System for monitoring parameters of functioning infrastructure objects and their external environment

I A Sidorov[1], R O Kostromin[1] and A G Feoktistov[1]

[1]Matrosov Institute for System Dynamics and Control Theory of SB RAS, Lermontov St. 134, Irkutsk, Russia, 664033

agf@icc.ru

Abstract. The paper addresses relevant issues applying the concept of Industry 4.0 in related to modeling infrastructure objects at the Baikal natural territory that use environmentally friendly technologies. In particular, the use of heat pumps belongs to such technologies, since this enables us to reduce air emissions. Object models are designed on the basis of their digital twins. Digital twins are intended to reflect the structure and processes of object functioning. In addition, we plan to delegate them the decision-making in managing these objects in real-time. Such a digital twin has to become smarter over time. This virtual entity has to gain knowledge and skills to select optimal scenarios for controlling object and improving its functioning parameters. Therefore, the initial problem in its development is creating a monitoring system for the collection, unification, aggregation, storage, and transmission of subject-oriented data. Such data include information about the object operation and environmental state. The data must be promptly obtained from peripheral equipment (controlling and measuring devices). For the effective operation of a digital twin, we have to partially transfer functions of primary data processing, their intellectual analysis and decision-making to the controlling and measuring devices. To this end, we use agents that implement software on peripheral equipment.

1. Introduction

Nowadays, one of the modern directions of modeling complex objects is the creation of digital twins [1]. They represent qualitatively new models of objects in the form of their digital profiles [2].

The paper addresses relevant issues of creating elements of such digital twins for the study of infrastructure objects of the Baikal natural territory. We consider a digital twin as the integration of adaptive analytical and simulation models, monitoring system of a real-time, and tools of the visualization and intellectual analysis of the retrospective and current data related to the object operation. The models have to realistically reflect the structure and functioning processes of an object. Monitoring, visualization, and intelligent analysis are used to support the automation of decision-making in the control and evolution of the object.

Today, the main applications of digital twins are energy, industry, transportation, finance, and urban infrastructure. At the same time, the study of infrastructure objects at nature protection territories and optimization of their work is a young scientific field that has not been fully advanced.

In this regard, designing models, methods, and tools for creating digital twins to describe the functioning processes of infrastructure objects at the Baikal natural territory is an extremely relevant problem. Especially when such objects use environmentally friendly technologies. We hope that the use of such digital twins will be in demand in studying, predicting, and optimizing technological, economic,

and environmental parameters of the operation for the studied objects. In addition, they will be applied in controlling these objects.

In the paper, we represent monitoring system tools designed to collect the current data about the functioning of the studied object and environment state. As an example, a scheme for obtaining such data in the Baikal Museum of the Irkutsk scientific center of the Siberian Branch of the Russian Academy of Sciences [3] is demonstrated.

The museum building is supplied with heat using two heat pumps NT-60 and NT-70. These pumps use the deep waters of Lake Baikal. Lowering the water temperature from about 4 to 2 ^{o}C is the main source for heating the museum. At low air temperatures in the Listvyanka village, where the museum is located, an additional electric boiler is used.

Applying heat pumps enables us to reduce air emissions. However, environmentally friendly technologies are often quite expensive. Therefore, it is important to ensure control over heat devices operation in order to optimize the cost of heat received.

The rest of this paper is organized as follows: Section 2 discusses the related work. System architecture for specifying and modeling infrastructure objects is considered in Section 3. Monitoring system database is represented in Section 4. In Section 5 and Section 6, we describe the used controlling and measuring devices, as well as software applied in this equipment. Section 7 shows equipment placement. Finally, conclusions are drawn in Section 8.

2. Related work

Nowadays, the development of industrial technologies and maintenance of technical systems lead to new challenges related to digitalization in manufacturing and servicing to achieve the higher-level quality [4]. The digital technologies known as Industry 4.0 technologies enable us to integrate and interconnect advanced intelligent methods and tools within such technologies [5]. They provide virtualization of the physical objects placing the cyber-physical elements within the framework of objects and uniting they by network infrastructure. Thus, this provides a remote sense, real-time monitoring, and intelligent control of the object equipment [6].

The evolution of information technologies, increase in the data storage size, and significant improvement in the computing systems performance create the conditions for processing big data and effectively eliciting knowledge [7, 8]. Together, this provides the necessary capabilities for the effective creation and use of digital twins.

Based on the [9], a digital twin is defined as an integrated multi-physics, multi-scale, probabilistic simulation of a complex product (or object). Wherein, the following characteristics of a digital twin are highlighted:

- Real-time representation of objects and their environments,
- Interaction and convergence for flows of the retrospective and current data in the physical and virtual spaces,
- Mapping the real world into the virtual world based on the set of the determined relations between their elements,
- Self-evolution of a digital twin.

Within applying to infrastructure objects of city, a digital twin can enable us to improve the human interaction with various infrastructures and their operation [10]. Examples of this use of digital twins are given in [11-13]. Nevertheless, the use of digital twins in modeling objects of natural protected territories is not sufficiently represented in the studies.

Each case of designing digital twins is individual. Therefore, the design of infrastructure for data processing in sensor networks is challenge [14]. In this regard, there is a need for data rotation.

The digital twin use involves the use of the fog and edge computing [15]. Software agents become an integral part of digital twins [16].

In this regards, we represent an original approach to selecting sensors, placing them at the infrastructure object (Baikal Museum), and collecting the current data. The represented approach is based on the above-considered trends in designing digital twins and adjusted to the object specifics.

In contrast to the approaches associated with the development and application of systems of similar purpose [17, 18], we transfer part of the computational load to peripheral equipment and perform it within fog computing under the control of software agents. This significantly reduces the time spent on data transferring.

