинку. Описано процес зв'язку елементів розумного будинку з сервісом на рівні моделі. Детально описаний процес взаємодії елементів розумного будинку з сервісом. Виконано аналіз існуючих протоколів для взаємодії пристроїв з доступом в Інтернет. Проаналізовано комерційні та відкриті сервіси для комунікації різних типів пристроїв. Запропоновано моделі взаємодії вузлів. Запропоновано послідовності створення сервісу взаємодії пристроїв. **Висновки:** Головним внеском цього дослідження є запропонований метод створення власного сервісу для взаємодії пристроїв з доступом в Інтернет, який орієнтований на використання широко розповсюджених мікроконтролерів. На основі запропонованої класифікації апаратних компонентів для IoT можна спростити процес прийняття рішень щодо вибору, ґрунтуючись на показниках ціни, необхідних зусиллях від кінцевого користувача для налаштування системи та можливості її розширення і подальшої інтеграції з новими більш широкими системами. Проведений аналіз існуючих сервісів дозволяє зробити висновок, що саме сервіс ThingSpeak цікавий для використання в рішеннях на основі пристроїв з найпростішими мікроконтролерами, Home Assistant цікавий для побудови домашньої автоматизації з використанням відкритого коду.

Ключові слова: сервіс комунікації; протокол комунікації; Інтернет речей, IoT; компоненти розумного будинку; модель вузла IoT; ThingSpeak; Home Assistant.

Перепелицин Артем Євгенович – канд. техн. наук, доц., доц. каф. комп’ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.

Вдовіченко Олександр Олександрович – асп., асист. каф. комп’ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.

Міхалевський Віталій Цезарійович – канд. фіз.-мат. наук., доц. каф. комп’ютерних наук, Хмельницький національний університет, Хмельницький, Україна.

Artem Perepelitsyn – PhD, Associate Professor of Computer Systems, Networks and Cybersecurity Department, National Aerospace University «Kharkiv Aviation Institute», Kharkiv, Ukraine,
e-mail: a.perepelitsyn@csn.khai.edu, ORCID: 0000-0002-5463-7889, Scopus Author ID: 56332607800.

Oleksandr Vdovichenko – PhD student, Assistant of Computer Systems, Networks and Cybersecurity Department, National Aerospace University «Kharkiv Aviation Institute», Kharkiv, Ukraine,
e-mail: o.vdovichenko@csn.khai.edu, ORCID: 0000-0001-8695-1752, Scopus Author ID: 58090577500.

Vitalii Mikhalevskyi – PhD, Associate Professor of Department of Computer Science, Khmelnytskyi National University, Khmelnytskyi, Ukraine,
e-mail: cezar61mv@gmail.com, ORCID: 0000-0002-8197-8005, Scopus Author ID: 58099263700.