Improving the Mobile Edge Computing Architecture Using Fog Computing Environments

Alexey Subbotin
St. Petersburg State
Electrotechnical University "LETI"
St. Petersburg, Russia
Alesu1543@gmail.com

Nataly Zhukova
St. Petersburg Institute for
Informatics and Automation of the
Russian Academy of Sciences
St. Petersburg, Russia
Nazhukova@mail.ru

Petr Glebovskiy
EdgeSoft LLC
St. Petersburg, Russia
Pietr@mail.ru

***Abstract*—This article discusses the Mobile Edge Computing (MEC) architecture. A broad overview of ready-made platforms for IoT systems is presented. Examples of real use of ready-made systems in various fields are considered. The problem of delays in data processing is identified. Nowadays reduction in delays require large computing resources. A solution is proposed based on the modification of the MEC architecture using fog computing environments and cloud computing. An approach for transferring computations to the cloud and fog has been developed. The proposed approach was tested and the results were obtained in relation to the processing of a large amount of data (video information). Specific examples are given.**

I. Introduction

The Mobile Edge Computing (MEC) architecture was first proposed in 2014 by the European Telecommunication Standards Institute (ETSI) and was targeted at mobile devices. In 2017, this architecture was extended to the entire IoT stack [1] with support of various virtualization systems (Docker containers, virtual machines, etc.). The ability to process large volumes of information at the border (Edge Computing) is demanding in the throughput of devices, both short-range and long-range, that is why the MEC algorithm is focused primarily on 5G networks, but the principle of distributed information processing on the edge (smartphones, tablets, etc.) is also used in other networks: RAN, LTE (eNodeB), 3G (RNC), RAT.

Edge computing occurs precisely at the level of smartphones, tablets and all neighboring devices that support distributed computing software [2]. Examples of such applications are well-known to everyone: Skype, Torrent, etc., also hardware and software for such calculations is provided, for example, Intel Smart Edge based on the Open Network Edge Services Software platform, OpenNESS (https://www.openness.org/), which was renamed to Mobile Edge Computing (MEC) in 2019.

Distributed edge computing helps reduce the load on the IoT stack devices by sharing computing resources and distributing the load across the private network, they allow reduce latency when working with cloud systems. Edge computing involves processing information only at the level of smartphones and other devices [3], but there are also possibilities of calculations in a fog environment [4].

The term fog computing was coined by the Cisco CTO in 2011 and finalized in 2012. The principle of fog computing is similar to edge computing, but information processing takes place on a high-performance server in close proximity to users who interact with it through mobile devices. It is supposed to connect all kinds of sensors in addition to smartphones, smart video cameras, quadcopters and other devices of the Internet of Things stack. This is the main difference between Fog Computing and MEC. Combining the two approaches will significantly increase the speed and accuracy of information processing and reduce delays [5].

An applied task for such combined distributed computing can be processing of video information from video cameras with provisioning of statistics and diagnostic information. The approach can be used when all the necessary sensors, cloud servers for data storage are connected to the fog environment, fog calculations are organized, and data preprocessing takes place on the edge [6].

This approach is applicable in a variety of areas: retail, industrial IoT systems [7] where machines and devices of the Internet of Things operate [8], medicine, where miniature video cameras in smart glasses, smart watches, smartphones and others are widely used [9]. Programs based on this technology are capable of recognizing faces, displaying information on displays (for the buyer) and on devices (for the seller). Such systems can now be found in popular fast food retail chains: McDonalds, KFC, Burger King, Dodo Pizza and many others, where information is transferred from a smartphone to the MEC system, and the customer receives his order at the entrance to the restaurant. The seller's screen displays the name of the customer, other parameters, and the screen in the dining room shows the name and the predicted ready time.

The advantages of such systems from a technical point of view are the following:

- higher performance of local computing in real time;
- more productive usage of computing resources;

ISSN 2305-7254

• protected operating environment;

• adaptive local and networking storage;

• ease of deployment and use;

• zero trust security and policy enforcement;

• faster response and less delays.

Edge computing is now widely demanded, its application in various fields allows reduce the time of interaction between employees, the cost of work. It is widely used in our daily life in restaurants, industry, medicine and public transport [10].

II. Background

The main difference between edge (MEC) computing [11] and Mobile Cloud Computing (MCC) is that information processing is executed directly on devices (smartphones, smartwatches, robots, quadcopters, technological computers, etc.), distributing the load in a single computer network, the devices also interact with cloud services [12], [13], [14]. The parameters (Table I) of edge and cloud computing are very similar, but, nevertheless, there are differences, which will be described in detail.