3. Architecture

The object model design, preparation and conduct of large-scale experiments are carried out using the Orlando Tools framework [19]. It is designed to develop a special class of scalable scientific applications (distributed applied software packages). The system of specification and modeling of the functioning processes of infrastructure objects in the form of a digital double belongs to this class (Figure 1).

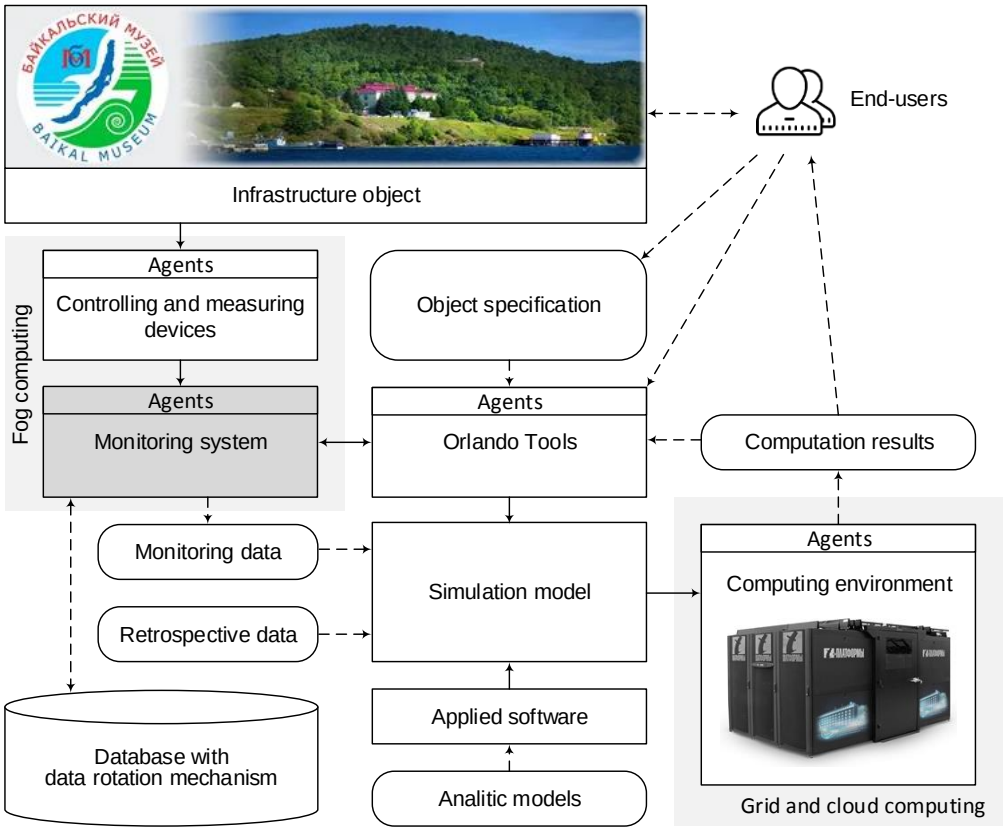


Figure 1. System architecture for specifying and modeling an object.

Designers construct a simulation model with Orlando Tools. They use an object specification, which describes object operation processes that can be interpreted and controlled.

In the process of model execution, current monitoring data and retrospective data about object operation processes and its environment are used. The current data is provided by the monitoring system, which receives them using peripheral equipment (controlling and measuring devices).

Agents represent the controlling and measuring devices. Together with monitoring system agents, they support fog computing. An agent design is considered in [20, 21].

Model instances are executed in parallel in an environment that integrates grid and cloud computing. Optimization analysis of the computation results is performed by Orlando Tools. The main resources of the environment are resources of the public access Irkutsk Supercomputer center [22].

The elementary functions of digital twins and agents are represented as micro-services. Such a way enables us to organize the interaction between the above virtual entities based on effective network

protocols. Additionally, it will simplify the processes of adding new and modifying existing twins and agents.

End-users of Orlando Tools of various categories will be provided with a web-based interface for access to the designed tools. In addition, they get necessary expert support in the process of preparing and conducting experiments. Such end-users are software developers, administrators of information and computation resources, decision-making experts, etc.

4. Database

The collection, unification, aggregation, storage, and transfer of data about the natural and climatic environmental conditions and object operation are carried out by a monitoring system [20]. It has been designed in Matrosov Institute for System Dynamics and Control Theory of SB RAS (ISDCT SB RAS) and adapted to servicing infrastructure objects. The database scheme of the monitoring system is shown in Figure 2. It includes the following main blocks of tables:

- Agent description,
- Characteristics of sensors and their relations with agents,
- Specification of measured parameter values.

Each table has a unique primary key id.

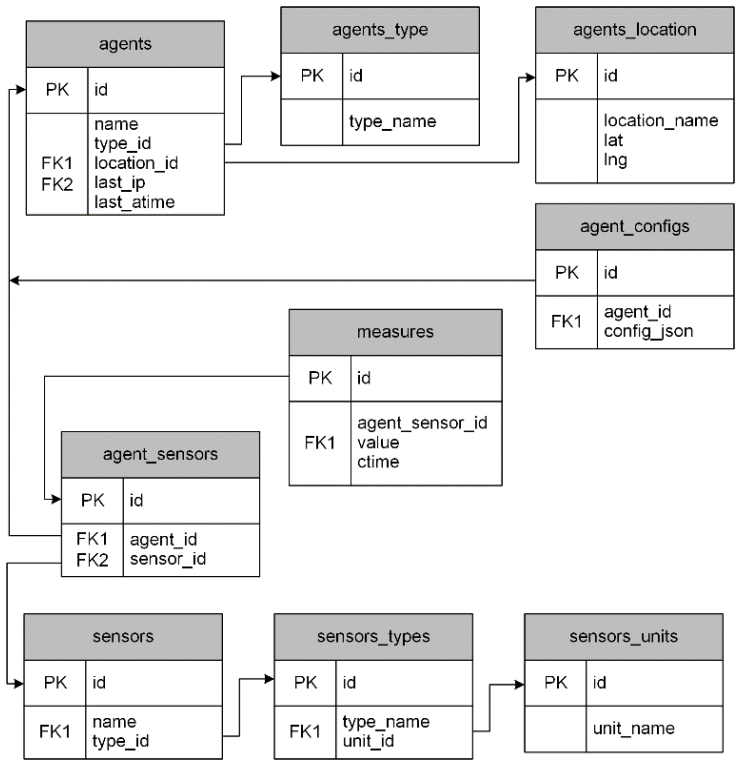


Figure 2. Database schema.

