**Influence of the Internet of Things and its Sub-Domains on Development of Architectural and Design Patterns**

Seminar Field: Software Engineering

Course Number: 20368

Student Name: Yonatan Giventer

Student ID Number: 324699123

Advisor: Professor Shmuel Tyszberowicz

Open University

Table of Contents

1. Introduction3
2. Introduction to Patterns and IoTx
3. Introduction to Patternsx
4. Introduction to IoTx
5. Domains of IoT and their effect on Patterns x
6. General Domain of IoT and Networking Patternsx
7. Bain-Computer Interfaces and Ontology Patternsx
8. Smart Resource Distribution and Computing Loads and Security Patternsx
9. Healthcare and Security Patternsx
10. Composer’s Thoughts x
11. Conclusion x
12. Bibliography x

**1. Introduction**

In this paper the correlation between the advancements in the field of the Internet of Things and the development of new architectural and design patterns are examined.

This paper starts, in Section 1, with providing a brief understanding and introduction to what the two major actors in this paper are, patterns and IoT.

The paper moves on, in Sections 2, to discuss various domains of IoT and new patterns that were developed due to the advancement of those domains. The discussion starts with the general domain of IoT and then delves into a few sub-domains. Each section provides an explanation of the domain and then moves on to the issues that the new patterns were invented to solve, finally ending with the new patterns that were created. Some are more expected than others, such as IoT in general would require new networking patterns but it isn’t as clear the BCI (Brain-Computer Interfaces) would need new ontology patterns without reasonable research done in that field.

In Section 3, the discussion moves on to the thoughts of this paper’s composer about all that has been discovered in the previous two section and speculation on the future of these fields.

Finally, in Section 4, this paper is concluded while going over what the paper covered.

**2. Introduction to Patterns and IoT**

This paper deals with IoT and its sub-domains as well as their effect on architectural and design pattern. Before those subjects can be delved into a basic understanding of the topics is necessary. A short introduction to design/architectural patterns is first in order and then a discussion on IoT will take place. Later, a more in depth examination of a selection of its sub-domains will be looked into.

**a. Introduction to Patterns**

Design patterns were popularized in the field of software engineering in the now famous book by the Gang of Four where a variety of patterns were described. The job of each pattern was to provide a generalized solution for recurring problems. The design patterns described in the book were object oriented patterns and they solved software development issues that showed up in that paradigm such as how listen to an event with the Observer pattern or how to enforce a single instance of class that can be accessed anywhere with the Singleton pattern. In all 24 patterns were listed and categorized into 3 pattern types, Creational, Structural and Behavioral.

These design patterns as well as others for other sources made a great impact on the realm of software engineering and were expanded in existing categories and new ones. Other well-known categories include Concurrency patterns dealing with recurring multi-threaded environment problems such as the Producer-Consumer pattern, Architectural patterns dealing with more macro architecture issues such as how to build a robust and maintainable app such as MVC (Model View Controller). Here the distinction between architectural patterns and design patterns are noted. They are similar in nature and the main difference is level of abstraction [1]. Design patterns are more specific solving smaller scale problems while architectural patterns are more abstract and solve larger overarching problems. In the paper the word “pattern” by itself may be used to refer to either.

With the development of software engineering more specialized patterns were developed, still dealing with providing a general solution to a common problem with the exception that the problem in question was common within a specific domain and not necessarily the larger field of software engineering. How various pattern domains evolved due the introduction and popularization of IoT and its sub-domains is a main topic of interest in this paper.

**b. Introduction to IoT**

The main subject of this paper is the Internet of Things or IoT. In this paper an examination of IoT as a technological field and its effects on software development will take place. To get started first an understanding of what IoT is is necessary. Put simply, it is a field of technology that deals with interconnected devices over a network. Network enabled devices can range from a common electric kettle, that will automatically boil water for you as you wake up, to the cutting edge of green energy wind turbines and everything in between. In fact, it is no exaggeration that the limits of IoT aren’t confined to the Earth itself and has taken flight to the bounds of space.

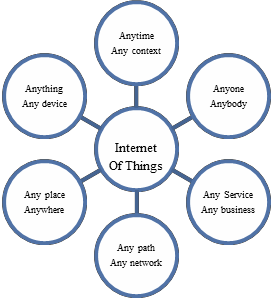


Figure Definition of IoT [10]

IoT is often thought of as a new up and coming technology and although it certainly has experienced incredible growth in the last decade or so it can be seen as far back as the 1980s from before the term was coined in old vending machines. What some see as an internet buzzword some hail as a major component of the fourth Industrial Revolution.