TABLE I. Comparison of MEC and MCC systems

Parameter	MEC	MCC
Server hardware	Small-scale data centers with moderate resources	Large-scale data centers (each contains a large number of highly-capable servers)
Server location	Co-locate with wireless gateways, Wi-Fi routers, and LTE BSs	Installed at dedicated buildings, with size of several football fields
Deployment	Densely deployed by telecom operators, MEC vendors, enterprises, and home users. Require lightweight configuration and planning	Deployed by IT companies, e.g., Google and Amazon, at few locations over the world. Require sophisticated configuration and planning
Distance to end users	Small (tens to hundreds of meters)	Large (across the country)
Backhaul usage	Infrequent use. Alleviate congestion	Frequent use. Likely to cause congestion
System management	Hierarchical control (centralized/distributed)	Centralized control
Supportable latency	Less than tens of milliseconds	Larger than 100 milliseconds
Applications	Latency-critical and computation-intensive applications, e.g., AR, automatic driving, and interactive online gaming	Latency-tolerant and computation- intensive applications, e.g., online social networking, and mobile commerce/health/learning

Seeing the perspectives of the MEC concept and following the changing market, companies that provide cloud services began to develop platforms that facilitate the deployment and management of MEC systems.

An example of such a system is Microsoft Azure private MEC, which includes a network manager, a core for corporate edge (support of devices, sensors, smartphones, etc.), security based private secure network, support of Kubernetes clusters, and Azure Stack Edge component that enables edge computing from devices. The presence of this component allows the developer to perform distributed MEC computations in their programs, where it is no longer required to develop a system for sharing the computational load between devices of the Internet of Things stack [15], [16], [17], [18]. The Microsoft Azure private MEC solution is a software platform where various applications can be located.

Other example of a virtual platform for mobile edge computing [19], [20], [21], [22] is Intel Smart Edge (https://www.intel.ru/), which works on a similar to Microsoft Azure Stack Edge principle. Smart Edge virtual platform supports two software components, not three as competitor Azure Stack Edge. The first component of the platform is the MEC controller that allocates resources. It consists of five modules:

• Management of hardware and software components of all MEC nodes

• Configuring all additional services, libraries and components (API)

• Management of settings of programs and operating systems (Windows, Linux, FreeBSD, MacOS, iOS, Android, Symbian OS 9.4, etc.) of each node (node)

• Capacity planning and reporting for all system components based on MEC

• Granting access roles (RBAC) and rights to users from the snap-in (EPC), which is developed by Intel for centralized management of devices and the MEC system.

All five modules are managed through the Resource & Design Center after registering on the corporation website (https://www.intel.com/) for Intel customers and partners.

The second component of the platform is the MEC peripheral node, which provides local network information exchange, stores small programs. It consists of the following modules:

• Ensuring interoperability without encryption

• Communication management

• The use of local applications for solving not general, but specific tasks

• Setting rights and security policies on the devices

• Support for wired and wireless devices.

The third example of MEC systems is Amazon Simple Storage Service (Amazon S3), which is well suited for IoT stack applications and for analyzing big data using machine learning [23], [24], [25], [26], [27], [28]. It provides simple management tools (https://aws.amazon.com/ s3/). The advantages are scalability, reliability, high performance, availability, S3 Intelligent-Tiering economical storage system based on S3 Outposts, support for security and data protection systems: FISMA, PCIDSS, HIPAA/HITECH, FedRAMP. Amazon Athena data access service supports the standards: SQL and Amazon Redshift. AWS Partner Network (APN) support is being introduced.

The Amazon S3 software toolkit is designed specifically for IoT stack applications. The toolkit includes: S3 Object Lambda to provide data to the application via S3 GET requests, S3

Storage Lens to work with cloud storage, S3 Intelligent-Tiering to optimize storage, S3 Access Points to create an access point (with VPN support), S3 Batch Operations to manage billions of objects and create jobs. The toolkit supports AWS Storage Gateway backup, disaster recovery (DR), archiving with the integrated S3 Glacier Deep Archive tool, big data lake analysis with AWS Lake Formation tool, hybrid storage based on AWS tools (Transfer Family, DataSync, Storage Gateway, PrivateLink, etc.) that support the most popular protocols (SFTP, FTPS and FTP), applications for cloud optimization for IoT: S3 REST API, AWS S3 access management and security, S3 data durability, S3 storage classes, S3 storage management and monitoring.

The services described above are used successfully by large companies to solve their business problems:

Tampnet Oil & Gas Solutions (https://www.tampnet.com/), which allows the management of offshore oil and gas rigs;

Goldcorp mining industry (https://www.newmont.com/);

Adidas sports shoes and apparel and etc.

Large companies can allocate budget and staff to deploy systems in new conditions to accelerate the processing of existing data streams, however, the use of such systems for experimental or scientific purposes is redundant [29], [30].

Therefore, when solving problems within the framework of this article, it was decided to use the main idea of the MEC concept, such as transferring computing from the cloud closer to the data source, but without using paid MEC solutions from large providers.