Each agent is described in the *agents* table and is determined by a unique name (the *name* field), type (the *type_id* field) and location (the *location_id* field). In addition, the agents table has two fields *last_ip* and *last_atime*, in which the last IP address of the agent and the time of last parameters sending in the Unix Time Stamp format are stored, respectively. The *agent_types* table contains a string field of the name of the agent type (for example, "raspberry pi", "node monitoring", "weather fetcher", etc.). The *agents_locations* table determines the agent locations (location name and geographic latitude and longitude).

Currently, there are the following agents:

- Agent for collecting climatic data from publicly available sources. It has the type “weather fetcher” and location Virtual Machine (VM) at IDSTU SB RAS.
- Agent for collecting operation parameters of the NT-70 pump. It has the type “raspberry pi” and location at the Baikal Museum.
- Agent for collecting operation parameters of the NT-60 pump. It has the type “raspberry pi” and location at the Baikal Museum.
- Agent for collecting climatic data in a building. It has the type “raspberry pi” and location at the Baikal Museum.

The tables describing the sensors have the following structures. The *sensors* table contains the fields with the sensor name (*name*) and its link to the sensor type (*type_id*). The *sensors_types* table contains the fields with the type name (*type_name*) and its link to the measurement unit of the parameter value from the sensor (*unit_id*). For example, there are the following types: “temperature sensor”, “pressure sensor”, “humidity sensor”, “current sensor”, “software sensor”, etc. The *sensors_units* table contains the field with the unit name *unit_name*. For example, the following units are available: “temperature (°C)”, “humidity (%)”, “pressure (at)”, “amperage (A)”.

Currently, we have installed 15 sensors in the Baikal Museum and Listvyanka village. The sensor locations were made in agreement with experts from the staff of the Baikal Museum.

The *agent_sensors* table is designed to set relations between sensors and agents. The table contains fields with two secondary keys *agent_id* and *sensor_id*. The *agents_configs* table contains additional agent configuration (for example, sensor querying intervals, data transfer intervals to the server, etc.) in the JSON format.

The *measures* table is designed to store values obtained using sensors. The table contains the secondary key *agent_sensor_id*, which links to the *agent_sensors* table. The use of this key enables us to determine the agent and sensor of each value. The table also includes the fields *value* for storing the values received from the sensor and *ctime* with the measurement time in the Unix Time Stamp format.

In order to prevent an excessive increase in the number of records in the measures table, we have implemented a data rotation mechanism. This mechanism provides support for the periodic overwriting of outdated data. The principle of data storage is as follows: we store all data for the last month. Averaged data for every 10 minutes is stored from one month to three months. Averaged data for every 30 minutes is stored for three to six months. Averaged data for each hour is stored from six months to one year. Averaged data for every 3 hours is stored from one year to three years. Subsequently, we store daily average data.

The proposed data rotation mechanism showed higher performance (on average about 15%) in data processing in comparison with the universal tool RRDtools [23]. Performance gains are achieved through optimizing data structures and additional data caching in our system.

5. Equipment

The process of obtaining and processing parameters of the heat pump operation includes the following main stages:

- Equipping infrastructure objects with sensors,
- Receiving primary weakly structured data from sensors using a controller,
- Transformation of primary data to a structured form,
- Saving structured data to the local device database,
- Transfer of structured data to the monitoring system database.

Nowadays, the use of single-board mini-computers is a popular way of collecting data from sensors. Among them, the Raspberry Pi, Arduino platforms, and their numerous analogues stand out. In our study, the Raspberry Pi was selected (Figure 3) [24]. This mini-computer supports the installation of the Raspbian version of the Debian operating system [25]. This version is adapted for ARM processors. It supports the operation of software (Python, Shell (BASH), Java) for processing data from sensors and tools for connecting to Virtual Private Network (VPN). Thus, we chose the Raspberry Pi 3 Model B+ single-board mini-computer. Its characteristics are presented in Table 2.

Figure 3. Raspberry Pi 3 Model B+.

Table 1. Characteristics of Raspberry Pi 3 Model B+

Model	Microarchitecture	Frequency	Number of cores	RAM	GPIO	Number of USB ports	Ethernet
Raspberry Pi 3 Model B+	Cortex-A53 (ARM v8)	1.4 GHz	4	1 GB	40 pins	4	Gigabit

A common advantage of this class of mini-computers (controllers) is the presence of a low-level interface General-Purpose Input/Output (GPIO). GPIO provides the ability to programmatically control devices through the appropriate assignment of contacts (ports) of this interface. In addition, GPIO has 3 and 5 V outputs, grounding contacts, and system contacts for connecting GPIO expansion cards. The assignment of GPIO ports on the Raspberry Pi 3 Model B+ is shown in Figure 4. One of the GPIO outputs (GPIO4, no. 7) provides support for the 1-Wire protocol.