IoT systems come in all different shapes and sizes from a small network in one’s home to entire cities interconnected and further. These all have something in common, that is smart devices and a system controlling them. A more general idea of IoT is various network enabled devices generally including sensors and sometimes actuator (sometimes both in one device) and a control system. Data typically flows from the sensors to the control system where the data is analyzed and monitored then, when needed, instructions are sent from the control system to the actuator devices (if relevant). A very simple example of this would be an air conditioner and a PC. The air condition would have a special chip making it network enabled and it internally would have both a sensors and actuators. The PC would work as the control system and would receive data about the temperature from the AC and might send instructions back to it to change the temperature depending on the time of day. A more complicated system may include motion sensors in the room and sensors along with actuators for the angle of the air flow fins in the AC. The computer could send instructions to redirect the airflow depending on a person’s position in the room.

The above example is quite small scale compared to other where and entire electric supply grid would be interconnected and controlled by AI where all data passes through multiple gateways and middleware that includes many different functions such as smart maintenance detection, power redirection and more. The structure of the network and the components comprising it differs depending on the use case and sub-domain as will become evident in later in the paper. In later sections a few sub-domains of IoT will be introduced and their effects on patterns of software development will be examined.

**3. Domains of IoT and their effect on Patterns**

In the previous section, Section 2, both IoT and patterns were briefly explained as well as their history touched upon. Each topic was talked about separately. The main subject of this paper, however, is seeing how these two interact.

In this section it will be shown that the development of IoT and its branching into various sub-domains brings about the development of new patterns. This occurs because with the development of new technology new recurring problems make themselves evident. With the discovery of new problems, new solutions will be invented. These are the new architectural/design patterns, acting as guidelines for future developers who will undoubtedly, at some point, come across the same problems.

In this section a selection of the domains of IoT, starting with the general domain of IoT itself, will be examined. First, each will be explained and discussed and then problems arriving from these domains as well as their solutions in the form of architectural/design patterns will be presented.

It is noteworthy that, although some of the patterns below are largely novel in concept, many of the new patterns are extended versions or variations of existing patterns that were adapted to fit the new domains.

**a. General Domain of IoT and Networking Patterns**

IoT is a rapidly growing technology with a main focus on data collection and analysis then reacting to the results of this analysis. The data is collected from sensor edge devices and sent over a network to a control center of some sort where the majority of the analysis takes place and instructions may be sent back over a network to actuate on the decisions made from the analysis. Just in this short and simplistic description of IoT it is clear that the network transfer is a major component of IoT. This should be obvious when considering the acronym is for “Internet of Things” implying many devices interconnected within a network. As such, a questions arises. How is all this data sent over a network?

In order to answer the question, another must be asked. Between what parties is data being transferred. Depending on the sender and receiver the transfer method or protocol may vary. To support these needs patterns were developed for various networking types or layers. The layers of interest when discussing IoT are Device to Device (D2D), Network to Network (N2N), Middleware to Middleware (MW2MW), Application and Service to Application and Service (AS2AS), Data and Semantics to Data and Semantics (DS2DS) and CROSS-Layer [2]. This paper will not talk about all the layers here but it is useful to know what layers have been looked into.

Listed below are some of the derived patterns (listed in [2]):

* IoT Gateway Event Subscription pattern. This pattern from the D2D layer dictates that the gateway is used as a subscription mechanism allowing different IoT devices to communicate with each other. Various devices will send messages to the gateway. The devices can subscribe to only certain messages and the gateway will forward messages to subscribes devices of the message type. This decouples the devices as a device does not know what devices if any are listening to its messages, and on the other side a device does not know what devices are send the messages it is listening to.
* D2D REST Request/Response Pattern. Another pattern of the D2D layer in which the IoT devices should be able to communicate using HTTP/REST such that when a request is sent a response will be sent back.
* IoT artifact’s Middleware Message Broker. This MW2MW layer pattern dictates that there will be a common messaging interface between the various components of the middleware. This allows the components to interact regardless of the component’s design. Each component interacts with the broker which, in turn, interacts with the relevant components.
* IoT Artifact’s Middleware Self-contained Message. Another MW2MW pattern for message structure. It states that each message should provide all necessary information needed for a given action. This allows the middleware components to contain only little information and still be able to successfully route and act on the message.
* IoT SSL CROSS-Layer Secure Access. This CROSS layer pattern deals with the systems security. In a broad IoT network, one consisting of multiple layers, all access to layers and interactions between them must be secured. With this pattern each layer will have a public API only through with it can be accessed and interacted with and all such interactions will be done using SSL (Secure Socket Layer).