III. Description of the problem

The main problem of IoT and IIoT systems based on MEC using the concept of distributed computing is that data processing is executed precisely on the device edge, and not in a fog, first of all, and not in the cloud [31]. The performance of such systems can be significantly increased using fog computing environment (Fig. 1) [32].

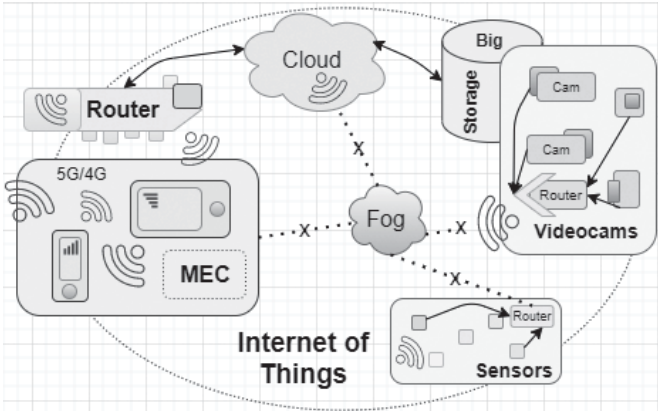

Fig. 1. The problem of not using a fog computing environment

An applied task is the processing of video information (Fig. 1), which is demanding on the computing resources.

The problem of delays and accuracy of object detection can be shown on a specific task. The St. Petersburg Subway has a system of assistance (http://www.metro.spb.ru/) for people with limited mobility, those who are lost and everyone who needs help, as they themselves decided. The system consists of interconnected parts: a kiosk, a cloud server, video surveillance system, microphone and speakers. On the screen in the kiosk there is an animated menu using which you can find out the necessary information: the police, the ambulance, the history of the subway and other materials.

The built-in video camera in the kiosk, located above the screen, transmits frames in real time for processing. The absence of fog computing causes delays in the exchange with the cloud, the resolution of the transmitted video is unacceptably low.

The absence of a mobile application from the IoT stack [33] often does not allow provide assistance to the passengers in time, if necessary. The incompleteness of the MEC concept and the absence of fog computing shows the obsolescence of the assistance and monitoring system that is currently used in the subway. The disadvantages of the system include:

- Lack of mobile phone support
- Outdated way of rendering
- Slow image processing
- No fog computing environment
- High latency when accessing the cloud.

The system requires modernization in accordance with new trends in Industry 4.0 with full support of mobile edge computing.

IV. Solution method

The solution of this problem is to design a system with new architecture using fog computing environment. To overcome all the shortcomings of the current system, not only a new structure is needed (Fig. 2), but also software development, which should include three mandatory components and one additional component:

- Management program in the fog, which deals with load balancing in the network
- A program in the cloud that directly processes video information by means of machine learning
- A program for smartphones, designed for easy visualization of statistics
- Updating the site in the kiosk using new technologies (additional component)

Fog clouds [34], [35], [36], [37], [38] are configured in a group to receive and transmit information through routers (in Fig. 2, right) to the cloud (top) from a desktop computer, laptop, phone, technology computer in a kiosk (left in the figure). The information transmitted to the cloud is placed in the storage (top right corner).

By experts and management six performance indicators (Table II) for the smart surveillance system embedded in the kiosk-based help system were derived.

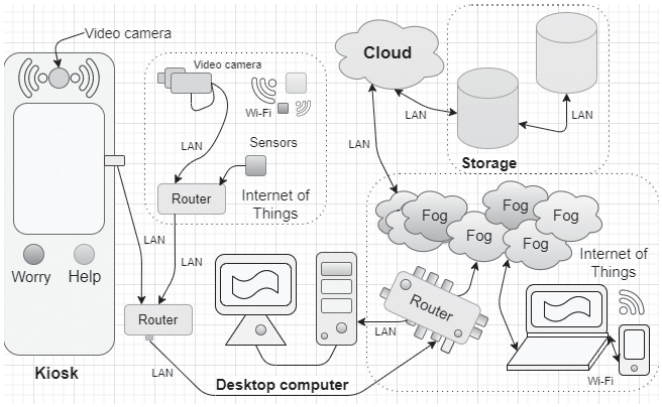

Fig. 2. Modernized system structure

TABLE II. EXPECTED INDICATORS

Index	Result
Reduced delays	3 times
Increased speed of information processing	8 times
Improved accuracy of object detection	10%
Prompt decision making	20%
Satisfaction with the system	35%
Accuracy of management decisions	15%

The architecture of the new system consists of a fog computing environment that is the focal point in the solution. The intelligent observation and assistance system is based on the signal and scheduling theories. The essence of the signal theory is that when a signal from a device (sensor, video camera, help buttons, etc.) arrives to the fog computing environment, then a certain decision is made: transmission of the signal-state to the phone or to the computer of the operator. The scheduling theory is used to determine the workload, busyness of cloud systems and select the appropriate server from the list of available servers. Thus, a highly deterministic system with a high degree of reliability is obtained.