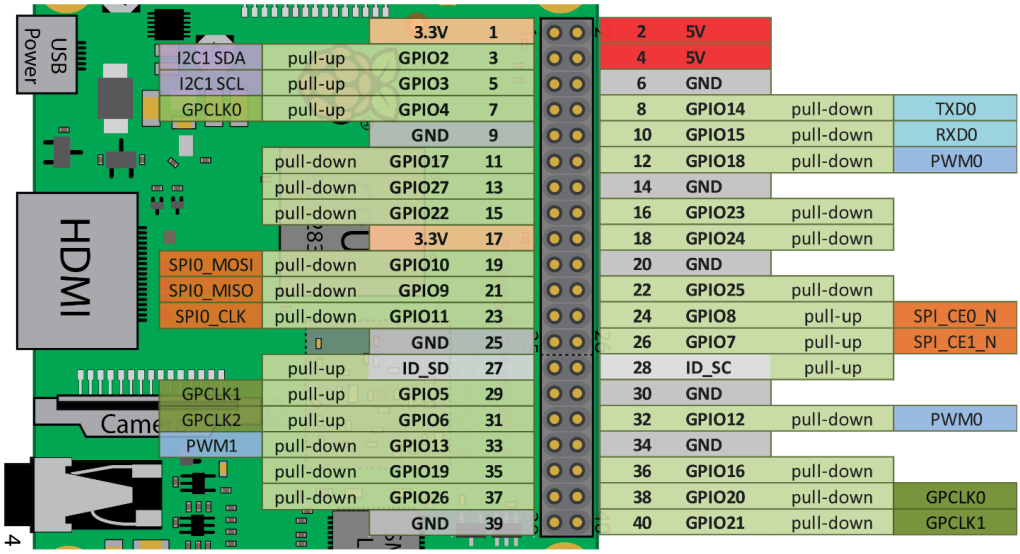


Figure 4. GPIO ports of Raspberry Pi 3 Model B+.

This means that several sensors operated in parallel can be connected via a single controller port with a common wire. Since each sensor has a unique 64-bit identifier, the number of devices connected to the bus can be quite large. It is permissible to connect sensors of various types. This allows us to use sensors with 1-Wire support in environmental control systems, temperature monitoring in buildings and equipment nodes. We have been selected waterproof temperature sensors (Figure 5) DS18B20 manufactured by Dallas Semiconductors. The GND output is connected to the grounding output of the GPIO. The Vdd output is connected to an output with a voltage of 3 or 5 V. The Data output is intended for data exchange. An important property of the selected sensors is that they can be connected by a common 1-Wire data bus. The sensor bus connection diagram with the corresponding GPIO4 outputs of the Raspberry Pi controller is shown in Figure 6.

Figure 5. Waterproof temperature sensor DS18B20.

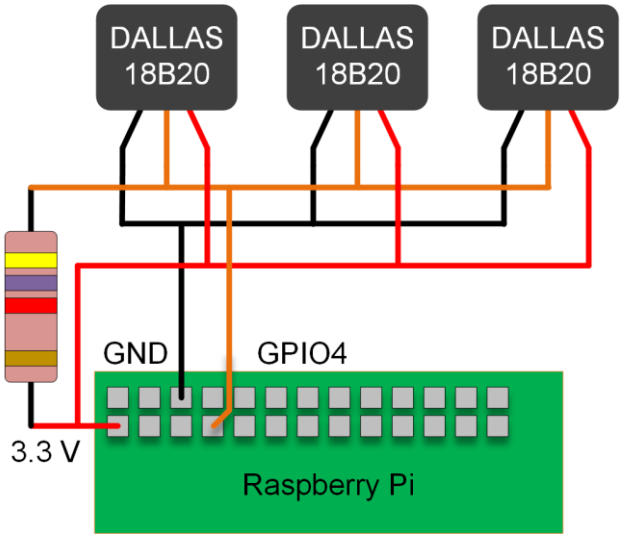


Figure 6. Scheme of connecting temperature sensors to Raspberry Pi.

The sensor DS18B20 has the following characteristics:

- Support for data exchange with the microcontroller via a single-wire communication line,
- Connection of several sensors through one common communication line,
- Assignment of a unique 64-bit serial code for each sensor,
- Operating voltage from 3.0 to 5.5 V,
- Current consumption 9 mA,
- Temperature measurement range from -55 to 125 °C,
- The measurement error does not exceed 0.5 °C in the range from -10 to 85 °C,
- The measurement time does not exceed 750 ms with a maximum accuracy of temperature,
- Transfer of temperature value from memory,
- Sending a checksum together with the temperature value in order to ensure the reliability and integrity of the transmitted data,
- Data transmission over twisted pair up to 300 m.

6. Software

To activate the 1-Wire protocol support in Raspbian, it is required to download the corresponding kernel modules: w1-gpio, which activates the protocol of the 1-wire module on GPIO4, and w1-therm, which loads the temperature reading module from the 1-wire bus. The loading commands for these modules are given below:

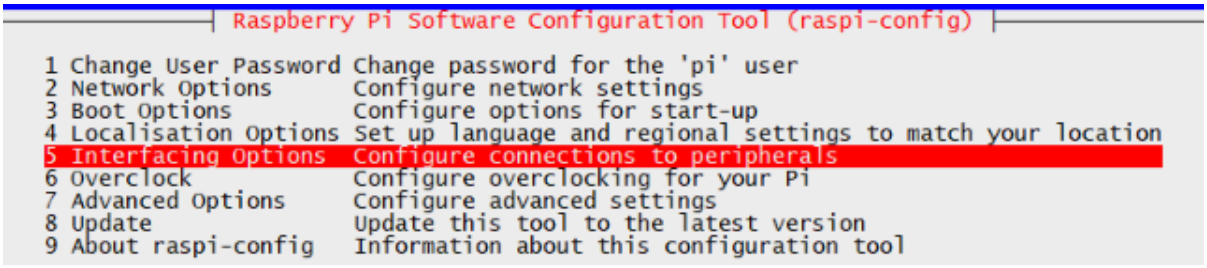
```
sudo modprobe w1-gpio
sudo modprobe w1-therm
```

To ensure automatic loading of these modules after turning off the Raspberry Pi, they must be placed in the directory /etc/modules:

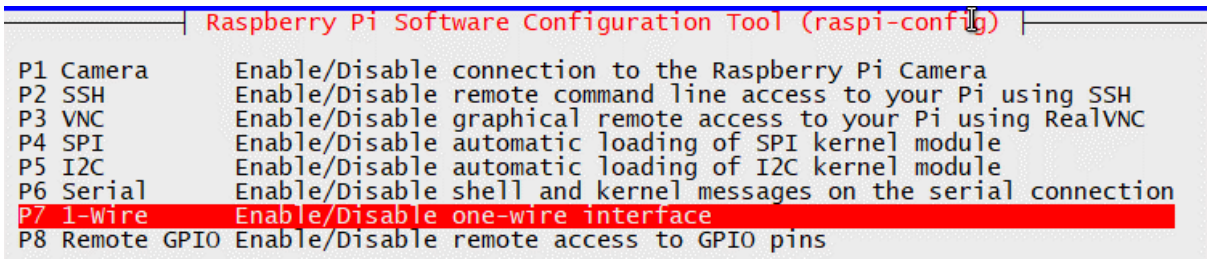
```
sudo echo "w1-gpio" >> /etc/modules
sudo echo "w1-therm" >> /etc/modules
```

In addition, we have to enable 1-Wire support using the setup program Raspberry Pi raspi-config (Figures 7a and 7b). After that, all directories corresponding to the identifiers (in the form of 28-xxxxxxxxxxxx) of the connected sensors will be accessible from the device directory /sys/bus/w1/ (Figure 8). In addition to sensors, the directory of the controller w1_bus_master1 is located there. To

obtain the temperature value from the sensor, it is necessary to read the contents of the file w1_slave from the corresponding directory of this sensor (Figure 9).



a)



b)

Figure 7. Program raspi-config use (a) to support 1 Wire (b).

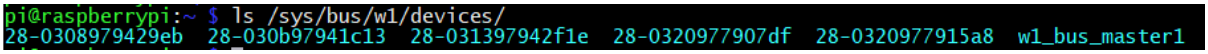


Figure 8. Catalog with available sensors.

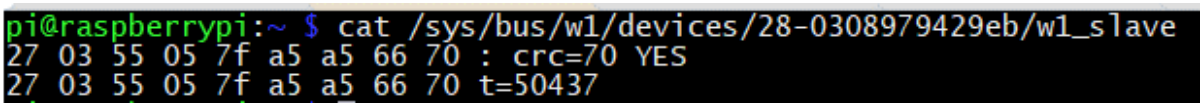


Figure 9. Contents of the file w1_slave.

The hexadecimal numbers in the file w1_slave reflect the byte information that the sensor returned to the controller. The correctness of the CRC checksum is verified on the first line (the last byte of the response must match the CRC). Upon successful verification, the answer "YES" is displayed, otherwise the value "NO" is returned. If the value is "NO", it is necessary to repeat reading the data from the sensor. The second line contains the read temperature in °C multiplied by 1000. Thus, we can implement the temperature reading from all files of different sensors in any programming language.

The OpenVPN client is installed on the Raspberry Pi, which provides a connection to VPN of the monitoring systems. VPN enables us to connect to the Raspberry Pi via SSH, regardless of the type of Internet connection used in the absence of an external static IP address. This eliminates the need to configure the router.

We have developed a system for collecting, storing, analyzing, and transferring data. Within its framework, a software agent is implemented in the programming language node.js. to read temperature readings on the Raspberry Pi and transfer them to a remote monitoring system server. The software agent starts when Raspberry Pi boots up and runs in the background. It has a modular structure and includes the following components:

1) Module for data reading,
2) Module for storing data in a local database,
3) Module for data transferring in the database of the monitoring system,
4) Module for local data analysis and notification message formation,

5) Module for agent configuring.

By default, Module 1 requests the sensors once every one minute. As a result of a survey of sensors, a data structure is formed that includes the following elements:

- Generated unique record identifier in UUID format,
- Sensor ID,
- Time of receiving data from the sensor in the Unix Time Stamp format,
- Sensor reading.

The formed data structure is transferred to Modules 2-4. Module 2 caches the collected values in the device memory and periodically writes them to disk (once every 30 minutes). Data caching is implemented in order to reduce the number of calls to the device's flash memory to extend its life. An embedded lightweight SQLite relational database is used as a local database [26]. Once a day, the agent applies the data rotation mechanism proposed in Section 4. Module 3 sends the received values to a remote server of the monitoring system using web-services. Module 4 verifies the sensor workability and the correctness of the value read from it. Module 5 is called when the agent starts and then every 24 hours. The configuration is transferred on request through the web-service of the monitoring system. The transferred parameter is the UUID of the Raspberry Pi device. In response, a document in JSON format with a description of the sensors and their limit values, intervals for periodically starting the agent modules, and notification parameters for operators are sent.

Figure 10 shows a screenshot of the web-portal of the temperature monitoring and visualization system. The following information is available to authorized end-users:

- Visualization of temperature values from available sensors for the last five hours,
- Tabular representation of temperature values for selected sensors,
- Photos of the heat pumps with designations of installed sensors.

