Development of a Remote Monitoring System of a Personal Computer User Actions

Vyacheslav Petrenko
vip.petrenko@gmail.com

Fariza Tebueva
fariza.teb@gmail.com

Sergey Ryabtsev
nalfartorn@yandex.ru

Nikolai Svistunov
svistunovn4@gmail.com

North Caucasus Federal University, Stavropol, 355009, Russia

Abstract

One of the important components of a teacher's work during classes with personal computers (PC) usage is the monitoring of students' activities at workplaces. In this paper, the architecture of the remote monitoring system of PC user actions is proposed. The structure and operation algorithms of the client and server modules of the system, as well as client-server communication principles, are described. In accordance with the developed architecture, a software implementation of the system is made. The proposed architecture allows to extend the functionality of the application by developing of additional modules without making significant changes to the existing program structure.

Keywords: user monitoring software, application, process monitoring, client-server architecture, server, client, personal computer, programming.

1 Introduction

One of the important components of a teacher's work during classes with personal computers usage is the students' activities monitoring. The objectives of the monitoring are:

- tracking the students' progress in the tasks performance;
- identification of difficulties arising in the process of tasks performing;
- detection of software malfunction;
- prevention of misconduct in the workplace and misuse of PC resources.

Nowadays electronic monitoring systems are widely used in the medicine, Internet of Things [Ver18], control of equipment and power systems [Kul19], but they are rarely used in education. The article [Dey19] discusses a system of monitoring students' attendance to improve their performance, but does not consider monitoring

In: Jože Rugelj, Maria Lapina (eds.): Proceedings of SLET-2019 – International Scientic Conference Innovative Approaches to the Application of Digital Technologies in Education and Research, Stavropol – Dombay, Russia, 20-23 May 2019, published at http://ceur-ws.org

student performance during the class itself. Thus, the urgent task is to develop software providing remote simultaneous monitoring of the PC users actions.

The main areas of control can be attributed to the desktop and running processes monitoring.

Desktop monitoring is one of the main ways to control user actions. It is carried out by the administrator in real time, either by viewing the stored images or videos.

Since desktop monitoring does not provide full information on running programs, it is necessary to introduce processes monitoring. The monitoring system should keep track of running applications and obtain various data, such as launch time, running time, etc. This allows to evaluate the efficiency of user's working time, as well as to identify the malicious or unwanted software activity. Process of the control also implies the ability to remotely terminate processes by the administrator command and use "black lists" of unwanted applications. Existing software tools that allow to control the PC user actions can be divided into three main classes:

- remote administration tools;
- employee monitoring and time tracking systems;
- DLP systems.

Remote Administration Tools (RAT) are used to remotely manage PCs or other devices. Usually they provide such features as file transferring, user actions monitoring, system configuration, input/output control, etc. Such applications are used to access the resources of another computer (for example, to control a work computer from a home one), as well as for providing remote technical support.

Examples of RAT are LiteManager [Lit19] and Radmin [Rad19]. These applications allow to monitor user activity in real time, but do not have the ability to save information about user actions (screenshots, process history) for deferred viewing. Also there is no possibility to set lists of prohibited processes.

Employee monitoring and time tracking systems provide advanced functionality in the field of user actions monitoring, such as:

- staff time accounting;
- evaluation and monitoring of staff performance;
- disloyal employees and fraudulent schemes identification;
- information leaks detection;
- data analysis for information security (IS) incidents investigation.

The architecture of such systems consists of a server, an administrator's console, a database (DB) for storing the collected information and modules installed on user PCs. In most cases, employee monitoring and time tracking systems implement one of the following concepts:

- collection of a large amount of data, the formation of analytical reports, tracking productivity of working time;
- monitoring employees by taking screenshots, recording videos or watching live broadcasts, monitoring violations;
- only the accounting of working hours.

Examples of employee control systems are Kickidler [Kic19] and Stakhanovets [Sta19]. Such software tools in most cases require the purchase of an expensive license.

DLP-systems are software solutions for preventing leakage of confidential information from an information system to the outside, or technical devices (software or hardware) that implement such functionality. Software of this class in most cases allows to control the PC users' actions. For example, in the paper [Tri12] concept of the system to protect children's access to information in schools using the DLP-system is described. In [Gar15] DLP-system effectiveness is shown in online cyber security competition. DLP-systems in general are described in [Mor18]. However, they are designed to solve a wider class of tasks aiming to ensure the organization's information security. As a result, DLP-systems have a high cost and complexity in installation and configuration. They are excessive for solving the tasks of controlling the actions of students.

Thus, the disadvantages of existing solutions determine the feasibility of developing and implementing software for monitoring the PC users' actions. The purpose of this work is to develop an architecture that can be used to implement software of this type.

2 Task

The functional purpose of the user's actions control system assumes the presence of a client module (CM) installed on controlled PCs and a server module (SM) installed on the administrator's PC (Fig. 1).