According to the TOGAF architecture description standard, the first Business level envisions a strategy to reduce injuries and accidents across the organization. At the Data level, a description of the interaction in the system at the data level is assumed. We do this according to the principle of signals and scheduling theory, the information is processed on the server and stored in the cloud. At the Application level, applications interact with employees and management through mobile applications that display the current situation. The Technology layer is implemented using Java technologies to communicate over the network. To support interaction with the cloud programs written in Phyton and Qt are used.

The description of the system according to the ISO 15288 standard is available, where the target system is a mobile application, the system in the operational environment is the kiosk software, the supporting system is the cloud and fog computing environments.

According to the DoDAF 2.0.2 architecture description standard, an operational view is video-based information for an intelligent video surveillance system operator, a system view is kiosk software, and a technical view considers the following components: cloud technologies, fog computing environment, a mobile application, and a PC site.

V. APPLICATION DESIGN

After creating a new architecture based on fog computing, the software was developed, which consists of three interconnected modules:

1) Control program (signal processing, decoding of a video stream into groups of frames, network transmission of packets and a system for encrypting traffic and archiving packets for sending to the cloud);

2) A program for direct processing of video information (receiving connections via an encrypted VPN channel from the list of trusted servers, unpacking an archive with decryption, processing a group of frames, transferring the result to a fog environment);

3) Visualization programs on the smartphone and on the kiosk monitor (the current state of the system and status signals about the events occurring in the subway are shown, if assistance is required).

The first program was developed in Java SE 16.0.2 (https://www.oracle.com/java/) in Apache NetBeans 12.2 (https://netbeans.apache.org/). Java works well with the network and is cross-platform, therefore it works equally well under different operating systems (Windows, Mac OS X, Linux, Solaris). Java performance is not much different from binary code, although it works in the interpreter. First, all the code is saved in XML format and packed into a JAR archive, at the moment the file is opened, the scripting language is compiled into bytecode and then it works no slower than the bytecode created in C and C++, C#, Delphi, Lazarus, Visual Basic, etc., and sometimes even faster, thanks to various optimizations at compile time, which makes Java an unrivaled tool in network programming. The only thing is that the program hangs when you first start it since it needs to compile. But it is almost invisible on fast computers, which are used in fog computing environment.

The second program was developed in the Qt Creator 4.15 environment (https://www.qt.io/product) with the connection of libraries for working with the network. It provides possibility of machine learning using TensorFlow 2.5.0 (https://www.tensorflow.org/). But the best solution would be a separate installation of the Keras 2.4.0 framework (https://keras.io/), which contains several machine learning libraries at once: TensorFlow, Microsoft Cognitive Toolkit, Deeplearning4j, and Theano. By applying deep machine learning and all the other capabilities that libraries provide, it is possible to significantly improve the accuracy of object detection in video, but this approach requires considerable

computing resources that are only available in the cloud. A combination of convolutional neural networks (CNN), deep learning (DL), recurrent neural networks (RNN), temporary additional training of the model, the presence of an expert system for selecting frames of successfully found and unsuccessfully found objects, careful filling of the dataset, the use of various libraries increases the accuracy of determining objects and working in the cloud improves the speed of video processing.

The third program was developed in Android Studio 4.1.3 (https://developer.android.com/studio) using the Dart 2.13.1 language from the search giant Google (https://dart.dev/). Programs for mobile devices developed in the Android Studio environment are cross-platform and can be compiled for different operating systems (Android Linux, Windows Mobile, iOS, Chrome OS, etc.), which greatly saves time on application development. The new site that runs on the kiosk's embedded computer is developed using PHP 7.0.33-0 (https://www.php.net/), JavaScript 1.7 on the client side from the browser (https://www.javascript.com/), JQuery 1.8.0 library (https://jquery.com/) for compatibility with Bootstrap 5.1.0 (https://getbootstrap.com/), which provides advanced styling options (styling buttons, menus, etc.). As a Web server, nginx 1.14.1 and Linux operating system are used with differentiated access rights for the developer, users and clients via the Web site via the HTTP/HTTPS protocol.

All programs interact with each other using signals. When a certain signal is received, the first program decides to issue a certain state to the program for visualization (display on a monitor screen through a website in a kiosk or in a mobile application, or on a PC screen).

VI. Case Study

An application was developed consisting of four parts: a control program, two programs for visualization, a program for working in the cloud, necessary for processing video images.