To answer the questions above, how is the data sent, after examining several new patterns, depending on the complexity of the system, the data may be sent from the edge device to any layer. Each layer may, in turn, relay the data onward. At each station the data will be analyzed to a degree, even if only enough to know where to relay the data to. Each relay, according to the proposed patterns [2] will use SSL to securely interact with the next layer. Finally the data will reach its final destination where what is left to analyzed will be dealt with.

**b. Bain-Computer Interfaces and Ontology Patterns**

BCI, that is Brain-Computer Interfaces, is quite an interesting domain of IoT connecting neurology, electrical engineering and machine learning in addition to some of the more standard aspect of IoT such as network connectivity.

Using various devices are used for sensing electromagnetic waves sent out from the body, and specifically the brain in the form of brain-waves. These devices, such as and electroencephalogram, an EEG, analyze these electromagnetic waves and classify them using machine learning algorithms to train an associated software making it “understand” the intention behind those waves. This is usually done by a device such as an EEG coming into physical or near physical contact with the body. After classification by the machine learning (generally deep learning) software, actuation may take place. [4]

Although this may seem like a futurist idea, perhaps something out of a science fiction novel, real world examples have already been developed. They range in purpose and field, from military to healthcare, not to mention entertainment.

Although healthcare is the main focus of our exploration of BCI in the remainder of this section, a short deviation from that for an example to solidify an image of what BCI can do follows. The following example has the potential to cause an upset in the world of gaming.

The NextMind device with its SDK for the Unity game engine allows users, when wearing the device, to control certain aspects of project created with then Unity game engine simply by concentrating [11]. It works by analyzing the brainwaves of the wearer when concentrating on special graphics displayed on the screen and with integrating the software this can be used to cause a myriad of effects, simplest of which would be perhaps movement in a game. This is not limited to a simple trigger of if the user is concentrating on the visual queue either (binary yes or no), it can analyze the intensity of the concentration. Via software interface any effect can be given to the values read from the brain.

Now that some of the capabilities of BCI are understood the discussion moves on to the field of healthcare, which will be discussed in its own right in a later section, where it becomes evident that the same type of technology isn’t limited to video games. It can be used, for instance, to move around a wheelchair of a patient who does not have the ability to walk. Once the software is trained for the patient and a classifier is created, it can be used to analyze the patient’s brainwaves and interpret how to actuate and move the wheelchair. Here the IoT technology is allowing all the various parts to communicate with each other such as the EEG device, the actuated device (the wheelchair) and the processing unit which may be on a separate device.

To more efficiently work with BCI as a subdomain of IoT ontology, patterns were developed. Here a couple of design patterns are examined (listed in [4]).

* Actuation-Actuator-Effect Ontology Design
* Stimulus-Sensor-Observer Ontology Design

Without going into detail of the tagging and models of the ontology itself, these two patterns work hand in hand cataloging the full picture of BCI model, that is the connection between the Sense Model (described by the Stimulus-Sensor-Observer Ontology Pattern) and the Actuation Model (described by the Actuation-Actuator-Effect Ontology Design). Each pattern is aligned to and expands upon existing ontologies in the relevant fields including, of course, the field of IoT.

In the Figure below the relationship between the two patterns is depicted:



Figure [fill in citation: BCI Ontology: A Context-based Sense and Actuation Model for Brain-Computer Interactions]

**c. Smart Resource Distribution and Computing Loads and Security Patterns**

With the emergence and popularization of IoT, vital resource (i.e. gas, water and electricity) distributors saw a tremendous opportunity in integrating the new technology into their systems. IoT based devices for monitoring and distributing resources were developed [7].

Major components of these system are:

* Smart Meters, the devices responsible for transmitting the state of each the meter in the system as well as actuating on the meter when receiving such a command,
* System specific hardware (sometimes part of the same device as the meter), these are used to actuate on components of the system such as windmills or water pumps.
* Control Centers, the computers where the data is consolidated and analyzed and then decisions (made by computers or people) are made such as changing the state of an edge device.

These meters can be spread out across wide areas from densely populated to sparse rural areas. As will become evident later, certain security issues arise from such setups. Many systems are in fact sub-systems of a larger overarching system.