It is necessary to consider the following principles of software development:

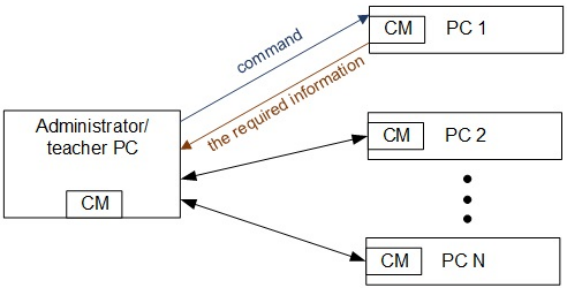

Figure 1: Computer connection diagram

- the use of a modular architecture to simplify software development and maintenance;
- parallel data processing in order to increase productivity and achieve rational usage of modern PCs resources [Nem13, Sod10].

In accordance with the previously reviewed areas of control, the CM must have the following capabilities:

- capturing images of the controlled PC desktop with subsequent transfer to the server module and saving to the database (DB);
- transferring the list of running processes to the server module;
- remotely stop running processes;
- processing the prohibited applications list.

SM should have the following capabilities:

- simultaneous displaying screenshots as miniatures;
- demonstrating full-sizes image of the desktop of a specific PC, selected by the administrator;
- displaying the list of processes running on the selected PC;
- stop a process on a remote PC;
- commands formation and transmission to the client module.

The target platform for the developed software is a PC based on x86-compatible CPU with Microsoft Windows 7/8/8.1/10 OS installed.

3 Development Of Methodology

3.1 Client Module

The structure of the CM (Fig. 2) is due to its functional purpose and includes the following components: communication module; database module; data collection module; user actions monitoring modules; control module.

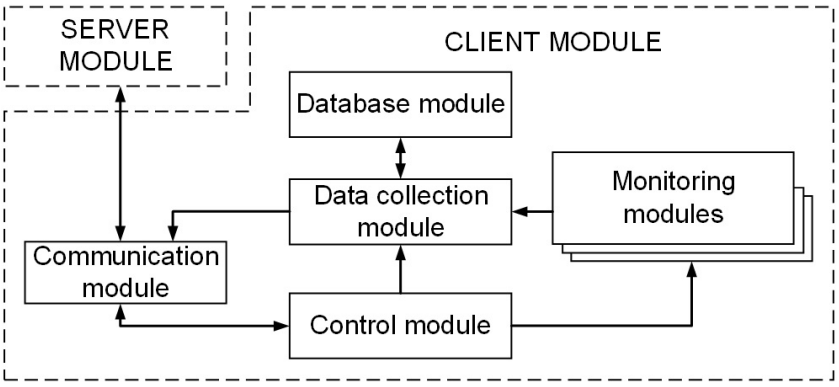

Figure 2: The structure of the client software module to monitor PC user actions

The principle of CM functioning is presented in fig. 3 in the form of state diagram. When the CM is started, the control module is initialized and attempts to read the settings from the registry. The lack of settings in the registry is a sign of the first run of the program performed by the administrator. In this case, the administrator

must set an access password to the settings dialog, and also specify the server IP address and port (the PC on which the CM operates).

After establishing a communication session, the data collection module starts to transmit information about the user's actions to the server module. When receiving a command from the server, the control module produces its processing and execution, or sends a command to another module.

It is assumed that the CM starts automatically when the user logs in the system and stops working when the session ends. In this case, the user does not have privileges to terminate the program, change the priority of the CM process, access to the program files and registry keys that store parameters. Compliance with these requirements can be ensured by the appropriate setting of user rights and autorun parameters.

User actions control modules are designed to obtain data on user actions.

From the object-oriented programming point of view, each actions control module is a class and contains the following elements:

- the construction set, that allows to set the initial data necessary for operation of the module, and carrying out preparatory actions;
- field for storing a pointer to the data collecting module;
- a set of fields for storing the parameters of the module, intermediate and auxiliary data;
- methods of starting and stopping the module;
- methods by which the functionality of the module is implemented and its configuration is made.