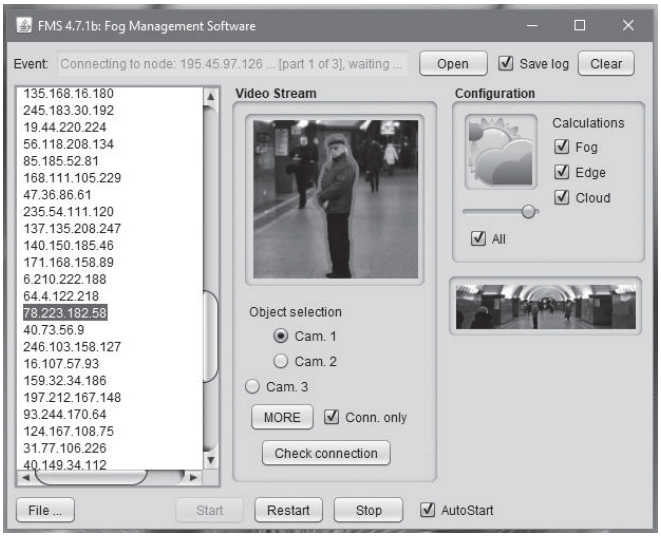

Fig. 3. Control program in a fog environment

The control program (Fig. 3) consists of a list of IP addresses to which the connection is made (on the left). It is possible to load a list of IP addresses from a text file using the "File ..." button. The last successfully found object is shown in the center, and a high-quality panoramic image obtained from the video stream is shown on the right. You can select a source (switches: Cam. 1, Cam. 2, Cam. 3) and make additional settings using the MORE button. Check the connection by clicking the "Check connection" button and connect to all cameras that are currently in the network (check the "Conn. Only" checkbox). There is a choice of server types for data processing: fog, edge, cloud and all (menu "Configuration" with the image of clouds). The basic functions of the program are controlled by three buttons: start of the process, restart, stop with the possibility of autostart (checkbox "AutoStart").

According to the experimental conditions, the video image is passed to the fog server from three Panasonic HC-V730 cameras expandable to five in Full HD (Full High Definition) format with a resolution of 1920 × 1080 pixels. In the fog environment (Cisco UCS C480 ML M5) three streams are decoded at a frequency of 12 frames per second (expandable up to 24), packed into a ZIP archive using the GZip 1.8 program in Astra Linux 2.12.40 Common Edition with additional information in XML format. Group of frames are sent to a group of servers running under Ubuntu 20.04 LTS for subsequent unpacking and processing. Characteristics of a cloud from a group of four Yandex.Cloud servers are the following: CPU with four Intel Broadwell cores with Nvidia Tesla v100, a graphics processor with NVIDIA Tesla V100 RAM with tensor cores, 96 GB RAM with the ability to increase or decrease by task, network 4,000 GB SSD storage with the ability to be expanded to a larger size if needed.

TABLE III. Reduction of the delays

Time	Before	After	Result
Monday, from 6 a.m. to 12 p.m.	6032 ms.	1527 ms.	3.96
Tuesday, from 10 a.m. to 2 p.m.	5497 ms.	1372 ms.	4.01
Tuesday, from 2 p.m. to 7 p.m.	5213 ms.	1683 ms.	3.1
Tuesday, from 7 p.m. to 10 p.m.	7068 ms.	1538 ms.	4.6
Tuesday, from 10 p.m. 11 p.m.	5017 ms.	1429 ms.	3.52
Wednesday, from 6 a.m. to 12 p.m.	5139 ms.	1372 ms.	3.75
Average:			**3.82**

After testing (Table III), the delay in image processing decreased 3.82 times.

The program for visualizing events from the system (Fig. 4) is implemented in two versions, one of which is in the form of an application for a smartphone. In case of an event, a frame with the found object appears under the ATTENTION label. Below are the buttons: ERROR, ATTENTION, ACCEPT to acknowledge the event, error and for further consideration. At the very bottom is the RETURN button, which allows return to the main menu of the program settings. The current time of the event is shown in the lower right corner.

Fig. 4. Event visualization program

TABLE IV. IMPROVEMENT OF ACCURACY OF OBJECT DETECTION

Name of the object	Algorithm 1	Algorithm 2	Result
Mothers with strollers	67.92	79.51	11.59%
Disabled people	73.41	83.71	10.3%
Aged people	71.3	84.77	13.47%
Children	64.19	73.75	9.56%
Animals	68.7	78.9	10.2%
Blind people	72.51	85.58	13.07%
Average:			**11.365%**

Thanks to the application of the cloud computing, the accuracy of object detection (Table IV) has increased by 11.4%, which is a good result, although it was assumed to be 2 times more.

The program for processing a group of frames (Fig. 5) in the cloud checks the connection, establishes an incoming VPN connection, checks the security settings, receives the packet with frames and displays it in the log (upper part of the picture). It is possible to clear the log using the "Clear" button and save it to a file that can be viewed on the Web page: http://rss2fido.sourceforge.net/video-log.php in the form of a table of events ordered by time and date with the ability of additional filtration.

The main control buttons are located under the log: process start, restart, stop and the ability to start the process when the program starts (checkbox "Run at startup"). The ability to search the log with a choice of the type of event is provided. It is allowed to load the list of trusted servers from a file by clicking the "Choose ..." button and choosing additional, optional parameters.