The responsibilities of such systems may include [9]:

* Load balancing. This lowers maintenance requirements by monitoring the edge devices and sending data to a control center. That center will compare the loads and send command back to the edge devices adjusting the loads. The edge devices will then actuate on the hardware, adjusting it to the specifications provided by the control center.
* Load diversion. This lowers lowers the impact of failures. By monitoring the state of edge devices in an interconnected Smart Meter system, if a failure is detected control can channel the flow of the resource from a different source. In such cases the end user may never know that a failure took place.
* Failure and maintenance alerts. This too lowers maintenance demands as the edge device will alert the command center of failures or when it requires maintenance. Instead of routinely doing maintenance for all devices, only those that need attention can be pinpointed and dealt with.
* Efficacy evaluation. This can save great costs on the system. A simple example of this is making sure that solar energy farm providing as much energy as possible by tilting the solar panels to follow the sun to generate as much energy as possible. This can greatly increase the benefit of the benefit to cost ratio.

As one of the most varied resources, smart resource management systems for electricity is specifically mentioned. Such a system is commonly referred to as a Smart Grid. A plethora of types of Smart Grids exist around the world, dealing with different energy sources. Aside traditional sources such as fossil fuel Smart Grids for green energy sources such as wind and solar have been a major source of development as well. In fact, several such such Smart Grids may work together forming a larger grid deriving energy from various sources [9].



Figure Real-time generation monitoring from an IoT-based control center [9]

As mentioned above, there is a major security issue with smart resource or smart meter systems. In addition to networked cyber-attacks, the threat of which all network systems face, in this case there is a fear of physical attacks [3]. Some of the meters, which communicate with the control center, are spread out in vast areas and cannot feasibly be constantly monitored. As such, they are left open to physical attacks. The threat in question is not simply damaging the devices, instead a great danger is maliciously tampering with them causing intentional false readings and sending false information to the control center.

Three elements were identified that needed to be secured as listed in [3]:

* Booting, prevents unauthorized devices from functioning.
* Communications, prevents unauthorized communication.
* Firmware updates, prevents unauthorized firmware from being installed.

Though none of these can physically safeguard the device, together they prevent a device that has been tampered with from hurting the system aside from that particular device being rendered not functional.

To address these three elements one method proposed in [3] utilizes design patterns both existing and new. With these patterns, solution to the three elements above are provided using a combination of public key cryptography where time isn’t vital (booting and firmware updates), stateless authentication where time is of the essence (regular communication) along with offloading heavy security calculations so as not to harm the real-time nature of the edge devices.

These new design patterns developed by this method:

* Computation Offloading: Due to the nature of the edge devices heavy computational tasks would harm their efficacy, especially for real-time devices, therefore these tasks will be offloaded to trusted gateways where the heavy computation will take place.
* Stateless Authentication of edge devices: Similar to existing stateless authentication such as in some smartphone apps, the device will use a token for authentication and no session data will need to be stored.

As interesting and relevant aside to note that the idea of Computation Offloading, similar to what is seen in the example above is a key aspect of “Fog”. Fog is middle ground between the cloud and the edge device. Heavy computations on the edge device are taxing on it and hurt the capability of being a real-time device as stated above. On the other hand, if all heavy computations are done on the cloud then latency will become an issue, again harming the real-time nature of the system as a whole. This is where fog comes into play, if utilized, heavy computations will be offloaded to the fog which will work as a local gateway to the cloud. If need be the fog device will interact with the cloud. [14]

**d. Healthcare and Security Patterns**

Healthcare has been radically transformed in recent years with the integration of IoT technologies. Various aspects of the healthcare domain have been affected from app connected to peripherals monitoring one’s basic vitals while exercising, counting the amount of steps a person walked throughout the day, week or month and so much more. Although a lot of these may seem like they are for the utility of generally healthy users allowing them to keep track of themselves, that only scratches the surface of what is now possible in this domain.

Using IoT systems hospital staff can monitor at-risk patients either on site or even from the comfort of their homes. Capturing data of general vitals such as pulse or blood pressure as well as more case specific data such as brain function, network enabled devices can be worn by, or sometimes implanted in, patients where the data is collected and sent out either directly to experts or to a management center. The collected data is analyzed and can sent out alerts a situation warrants. Given a relevant situation a healthcare professional may give instructions to an on or off site patient using speakers, meaning that a patient at home may be able to receive immediate medical advice from a professional when they need it. [13]

These healthcare management centers may also employ the use of prediction models trained using artificial intelligence algorithms on dig data to attempt to predict diseases and doing so help in reaching effective treatments.