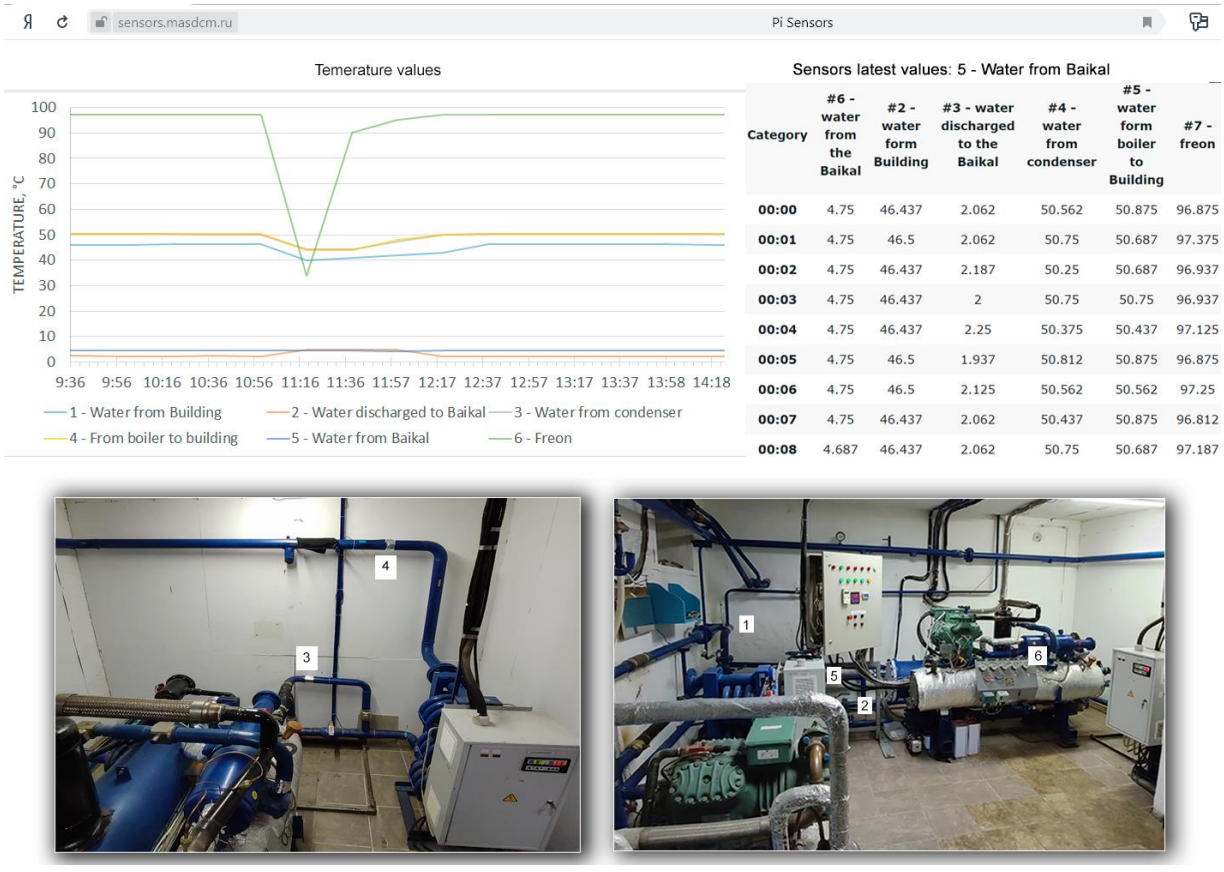


Figure 10. Example of visualization of data collected from heat pumps of the Baikal Museum.

Graphs and tables are updated in real-time in accordance with information from the database. The operator of the monitoring system can select any time interval for visualization.

Viewing sensor readings for a certain period of time is implemented in the form of temperature graphs (Figure 11).