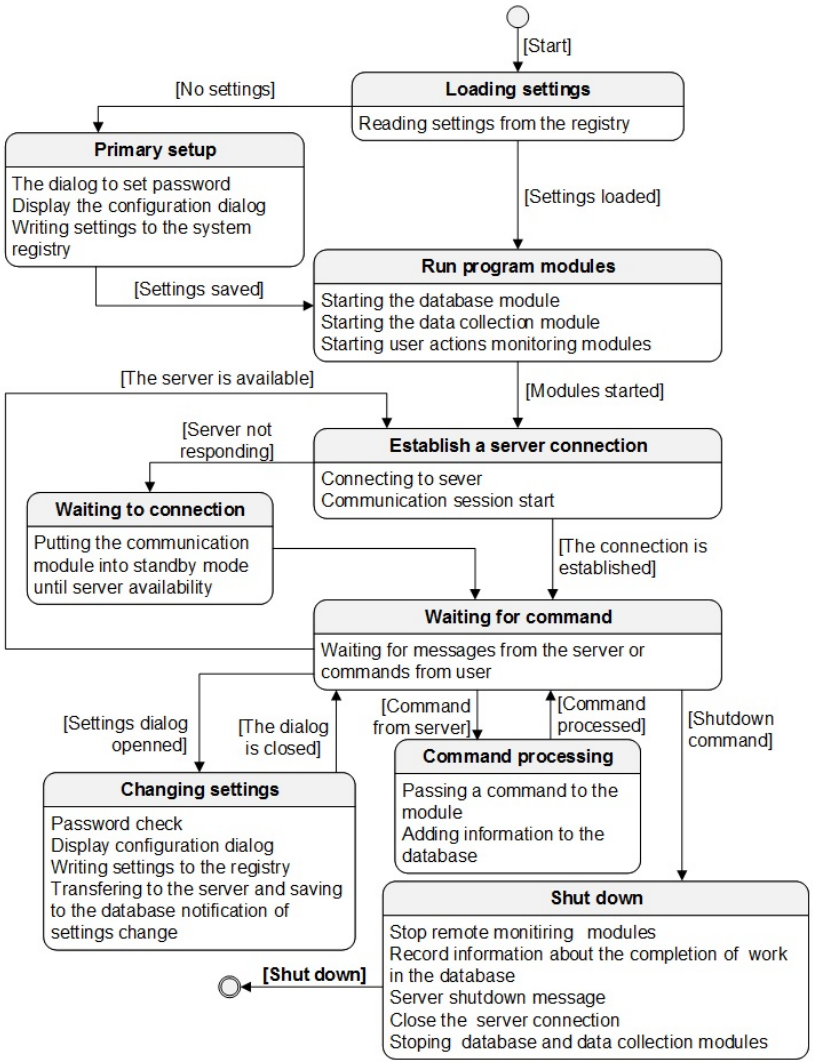

Figure 3: CM state diagram

The constructor of each class corresponding to the user actions control module takes as arguments the pointer to the data collecting module, as well as a set of parameters defining the mode of operation. A parameter required for each action control module is the time interval for sending messages to the data collecting module.

Let us consider in more detail the purpose and principles of functioning of the main modules for controlling user actions.

3.2 Desktop Monitoring Module

The desktop monitoring module is designed to collect screenshots at a specified frequency and then transfer them to the data collection module. The module state diagram is shown in fig.4.

Taking screenshots and transferring them to the data collection module is carried out in a separate thread created by the constructor.

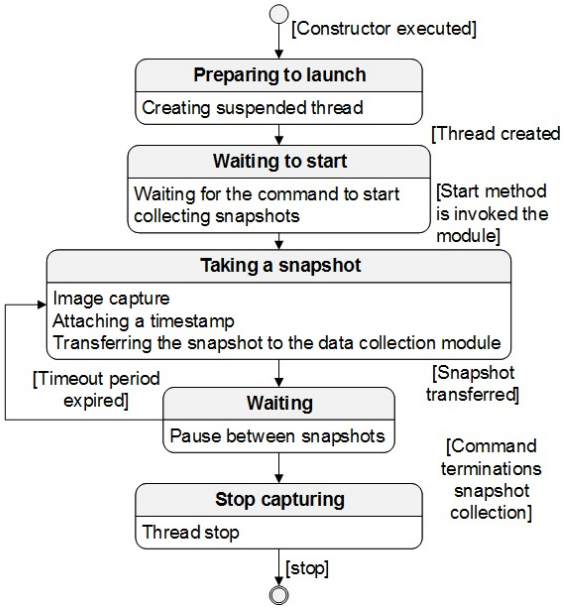

Figure 4: State diagram of the desktop monitoring module

The process of collecting screenshots can be stopped only when the application is shutting down as a result of the user's logout.

3.3 Process Control Module

The process control module is intended to periodically collect information about running processes, detecting prohibited processes, remote start and end processes upon an administrator's command.

The module state diagram is shown in fig. 5. When an instance of the class corresponding to this module is created, the constructor creates a thread for gathering information about processes, process start thread, processes stop thread, as well as a prohibited processes control thread.

The thread of collecting the information about processes is intended for the periodic generation and transfer to the data collection module of a list of running processes. The thread operation algorithm is presented in fig. 6. If the parameters for detecting prohibited processes are set, the process list is also transferred to the control thread of prohibited processes.

The collection may be subject to such information as:

- process name;
- process identifier;
- the percentage of CPU time consumed;
- the amount of used RAM;
- the user on whose behalf the process is running;
- the path to the file;

- process description.

The control thread of prohibited processes has two modes of operation:

- “black list” mode - processes whose names are on the list of prohibited processes are prohibited;
- “white list” mode - processes whose names are missing from the list of allowed processes are prohibited.