Figure IoT healthcare architecture [13]

All this advancement, though positive overall, bring a new set of challenges to the healthcare practitioner and patients alike. Although patient is the main focus, one of the larger concerns from the practitioner’s perspective is that the data’s quality may be called into question as noise may added during the transfer. This can be mitigated with better architecture and noise removal techniques.

From the patient’s perspective very different types of questions emerge and take the spotlight of the conversation. Some of the new solutions require patients to give up a certain amount of confidentiality to allow their data to be processed and used by the systems which may cause doubt and suspicion on the side of the patients. Some systems require patients be tech-savvy to a certain degree. Generally, these are considered minimal but for some patients who do not use much technology at all, this can be quite the hurdle. Another concern is that some patients simply distrust technology on and security or stability. These can cause great discontent among patients when presented with new healthcare system. A prime example would be healthcare for the elderly. The elderly is demographic that is not used to dealing with technology and find it intimidating and untrustworthy. Coupled with the fact that the elderly as a demographic require more attention from the healthcare systems, a demand to better the situation arose.

To address these issues a number of safeguards can be setup. First of all, patients may specify what information they will share as well as who has access to it and how that information will be used, as such making the system more trustworthy to the patients. To enforce this requirement, the software run in the healthcare systems must support this link between information and role and only allow access to those with an allowed role. Exiting design patterns used for this are the Role Based Access Control pattern and the Contextual Based Role Based Asses Control. Both of these deal with giving access to specific information based on the role of the accessor and the context of the access (in our case the context may be an emergency and so access will be provided even for roles not generally authorized if the situation allows it).

The second point of interest regarding the trustworthiness of the the system is the sensors themselves and the data they gather. As mentioned above many different sensors may be used in healthcare systems be them wearables, implants or any other type of sensor. These sensors collect data from the patient which is later used for analysis. In order to protect this data, the Sensor Design Pattern was introduced. Including authorization techniques, it protects the data within the sensor from being access by unauthorized parties. In addition, it dictates that minimal data is stored in the sensor devices themselves, most of the data stored being what is necessary to authentication and collected data is sent into the system and stored there. This means that even if the sensor device is stolen, past collected data cannot be extracted from it.

**4. Composer’s Thoughts**

Throughout the time preparing and writing this paper, the composer has consumed many articles on various fields pertaining to IoT in one way or another. The paper that sparked the original idea for this paper (thought that idea has evolved over time) was [1] where a survey was performed and patterns were examined. The finds most relevant to this paper were the patterns discovered that were new IoT patterns rather than non-IoT patterns in use in IoT development. Some of these patterns are listed in this paper along with details about the sub-domain that spawned them.

The composer’s interests were greatly peaked seeing how new patterns were developed for various fields. After doing research on the new patterns and seeing that they sometimes originated from seemingly unrelated domains of IoT such as ontology patterns being developed due to development in field of BCI (Brain-Computer Interfaces). As one who was familiar with BCI and indeed the NextMind device mentioned in this paper, but not with ontology patterns this connection alone was enough to solidify the composer’s desire to write this paper.

During the research for this paper it was made evident that some of the new patterns that were brought about by the domains of IoT were very similar to existing patterns, many based off parent patterns but altered or combined in a new way creating new patterns that better fit the new domain. In [2] in the description of the new patterns the base patterns are listed. In other papers as well although not as clearly we see that some patterns are extensions or alterations of existing patterns such as both Computation Offloading and Stateless Authentication mentioned in Section 2.c. The core idea behind both exist outside the realm of IoT but the new patterns doctor them making them more compatible for the new use case.

It occurred to the composer that this is to be expected. A fundamental principal of software engineering is that new development is made of build-blocks of existing ideas. It seems that is the principal under which these new patterns were developed when possible. Just a software engineer generalizes and compartmentalized pieced of code for general use, such that these pieces of code may be reused in way that even the original programmer or architect did not image, the same is with the very patterns that govern the best practices of code being programmed.

Another note that the composer found interested though was not directly linked to any of the subjects above was also found in [1]. The paper found that although there are many new IoT patterns, only a select few were mentioned in more that on paper that was used in the survey. This implies that, although new patterns are being developed, it is uncommon for those unfamiliar with the new pattern to find said pattern and use it.