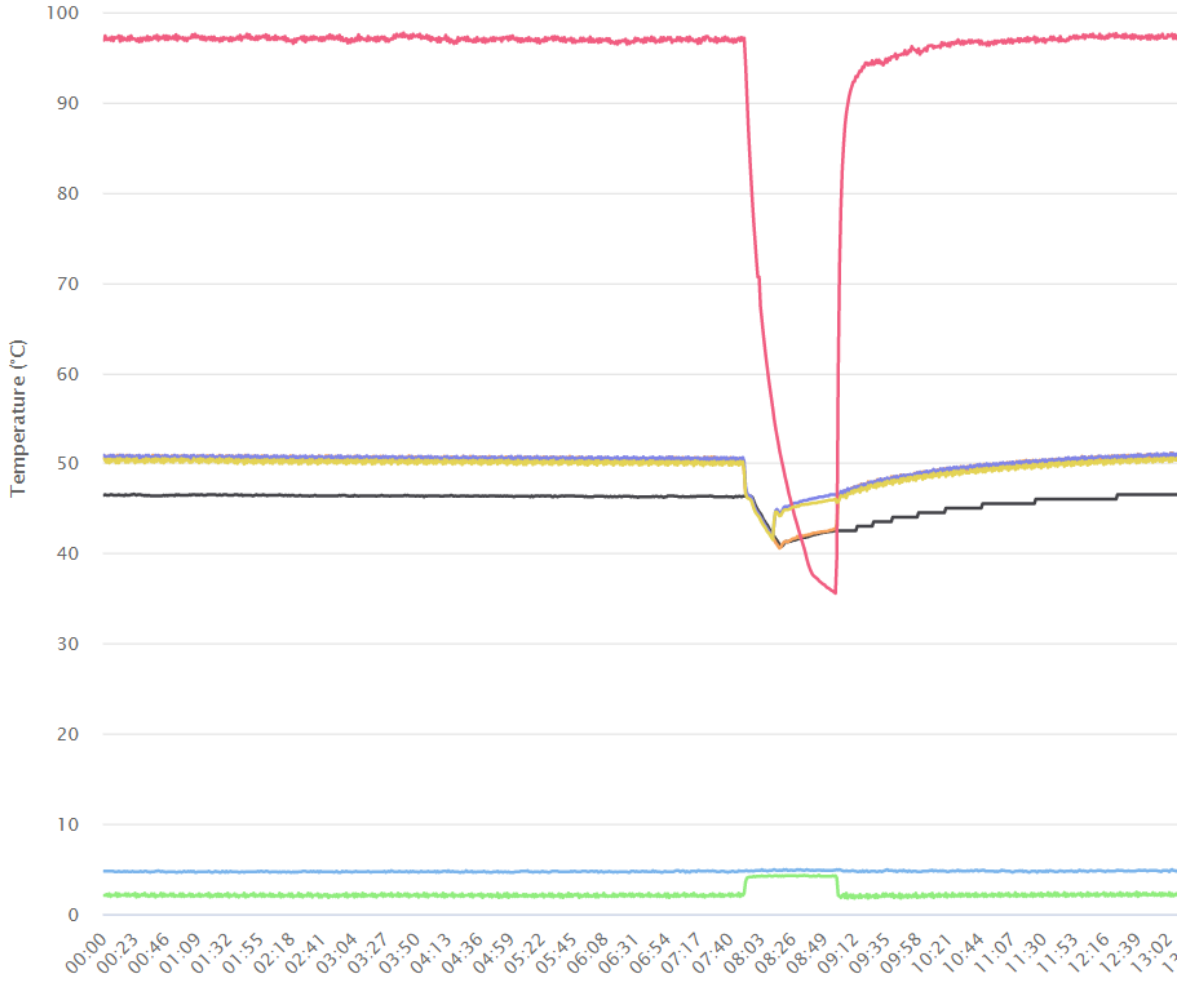


Figure 11. Parameters of temperature sensors of the heat pump NT-70.

7. Sensor mounting

Sensors are fixed with heat-resistant metallic scotch-tape on pipes of thermal devices (Figure 12). Each sensor is marked. The correctness of the sensor readings was verified by measuring the corresponding temperatures with a pyrometer.

The output contacts of the sensors are connected to the common bus by soldering. A twisted-pair cabling is used as the bus. From the Raspberry Pi controller, there is one cable connected by three pins to GPIO4, 5 V outputs, and to grounding, respectively. The buses from all sensors are connected in parallel to the common bus.

The Raspberry Pi controller is placed in the heat pump control unit (Figure 13). A 220 V power supply and twisted-pair cabling from one of the switches of the Baikal Museum are connected to it. The Raspberry Pi controller automatically receives the IP address from the Dynamic Host Configuration Protocol server after it is turned on. Then he independently connects to VPN.

To obtain temperature values from sensors, a special script is run by the cron scheduler [27] on a schedule (once every 10 minutes). The received data is placed in the device directory /sys/bus/w1/devices.

Figure 12. Sensor mounting.

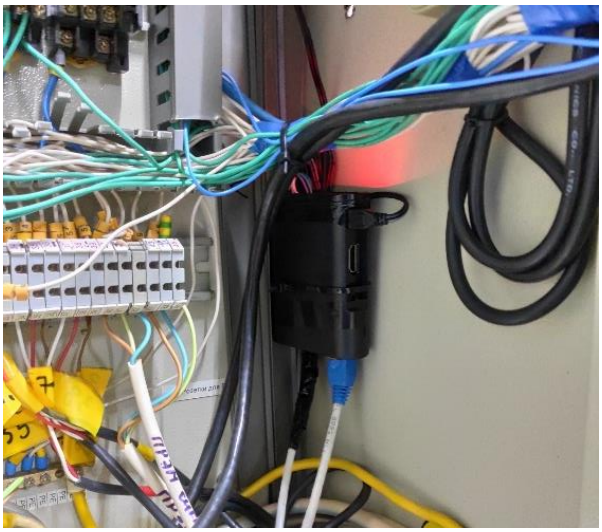


Figure 13. Controller mounting.

8. Discussion

The implementation of the system for monitoring the temperature parameters of an object is characterized by a specific set of conditions:

- Restricted access to basic equipment owing to the functioning mode of the technical premises,
- Insufficient access to sensors because of the lack of a dedicated server and ability to manage router settings,
- Unstable operation of the Internet,
- Inability to access the local IP addresses of computing devices from outside owing to applying the Network Address Translation mechanism.

Under such conditions, the use of the traditional tools of monitoring computing devices (for example, Zabbix and Nagios, involving direct access to them via the SSH or HTTP protocols) is impossible.

In this regard, we proposed a specialized approach to providing access to the control device (Raspberry Pi) through OpenVPN. Within this approach, ways to connect to the Internet and configure the router are not important. When the Internet access is available, direct access to the Raspberry Pi via SSH is always provided. In addition, we implemented the collection, initial processing, and recording of data in a local database with the subsequent synchronization of this database with a remote server. This ensures data safety even with a temporary lack of access to the Internet.

Moreover, we have developed a specialized web-portal for visualizing relevant data from a synchronized database. The web-portal is located on the server with a dedicated IP-address. Therefore, access to the portal is always available.

Thus, the novelty of the proposed implementation of the monitoring system lies in the development, specialization, and integration of the three independent subsystems designed for the following purposes:

- Providing access to peripheral equipment,
- Collecting, processing, and storing of data,
- Data visualization.

Agents located on peripheral equipment to collect climatic data and operation parameters of the pumps perform primary data processing within the framework of fog computing. They operate within the created VPN, and not on the cloud server. Such an organization of agents operation is due to a decrease in the delay in data transfer and an improvement in the interconnection with end-devices. Thus, the overheads of data transferring and processing in the cloud are reduced.

In the future, the functions of agents will be expanded by the intellectual analysis of current data and decision-making on equipment control.

9. Conclusions

In the paper, we consider relevant issues related to designing digital twins for studying infrastructure objects at the Baikal natural territory. In particular, the tools for collecting the current data about the functioning of the studied objects and environment state are represented.

We proposed and implemented the original scheme for obtaining data about the heat pump functioning at the Baikal Museum. At the controlling and measuring devices, we use software agents. They support fog computing in the data processing. Moreover, we proposed the new data rotation mechanism that shows higher performance in data processing in comparison with the known universal tools for a similar purpose.

The development and inclusion of new analytical models of heat pumps in the simulation system are the closest directions of our further research. In addition, the effective distribution of data processing between grid, cloud, and fog computing resources will be studied.