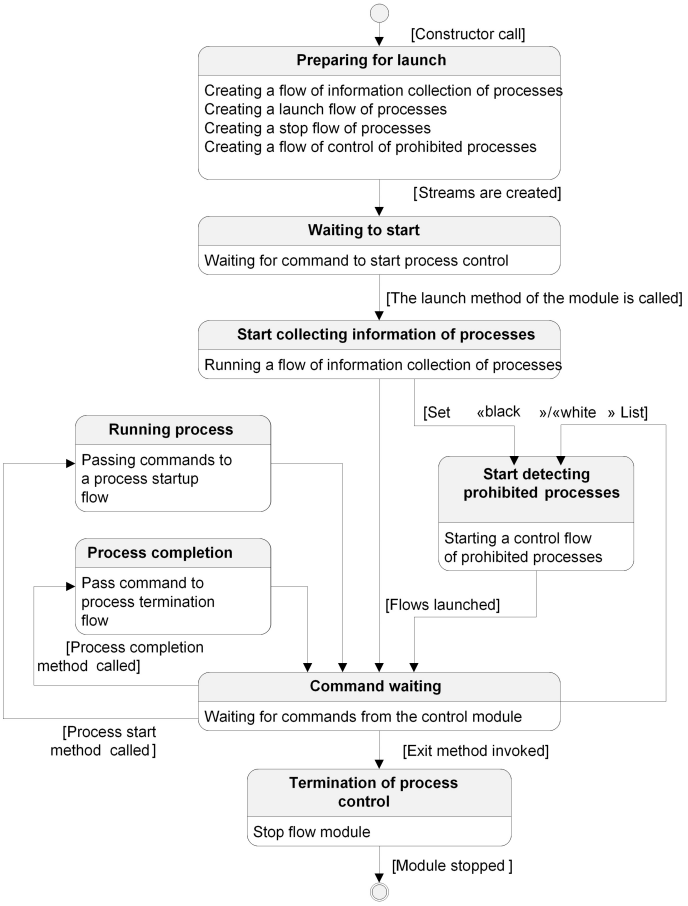

Figure 5: State diagram of the process control module

The mode operation parameter of the control thread of forbidden processes and the list of forbidden / allowed processes can be transferred both to the construction set when creating the object, and later set by calling the appropriate method.

For each process from the “black list” , it should be possible to establish a list of actions that will be performed when it is detected. These actions include:

- entering information into the database;
- sending a message to the server module in order to draw the administrator’s attention to the fact of violation;
- automatic completion of the process.

The list of actions can also be defined for all prohibited processes in general (both for the "black" and for the "white" list). In this case, action settings for each individual process complement the general settings.

Starting and stopping processes can take quite a long time. In order to avoid delays in the operation of threads associated with performing these operations, it is advisable to implement separate threads for starting and stopping processes.

The process launch thread is activated upon receipt of an application launch command from the SM. The process stop thread can be activated by a command coming from both the SM and the control thread of prohibited processes.

There is a possible situation when it’s necessary to start or terminate several processes on the client computer at once. For this case, it should be possible to transfer the entire list of processes in one message.

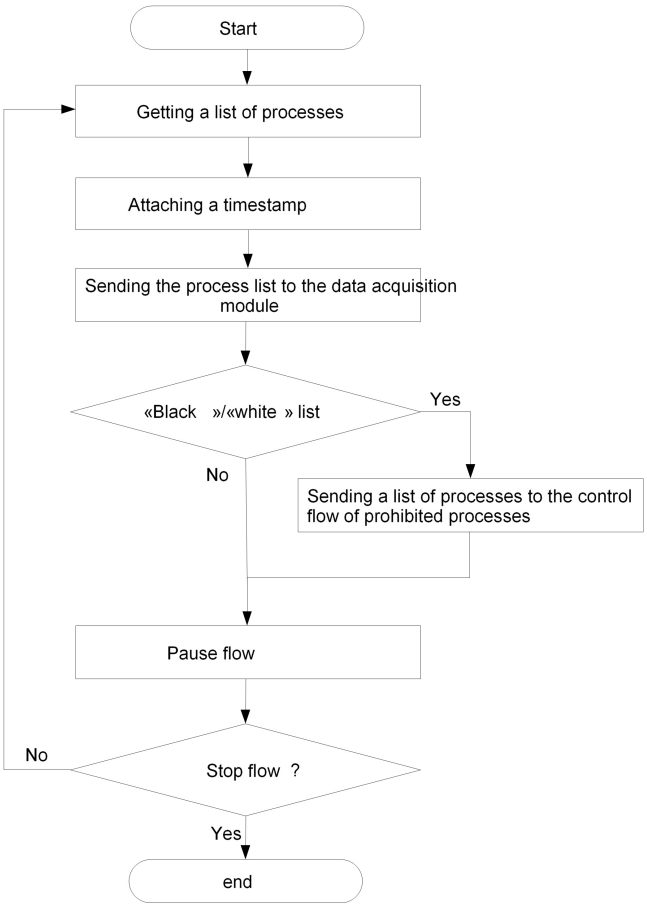

Figure 6: Flowchart of the thread collecting information on running processes

Since the structures through which data is transmitted to start and stop the processes are accessed by several threads, it is necessary to use synchronization mechanisms, such as critical sections. At the same time, the code of the critical section should not cause long idle threads. The way to solve this problem is to use the FIFO queue as the data transfer structure. The work with the queue is shown in Fig. 7.

The method of placing objects in the queue requires synchronization, since this method can be called by several threads. The method of retrieving an object from the queue is called by a single thread, and therefore does not require synchronization. The time spent on placing objects in the queue is much less than the time required to start or end the process. Thus, the proposed approach reduces the waiting time for threads to release a critical section. In addition, for various programming languages, there are implementations of thread-safe queues, which simplifies the application of this approach [Mic19, Jav19].

3.4 Data Acquisition Module

The data collection module is a class containing data structures and methods designed to solve the following tasks:

- collection of information transmitted by user control modules;
- conversion of the received data into the form required for sending messages to the server module, as well as storing it in the database;
- transfer of the converted messages to the server connection modules and work with the database.

When instantiating a class corresponding to a data collection module, pointers to the objects of the server connection module and the database module should be passed to the constructor set.