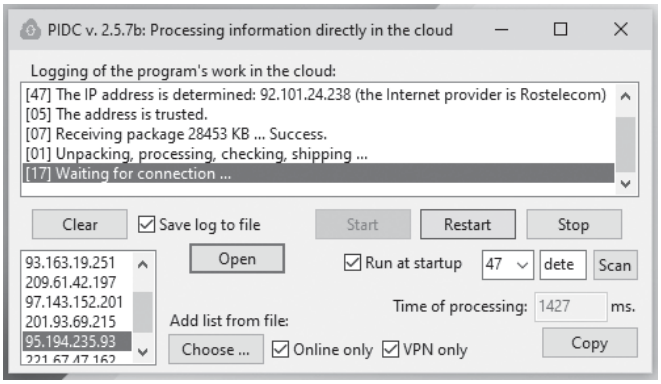

Fig. 5. A program that runs directly in the cloud

TABLE V. INCREASE OF THE SPEED OF INFORMATION PROCESSING

Time	Fog	Cloud	Result
Monday, from 6 a.m. to 12 p.m.	4824 ms.	581 ms.	8.31
Tuesday, from 10 a.m. to 2 p.m.	5291 ms.	536 ms.	9.87
Tuesday, from 2 p.m. to 7 p.m.	4952 ms.	627 ms.	7.9
Tuesday, from 7 p.m. to 10 p.m.	4702 ms.	554 ms.	8.49
Tuesday, from 10 p.m. 11 p.m.	4691 ms.	537 ms.	8.73
Wednesday, from 6 a.m. to 12 p.m.	5192 ms.	564 ms.	9.2
Average:			**8.75**

Thanks to the use of fog computing environments, it was possible to increase the processing speed (Table V) of information by 8.75 times compared to a PC, although it was supposed to be 8 times.

TABLE VI. PROMPT DECISION MAKING

Service	Earlier	Now	Result
Fire department	12-17 min.	10-15 min.	23.7%
Ambulance	5-10 min.	3-7 min.	41.28%
Police	2-3 min.	1-2 min.	31.04%
Help for disabled	15-30 min.	10-15 min.	23.5%
Electricians	20-40 min.	15-30 min.	15.7%
Energy	30-55 min.	25-45 min.	12.63%
Communication	15-25 min.	12-20 min.	9.41%
Average:			**22.47%**

Thanks to the new system, the efficiency of decision-making (Table VI) increased by 22.47%, although no more than 20% was expected.

TABLE VII. SATISFACTION WITH THE SYSTEM

Position	Per	Against	Neutral	Divergence	Result
Controller	71	23	4	2	32.39
Cashier	63	31	3	3	49.2
Help for disabled	87	9	3	1	10.34
Security guard	74	13	6	7	17.6
Escalator operator	69	19	8	4	27.54
Visitors	57	36	4	3	63.16
Electrician	52	41	2	5	78.9
Average:					**39.86%**

Satisfaction with the system (Table VII) turned out to be higher than expected and amounted to 39.86% versus expected 35% . A standard psychological questionnaire consisting of 100 open-ended questions was used to assess the attitude towards the system. The interview was passed through Google forms online in their free time (https://forms.google.com/).

TABLE VIII. ACCURACY OF MANAGEMENT DECISIONS

Solution	1 mon.	3 mon.	6 mon.	Result
Dismissal	0,003	0,52	0,68	17%
Retraining	0,052	0,36	0,42	8%
Rebuke	0,09	0,41	0,57	19%
Meeting	0,27	0,89	0,91	14%
New projects	0,002	0,37	0,42	15%
Recruitment	0,0734	0,46	0,49	26%
Average:				**17%**

The accuracy of managerial decision-making can only be assessed by a manager. The standard Google calendar (https://calendar.google.com/) was used to record management decisions and subsequent analysis. Despite the fact that it was planned to increase the accuracy of management decisions (Table VIII) by 15%, it turned out that it is 17%, that is more than expected.

VII. CONCLUSION

The article discusses methods for remote processing of video images using cloud technologies and fog computing environment in relation to mobile computing. Due to using fog computing latency was reduced in 3.8 times.

The authors of the article have developed an application for visualizing events on a smartphone for processing video images on virtual machines using cloud technologies and fog computing environment. The stages of the program operation are considered, separate details are disclosed. Specific examples of the program's work in relation to the MEC subject area and finding objects in the video images are given.

The task of processing video images by means of fog calculations is also relevant for solving other problems. For example, using the subway system for building the process of pattern recognition can allow recognize events on escalators and track areas. The developed program can also be used not only in the subway, but also in airports, shopping centers and other public places.