Acknowledgments

The study is supported by the Russian Foundation of Basic Research and Government of Irkutsk Region, project no. 20-47-380002-p_a. We sincerely thank the director, scientific leader, management, and employees of the Baikal Museum for advising and expert supporting in our study.

References

[1] Uhlemann T H-J, Lehmann C and Steinhilper R 2017 The Digital Twin: Realizing the Cyber-Physical Production System for Industry 4.0 *Procedia CIRP* **61** 335–340

[2] Kritzinger W, Karner M, Traar G, Henjes J and Sihn W 2018 Digital Twin in manufacturing: A categorical literature review and classification *IFAC-PapersOnLine* **51(11)** 1016–1022

[3] Baikal Museum. Available at: http://bm.isc.irk.ru/ (accessed: 10.05.2020).

[4] Uhlemann T H-J, Schock C, Lehmann C, Freiberger S and Steinhilper R 2017 The Digital Twin. Demonstrating the Potential of Real Time Data Acquisition in Production Systems *Procedia Manuf.* **9** 113–120

[5] Negri E, Fumagalli L and Macchi M 2017 A Review of the Roles of Digital Twin in CPS-based Production Systems. *Procedia Manuf.* **11** 939–948

[6] Lee J, Bagheri B and Kao H-an 2015 A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems *Manuf. Lett.* **3** 18–23

[7] Iafrate F 2014 A Journey from Big Data to Smart Data Proceedings of the Second International Conference on Digital Enterprise Design and Management *Adv. Intel. Syst. Comput.* **261** 25–33

[8] Didenko N I, Skripnuk D F and Mirolyubova O V 2017 Big data and the global economy *Proc. of the 2017 Tenth Intern. Conf. on Management of Large-Scale System Development (ML SD)* (IEEE) pp 1–5

[9] Tao Fei, Cheng J, Qi Q, Zhang M, Zhang He and Sui F 2018 Digital twin-driven product design, manufacturing and service with big data *Int. J. Adv. Manuf. Technol.* **94(9-12)** 3563–3576

[10] Mohammadi N and Taylor J E 2017 Smart city digital twins *Proc. of the 2017 IEEE Symposium Series on Computational Intelligence (SSCI)* (IEEE) pp 1–5

[11] Harrison C, Eckman B, Hamilton R, Hartswick P, Kalagnanam J, Paraszczak J and Williams P 2010 Foundations for Smarter Cities *IBM J. Res. Dev.* **54(4)** 1–16

[12] Schlapfer M, Bettencourt L M, Grauwin S, Raschke M, Claxton R, Smoreda Z, West G B and Ratti C 2014 The scaling of human interactions with city size *J. R. Soc. Interface* **11(98)** 20130789

[13] Mohammadi N, Taylor J E and Wang Y 2017 Towards smarter cities: linking human mobility and energy use fluctuations across building types *Proc. of the 50th Hawaii Intern. Conf. on System Sciences (HICSS 50)* (IEEE) pp 2824–2833

[14] Aberer K, Hauswirth M and Salehi A 2007 Infrastructure for data processing in large-scale interconnected sensor networks *Proc. of the Intern. Conf. on Mobile Data Management*,

(IEEE) pp 198–205
[15] Qi Q, Zhao D, Liao T W and Tao F 2018 Modeling of cyber-physical systems and digital twin based on edge computing, fog computing and cloud computing towards smart manufacturing. *Proc. of the 13th Intern. Manufacturing Science and Engineering Conf.* (*ASME*). Available at: https://doi.org/10.1115/MSEC2018-6435 (accessed: 10.05.2020).
[16] Datta S P A 2016 Emergence of digital twins. *arXiv preprint arXiv:1610.06467*. Available at: https://arxiv.org/abs/1610.06467 (accessed: 10.05.2020).
[17] Tugarinov P, Truckenmüller F and Nold B 2019 Digital twin of distributed energy devices *Proceedings of the International Scientific And Technical Conference: Forum of Mining Engineers* (NTU Dnipro Polytechnic Press) pp 323–331
[18] Vering C, Mehrfeld P, Nürenberg M, Coakley D, Lauster M and Müller D 2020 Unlocking Potentials of Building Energy Systems' Operational Efficiency: Application of Digital Twin Design for HVAC systems *Proceedings of the 16th IBPSA Intern. Conf. and Exibition* (IBPSA) pp 1304–1310
[19] Tchernykh A, Feoktistov A, Gorsky S, Sidorov I, Kostromin R, Bychkov I, Basharina O, Alexandrov A and Rivera-Rodriguez R 2019 Orlando Tools: Development, Training, and Use of Scalable Applications in Heterogeneous Distributed Computing Environments *Comm. Com. Inf. Sc.* **979** 265–279
[20] Bychkov I V, Oparin G A, Novopashin A P and Sidorov I A 2015 Agent-Based Approach to Monitoring and Control of Distributed Computing Environment *Lecture Notes Comp. Sci.* 9251 253–257
[21] Feoktistov A, Kostromin R and Tchernykh A 2018 Agent Behavior Model for Distributed Computing Management in the Environment with Virtualized Resources *Proc. of the 41st Intern. Convention on information and communication technology, electronics and microelectronics (MIPRO-2018)* (Riejka: IEEE) pp 1153–1158
[22] Irkutsk Supercomputer Center SB RAS. Available at: http://hpc.icc.ru/ (accessed: 10.05.2020).
[23] RRDtool – The Time Series Database. Available at: https://oss.oetiker.ch/rrdtool/ (accessed: 10.05.2020).
[24] Raspberry Pi. Available at: https://www.raspberrypi.org/ (accessed: 10.05.2020).
[25] Raspbian OS. Available at: https://www.raspbian.org/ (accessed: 10.05.2020).
[26] SQLite. Available at: https://www.sqlite.org/ (accessed: 10.05.2020).
[27] CRON. Available at: https://www.gnu.org/software/mcron/ (accessed: 10.05.2020).