The fig. 8 shows the message distribution pattern of the data collecting module. The method by which information is transmitted to the data collecting module accepts messages from monitoring modules and places them in the primary queue processed by the message distribution thread, which forwards the messages to two

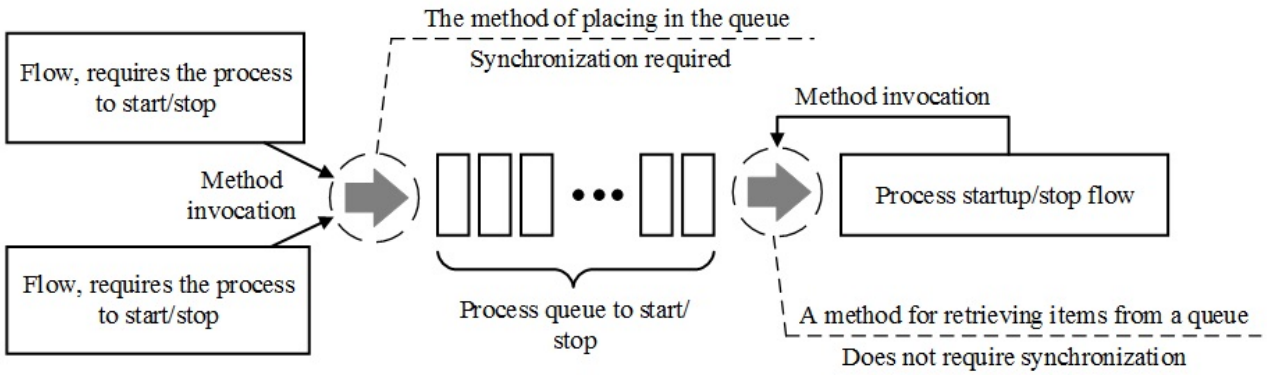

Figure 7: A scheme for using a queue to transfer data to processes start and stop threads

queues corresponding to the server connection and database modules. For processing each of the queues using a separate thread.

Before sending information to the server and storing the data in the database, the messages from the user action control modules, which are objects, must be converted to sequences of bits, i.e. it is necessary to produce their serialization.

Serialization allows to make a unified processing of messages from various control modules of user actions. To ensure that this mechanism can be applied, the serialization methods shown in Figure 2 should be implemented in each object class that stores messages. 8.

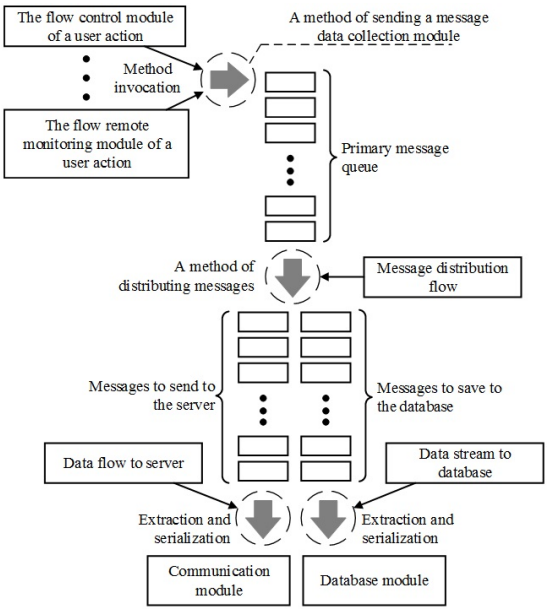

Figure 8: The message distribution scheme and the generalized structure of the classes of messages transmitted to the data collection module

Since the lists of information stored in the database and information transmitted to the server module can be defined independently, as well as data conversion parameters (for example, resolution and image compression ratio), the serialization process should be customizable. Required parameters can be passed to the serialization method as arguments. The result of the method is a bit sequence.

To create a universal code that can be used in the development of both client and server modules, you should implement structuring methods that allow to restore the structure of messages from the bit sequence.

Saving information about user actions in the local database is performed by the database operation module, which also contains a set of methods that allow the server module to retrieve the collected data of a certain type

for a specified period. This functionality is implemented through the component of interaction with the DBMS, which allows to perform the following actions: creating a database, connecting to a database, executing SQL queries for inserting, selecting and deleting data.

3.5 Server Module. Data Transmission Technology

The proposed scheme of interaction between the client and server parts of the software package for controlling user actions is shown in Fig. 9.

SM receives from M lists of running processes and screenshots. This information is sent by client modules at intervals determined by their settings. This approach allows the server to get a lot of desktop thumbnails without creating a heavy load on the network, and also to display the desktop of a single client computer in high quality with an increased frame rate. The task of the CM is also to send the CM commands for the completion of processes and remotely change the settings of the CM (for example, the list of prohibited processes) at the command of the administrator.