ACKNOWLEDGMENT

The authors express their gratitude to The State Unitary Enterprise "Petersburg Subway" (http://www.metro.spb.ru/), The Foundation for Assistance to Small Innovative Enterprises (FASIE), SPIIRAN (http://www.spiiras.nw.ru/) for scientific support and the opportunity to create an article.

The research was supported by the state budget (project No. 0060-2019-0011 and supplementary contract No. 198GS1CTNTIS5/64231).

REFERENCES

[1] Bonomi F., Milito R., Natarajan P., Zhu J.: Fog Computing: A Platform for Internet of Things and Analytics, pp. 169–186. Springer International Publishing, Cham (2014)
[2] M. Chiang and T. Zhang, “Fog and IoT: An overview of research opportunities,” IEEE Internet Things J., vol. PP, no. 99, pp. 1–1, 2016.
[3] G. P. Fettweis: The tactile Internet: Applications and challenges, IEEE Veh. Techn. Mag., vol. 9, no. 1, pp. 64–70, Mar. 2014.
[4] Aazam M., Huh E.: Fog computing and smart gateway-based communication for cloud of things. In: 2014 International Conference on Future Internet of Things and Cloud, August 2014, pp. 464–470 (2014)
[5] Bhatia J., Patel T., Trivedi H., Majmudar V.: Htv dynamic load balancing algorithm for virtual machine instances in cloud. In: 2012 International Symposium on Cloud and Services Computing, pp. 15–20. IEEE (2012)
[6] Rahm E., Do H.H.: Data cleaning: problems and current approaches. IEEE Data Eng. Bull. 23, 2000 (2000)
[7] Saecker M., Markl V.: Big Data Analytics on Modern Hardware Architectures: A Technology Survey, pp. 125–149. Springer, Berlin (2013)
[8] Sarkar S., Chatterjee S., Misra S.: Assessment of the suitability of fog computing in the context of internet of things. IEEE Trans. Cloud Comput. 6(1), 46–59 (2018)
[9] Yannuzzi M., Milito R., Serral-Graci R., Montero D., Nemirovsky M.: Key ingredients in an IoT recipe: fog computing, cloud computing, and more fog computing. In: 2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), December 2014, pp. 325–329 (2014)
[10] Huang C., Lu R., Choo K.-K.R.: Vehicular fog computing: architecture, use case, and security and forensic challenges. IEEE Commun. Mag. 55(11), 105–111 (2017)
[11] Yuyi Mao, Changsheng You, Jun Zhang, Kaibin Huang, and Khaled B. Letaief: A Survey on Mobile Edge Computing: The Communication Perspective. IEEE Communications Surveys & Tutorials PP (99):1-1, August 2017, Pages 37-51 [https://doi.org/10.1109/COMST.2017.2745201].
[12] Tanwar S., Tyagi S., Kumar N.: Multimedia Big Data Computing for IoT Applications: Concepts, Paradigms and Solutions, vol. 163. Springer (2019)
[13] Jaykrushna A., Patel P., Trivedi H., Bhatia J.: Linear regression assisted prediction-based load balancer for cloud computing. In: 2018 IEEE Punecon, pp. 1–3. IEEE (2018)
[14] Bhatia J., Dave R., Bhayani H., Tanwar S., Nayyar A.: Sdn-based real-time urban traffic analysis in vanet environment. Comput. Commun. 149, 162–175 (2020)
[15] Bhatia J., Modi Y., Tanwar S., Bhavsar M.: Software defined vehicular networks: a comprehensive review. Int. J. Commun. Syst. 32(12), e4005 (2019)
[16] Dastjerdi A.V., Gupta H., Calheiros R.N., Ghosh S.K., Buyya R.: Fog computing: principles, architectures, and applications. CoRR, abs/1601.02752 (2016)
[17] Liu Y., Fieldsend J.E., Min G.: A framework of fog computing: architecture, challenges, and optimization. IEEE Access 5, 25445–25454 (2017)
[18] Dragi Kimovski, Roland Math: Cloud, Fog or Edge: Where to Compute? IEEE Internet Computing 2101.10417, 1–8 (2021)
[19] Khaled Matrouk, Kholoud Alatoun: Scheduling Algorithms in Fog Computing: A Survey. International Journal of Networked and Distributed Computing 9(1); January (2021), pp. 59–74.
[20] Zeeshan Ali, Shehzad Ashraf Chaudhry, Khalid Mahmood, Sahil Garg, Zhihan Lv, Yousaf Bin Zikria. A clogging resistant secure authentication scheme for fog computing services. Computer Networks. Volume 185, 11 February 2021, 107731 [https://doi.org/10.1016/j.comnet.2020.107731].
[21] Ahmad Raza Hameed, Saif ul Islam, Ishfaq Ahmad, Kashif Munir. Energy- and performance-aware load-balancing in vehicular fog computing. Sustainable Computing: Informatics and Systems. Volume 30, June 2021, 100454

[https://doi.org/10.1016/j.suscom.2020.100454].