The composer finds this fact as quite unfortunate as many could benefit from the patterns developed by others. That being said, these patterns are new, from a historical standpoint, and it is the opinion of the composer that with time, more and more of the new patterns will make their way around the community of IoT developers and software engineering as a whole. Just as so many patterns have become prevalent throughout the field in the past, it is the opinion of the composer that the same will happen with IoT patters, probably sooner rather than later.

**5. Conclusion**

In this paper it has been shown how, although the Internet of Things, isn’t technically a new technology, its rapid development in the last decade or so has drastically changed the current day landscape of technology. The development revolutionary systems such as Smart Grids and video games or wheelchairs that are literally controlled by the mind will have profound consequences on a variety of sectors.

In this paper, the point was made and demonstrated that development of IoT technologies both in general and in its sub-domains has brought forth new patterns of various fields, from networking to security and more.

Several cases have been shown how, given a problem or problems in a relatively new sector, such as Smart Resource Distribution, or an existing sector with new problems, such as care for the elderly, new patterns have been created to solve said problem. In some cases, the new design patterns were closely related to existing ones, yet altered in a way to better fit the new domain, while other deviate more from establish patterns to work better with new problems that need solving.

In the last section, Section 4, where the composer’s thoughts were given, it was mentioned that although some problems recur, many design patterns developed to solve said problems are not well known throughout the field. It is the composer’s opinion that as time moves on, more information will naturally be shared and the new patterns will become more wide spread. In addition, it is the composer’s opinion that the realm of IoT is the clear path for the future and in fact a great future for IoT and in turn for technology and all who benefit from it is unfurling even now that these words are being written.

**6. Bibliography**

[1]  WASHIZAKI, Hironori, et al. Landscape of architecture and design patterns for iot systems. IEEE Internet of Things Journal, 2020, 7.10: 10091-10101.‏

[2] TKACZYK, Rafal, et al. Cataloging design patterns for internet of things artifact integration. In: 2018 IEEE International Conference on Communications Workshops (ICC Workshops). IEEE, 2018. p. 1-6.‏

[3] NTULI, Nonhlanhla; ABU-MAHFOUZ, Adnan. A simple security architecture for smart water management system. *Procedia Computer Science*, 2016, 83: 1164-1169.‏

[4] MÉNDEZ, Sergio José Rodríguez; ZAO, John K. BCI Ontology: A Context-based Sense and Actuation Model for Brain-Computer Interactions. In: *SSN@ ISWC*. 2018. p. 32-47.‏

[5] PERIYASAMY, Kasi; ALAGAR, Vangalur; WAN, KaiYu. Dependable design for elderly health care. In: 2017 Federated Conference on Computer Science and Information Systems (FedCSIS). IEEE, 2017. p. 803-806.‏

[6] QANBARI, Soheil, et al. IoT design patterns: computational constructs to design, build and engineer edge applications. In: 2016 IEEE first international conference on Internet-of-Things design and implementation (IoTDI). IEEE, 2016. p. 277-282.‏

[7] LLORET, Jaime, et al. An integrated IoT architecture for smart metering. IEEE Communications Magazine, 2016, 54.12: 50-57.‏

[8] MOCNEJ, Jozef, et al. Decentralised IoT architecture for efficient resources utilisation. IFAC-PapersOnLine, 2018, 51.6: 168-173.‏

[9] SHAHINZADEH, Hossein, et al. IoT architecture for smart grids. In: 2019 International Conference on Protection and Automation of Power System (IPAPS). IEEE, 2019. p. 22-30.‏

[10] SOUMYALATHA, Shruti G. Hegde. Study of IoT: understanding IoT architecture, applications, issues and challenges. In: 1st International Conference on Innovations in Computing & Net-working (ICICN16), CSE, RRCE. International Journal of Advanced Networking & Applications. 2016.‏

[11] UYANIK, Cihan, et al. Brainy Home: A Virtual Smart Home and Wheelchair Control Application Powered by Brain Computer Interface. In: BIODEVICES. 2022. p. 134-141.‏

[12] BLOOM, Gedare, et al. Design patterns for the industrial Internet of Things. In: 2018 14th IEEE International Workshop on Factory Communication Systems (WFCS). IEEE, 2018. p. 1-10.‏

[13] SELVARAJ, Sureshkumar; SUNDARAVARADHAN, Suresh. Challenges and opportunities in IoT healthcare systems: a systematic review. SN Applied Sciences, 2020, 2.1: 1-8.‏

[14] CHIANG, Mung; ZHANG, Tao. Fog and IoT: An overview of research opportunities. IEEE Internet of things journal, 2016, 3.6: 854-864.‏