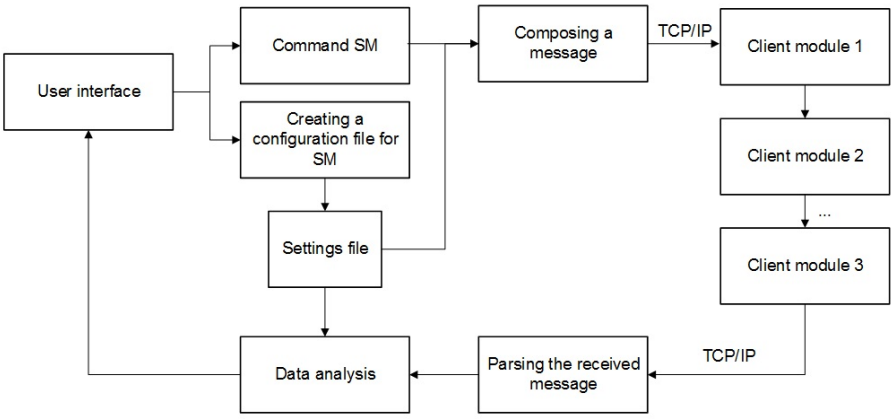

Figure 9: Generalized operation scheme of the SM

Interaction with other computers is provided through the use of sockets. The Microsoft Windows operating systems use Windows Sockets. This technology is compatible with the industry standard Berkeley Sockets. Using sockets allows the server to accept requests from UNIX clients or any other non-Windows systems. Thus, this technology allows you to create a cross-platform application in the future. In fig. 10 shows the complete algorithm of the socket technology. According to this algorithm, it becomes clear that there is no need for constant polling of a client or server about the presence of messages and it is quite enough to wait for a response message after sending a message.

Upon receipt of a new connection request, the SM checks the availability of a free child window for working with the client. If the number of connections exceeds the limit set in the settings, the connection will be interrupted.

After the connection, you can control the running processes of the client computer by sending commands from the side menu of the window in which the screen is broadcasted. Upon completion of the SM, all the "clients" of the EOF message (End Of File) are sent and the program is closed.

4 Results

In accordance with the architecture described in this article, a system of remote control of PC user actions is implemented. The implementation was performed in Embarcadero RAD Studio (C ++ Builder) in C ++. The interface of the developed server module is shown in fig. 11.

The main settings of the SM are the port number used for data transmission, as well as the maximum number of clients. Demonstration of the SM in the mode of viewing desktops is shown in Fig. 12.

The ability to view a list of processes running on a client PC (Fig. 13), as well as the completion of processes from this list, has been implemented.

To implement the corresponding functionality on the CM side (collecting screenshots, obtaining a list of processes, completing processes), the functions of the Windows application programming interface (WinAPI) are

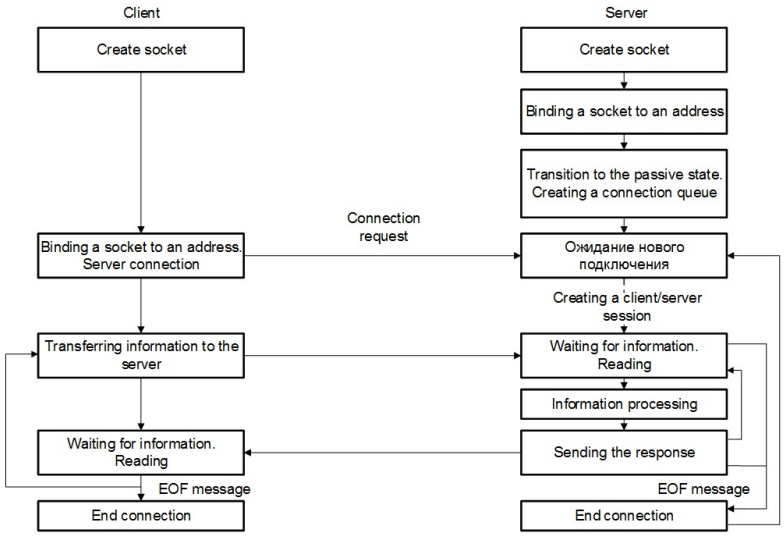

Figure 10: Socket technology operation algorithm

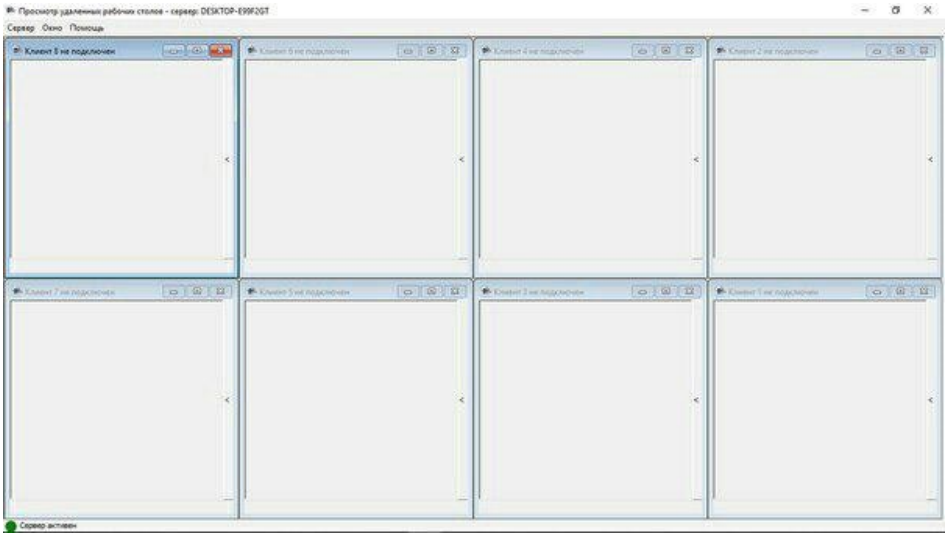

Figure 11: SM interface ready for operation

used. SQLite DBMS is used for a work with a local database.