[22] Vasileios Karagiannis, Stefan Schulte. Distributed algorithms based on proximity for self-organizing fog computing systems. Pervasive and Mobile Computing. Volume 71, February 2021, 101316 [https://doi.org/10.1016/j.pmcj.2020.101316].

[23] Wanchun Dou, Wenda Tang, Bowen Liu, Xiaolong Xu, Qiang Ni. Blockchain-based Mobility-aware Offloading mechanism for Fog computing services. Computer Communications. Volume 164, 1 December 2020, Pages 261-273 [https://doi.org/10.1016/j.comcom.2020.10.007].

[24] Mandeep Kaur, Rajni Aron. Energy-aware load balancing in fog cloud computing. Materialstoday: Proceedings. 17 December 2020. [https://doi.org/10.1016/j.matpr.2020.11.121].

[25] A. Subbotin, N. Zhukova and T. Man, "Architecture of the intelligent video surveillance systems for fog environments based on embedded computers," 2021 10th Mediterranean Conference on Embedded Computing (MECO), 2021, pp. 1-8, doi: 10.1109/MECO52532.2021.9460270.

[26] Raafat O. Aburukba, Taha Landolsi, Dalia Omer. A heuristic scheduling approach for fog-cloud computing environment with stationary IoT devices. Journal of Network and Computer Applications. Available online 4 February 2021, 102994 [https://doi.org/10.1016/j.jnca.2021.102994].

[27] Maria Gorlatova, Hazer Inaltekin, Mung Chiang. Characterizing task completion latencies in multi-point multi-quality fog computing systems. Computer Networks. Volume 181, 9 November 2020, 107526 [https://doi.org/10.1016/j.comnet.2020.107526].

[28] A. N. Subbotin, "Data Processing in Foggy Computing Environments for Machine Learning," 2021 II International Conference on Neural Networks and Neurotechnologies (NeuroNT), 2021, pp. 51-53, doi: 10.1109/NeuroNT53022.2021.9472203.

[29] Fenghe Wang, Junquan Wang, Wenfeng Yang. Efficient incremental authentication for the updated data in fog computing. Future Generation Computer Systems. Volume 114, January 2021, Pages 130-137 [https://doi.org/10.1016/j.future.2020.07.039].

[30] Changhao Zhang. Design and application of fog computing and Internet of Things service platform for smart city. Future Generation Computer Systems. Volume 112, November 2020, Pages 630-640 [https://doi.org/10.1016/j.future.2020.06.016].

[31] Masoumeh Etemadi, Mostafa Ghobaei-Arani, Ali Shahidinejad. Resource provisioning for IoT services in the fog computing environment: An autonomic approach. Computer Communications. Volume 161, 1 September 2020, Pages 109-131 [https://doi.org/10.1016/j.comcom.2020.07.028].

[32] A. N. Subbotin, "Applying Machine Learning in Fog Computing Environments for Panoramic Teeth Imaging," 2021 XXIV International Conference on Soft Computing and Measurements (SCM), 2021, pp. 237-239, doi: 10.1109/SCM52931.2021.9507120.

[33] Sunday Oyinlola Ogundoyin, Ismaila Adeniyi Kamil. A Fuzzy-AHP based prioritization of trust criteria in fog computing services. Applied Soft Computing. Volume 97, Part A, December 2020, 106789 [https://doi.org/10.1016/j.asoc.2020.106789].

[34] Hadi Zahmatkesh, Fadi Al-Turjman. Fog computing for sustainable smart cities in the IoT era: Caching techniques and enabling technologies - an overview. Sustainable Cities and Society. Volume 59, August 2020, 102139 [https://doi.org/10.1016/j.scs.2020.102139].

[35] Julian Bellendorf, Zoltán Ádám Mann. Classification of optimization problems in fog computing. Future Generation Computer Systems. Volume 107, June 2020, Pages 158-176 [https://doi.org/10.1016/j.future.2020.01.036].

[36] Xincheng Chen, Yuchen Zhou, Long Yang, Lu Lv. Hybrid fog/cloud computing resource allocation: Joint consideration of limited communication resources and user credibility. Computer Communications. Volume 169, 1 March 2021, Pages 48-58 [https://doi.org/10.1016/j.comcom.2021.01.026].

[37] José Santos, Tim Wauters, Bruno Volckaert, Filip De Turck. Towards end-to-end resource provisioning in Fog Computing over Low Power Wide Area Networks. Journal of Network and Computer Applications. Volume 175, 1 February 2021, 102915 [https://doi.org/10.1016/j.jnca.2020.102915].

[38] Arash Bozorgchenani, Simone Disabato, Daniele Tarchi, Manuel Roveri. An energy harvesting solution for computation offloading in Fog Computing networks. Computer Communications. Volume 160, 1 July 2020, Pages 577-587 [https://doi.org/10.1016/j.comcom.2020.06.032].