The developed program modules are registered in the Computer Program Register [Cli18, Ser18].

5 Discussion

The functionality of the application can be expanded by implementing of the additional areas of monitoring, allowing to monitor the state of the PC hardware.

Hardware state monitoring involves collecting information about the PC hardware configuration (internal and external devices), information on the physical parameters (temperatures, voltages, power dissipation, fan speeds, and other available indicators), hardware resources usage (CPU and graphics accelerator load, used and free RAM space, disk subsystem utilization, network interface bandwidth usage, etc.), drive status (values of the S.M.A.R.T. attributes, remaining free space, etc.) [Ogr18].

The implementation of hardware resources control will simplify solving of such problems as malware detection (increased or atypical usage of hardware resources due to infection), prevention and detection of hardware malfunction (different from the nominal supply voltages, high operating temperatures of the components, low

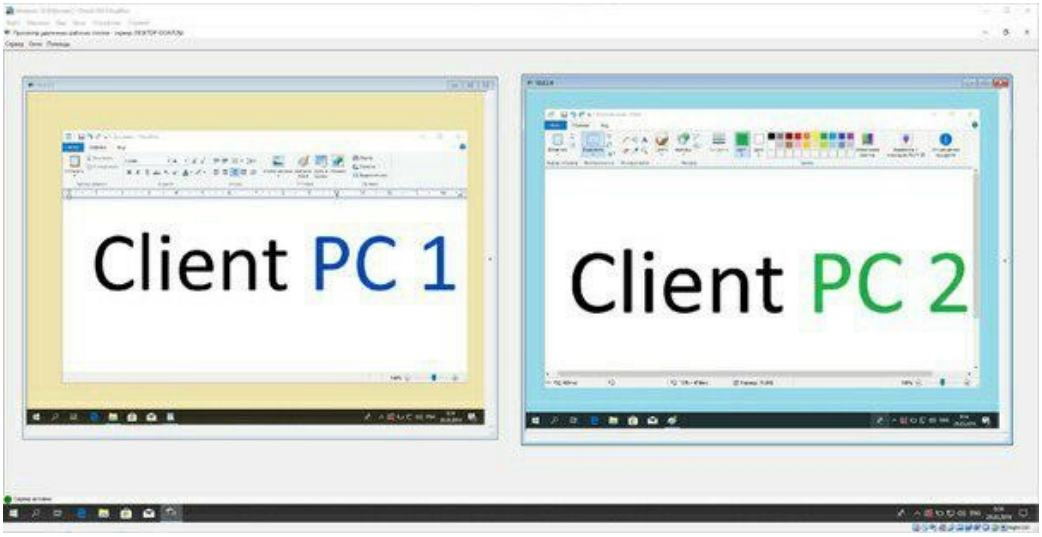

Figure 12: View snapshots of client desktops

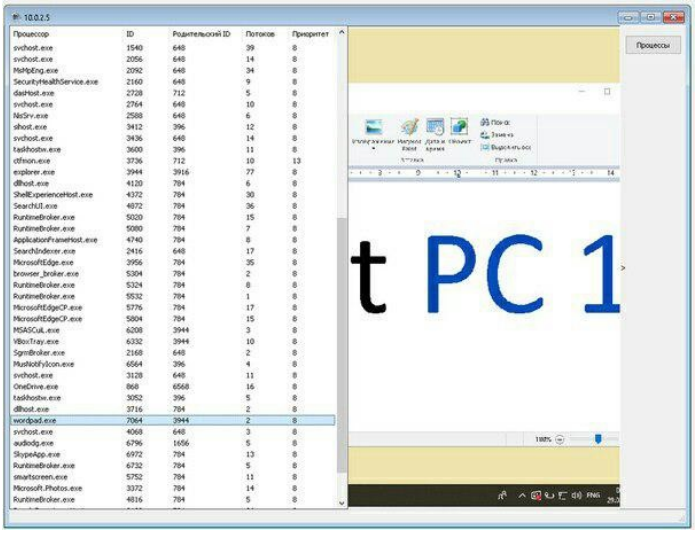

Figure 13: View processes running on client PC

or zero fan speed, deterioration of the attributes S.M.A.R.T.), more rational use of hardware resources (if most of the time hardware resources of a specific computer are idle or are not very involved, this machine can be loaded with additional tasks). Also it can help in making recommendations for upgrading the hardware. If usage of certain components of the computer (CPU, RAM, graphics accelerator, etc.) is close to maximum when performing work tasks, this may indicate the need to replace these components with more productive ones.

To ensure the information security of the transmitted data, it is also possible to implement traffic encryption between the server and client modules [Ogr16].

In view of the development of the technogenic society [Pel18], there are currently tendencies of expanding the areas of using digital information and communication technologies [Yus18]. In particular, employees of various organizations often use personal computers when dealing with confidential information. Therefore, the developed system can be used to monitor the actions of employees in order to improve the information security of the organization. To solve such problems, the functionality of the system can be expanded through the introduction of such areas of control as keylogging, printed documents, as well as visited network resources.

6 Conclusion

In this paper, an architecture of remote monitoring system of PC user actions is proposed, the algorithms and ways of interaction between individual program modules are considered.

The proposed architecture is modular, thus, the processes of software development, modification and maintenance are simplified. Parallel execution of the modules allows to use the benefits of multithreading capabilities of modern CPUs.

The collected data is saved to the local database of the controlled PC and may be used for deferred analysis. Data collection parameters can be adjusted in accordance with operating conditions.

Despite the use of C ++ Builder development environment for the implementation, the proposed architecture can be adopted for various operating systems (e.g. GNU/Linux).

References

[Ver18] Verma, A. S., and Mhetre, M. R. (2018). ECG Monitoring System Based on Internet of Things Technology // International Conference on Intelligent Data Communication Technologies and Internet of Things (ICICI) 2018, P.1241-1249.

[Kul19] Kulkarni, N and Lalitha, S.V.N.L. and Deokar, S.A. Real time control and monitoring of grid power systems using cloud computing // International Journal of Electrical and Computer Engineering 9(2). 2019. 9. P. 941-949.

[Dey19] R. Dey, S. Sen. Implementation of a Monitoring System using RF Technology. 13. 38-46. 10.21172/ijiet.131.05.

[Lit19] LiteManager – remote computer management and remote access. Remote Windows administration [Electronic resource]. – URL: http://litemanager.ru/ (accessed: 10.02.2019).

[Rad19] Radmin: reliable SOFTWARE for remote technical support. Selection of it professionals [Electronic resource]. – URL: https://www.radmin.ru/ (accessed: 10.02.2019).

[Kic19] System of control and accounting of working time of employees – Kickidler [Electronic resource]. – URL: https://www.kickidler.com/ru/ (accessed: 11.02.2019).

[Sta19] System of control of work (tasks) of employees (workers) Stakhanovite [Electronic resource]. – URL: https://stakhanovets.ru/ (accessed: 15.02.2019).

[Tri12] U. Trifonova, R. Zharinov. Concept of the system to protect children's access to information in schools using the DLP-system and RFID-technology // Conference of Open Innovation Association, FRUCT, Vol. 2012-November, pp. 142-146.

[Gar15] A. Garkusha. Building data in motion DLP system from scratch using opensource software and confirming its effectiveness within "capture the Flag" competitions // ACM International Conference Proceeding Series, Vol. 08-10-Sep-2015.

[Mor18] V. Morozov, N. Miloslavskaya. DLP systems as a modern information security control // Advances in Intelligent Systems and Computing, Vol. 636, pp. 296-301, 2018.

[Nem13] M. Nemirovsky, D.M. Tullsen. Multithreading architecture // Synthesis Lectures on Computer Architecture, Vol. 21, pp. 1-111, 2013.

[Sod10] A.C. Sodan, J. Machina, A. Deshmeh, K. Macnaughton, B. Esbaugh. Parallelism via multithreaded and multicore CPUs // Computer, Vol. 43(3), pp. 24-32, 2010.

[Mic19] ConcurrentQueue<T> Class / Microsoft Docs [Electronic resource]. – URL: https://docs.microsoft.com/en-us/dotnet/api/system.collections.concurrent.concurrentqueue-1?view=netframework-4.8 (accessed: 19.02.2019).

[Jav19] Thread-safe concurrent collections [Electronic resource]. – URL: http://java-online.ru/concurrent-collections.xhtml (date accessed: 19.02.2019).

[Cli18] The client software module, for remote control of desktops. UD. 2018666410 Russian Federation. The certificate of official registration of computer programs / D. L. Osipov, N. Yu. Svistunov, E. A. Ogrisko, A. S. Pavlov, applicant and owner Federal STATE Autonomous educational institution North Caucasus Federal University (RU). No 2018663924; Appl. 05.12.2018; publ. 17.12.2018, Register of computer programs. – 1 P.

[Ser18] Server software module for remote desktop monitoring. UD. 2018666572 Russian Federation. The certificate of official registration of computer programs / D. L. Osipov, E. A. Ogrisko, N. Yu. Svistunov, S. S. Ryabtsev, applicant and owner Federal STATE Autonomous educational institution North Caucasus Federal University (RU). No 2018663919; Appl. 05.12.2018; publ. 18.12.2018, Register of computer programs. – 1 P.

[Ogr18] Ogrisko, E. A., Ogur, M. G., Svistunov N. Yu., Osipov D. L. Enhancing the functionality of the software for monitoring user activity of a computer // Student science for development of the information society: collection of materials of VIII all-Russian scientific-technical conference. Part 2. – Stavropol: North Caucasus Federal University, 2018. – Pp. 165-168.

[Ogr16] Ogrisko E. A., Svistunov N. Yu. Osipov D. L. Principles of software development to control the actions of users of personal computers // Student science for the development of the information society: proceedings of the V all-Russian scientific and technical conference: in 2 parts. Part 2. Stavropol': KGU, 2016. – P. 335-337.

[Pel18] Pelevin, S. I., Taubaev, B. D., Baklanov, I. S. Problem of technogenic society dynamics under the conditions of contemporaneity (2018) International Journal of Civil Engineering and Technology, 9 (11), pp. 2437-2443.

[Yus18] Yusupova, N., Mironov, K. Key information technologies for digital economy // Proceedings of REMS 2018 Russian Federation and Europe Multidisciplinary Symposium on Computer Science and ICT. 2018. Vol-2254. P. 330-334.