

JOINT INSTITUTE FOR NUCLEAR RESEARCH

Manuscript copyright

Priakhina Daria Igorevna

**Digital twins for solving management and development tasks of
distributed data acquisition, storage and processing centers**

Scientific specialty 2.3.1. System analysis, management and information processing,
statistics

Thesis for a Candidate Degree in Technical Sciences

Translation from Russian

Supervisor:
Doctor of Technical Sciences
Korenkov Vladimir Vasilyevich

Dubna

2024

Contents

Introduction	4
Chapter 1. Analysis of the task of creating digital twins of distributed data acquisition, storage and processing centers	15
1.1. Distributed data acquisition, storage and processing centers.....	15
1.2. State of the problem of modeling distributed data acquisition, storage and processing centers	16
1.3. Digital twin: definition, application examples.....	18
1.4. Possibilities of using digital twins of data storage and processing centers.....	20
1.5. Conclusions to Chapter 1	21
Chapter 2. Models, methods and algorithms for creating digital twins of distributed data acquisition, storage and processing centers	23
2.1. Preliminary remarks.....	23
2.2. Requirements for digital twins of distributed data acquisition, storage and processing centers	24
2.3. Method of creating digital twins of distributed data acquisition, storage and processing centers	27
2.4. Algorithms for implementing of the method of creating digital twins of distributed data acquisition, storage and processing centers	30
2.5. Data model	35
2.6. Conclusions to Chapter 2	40
Chapter 3. Implementation of algorithms and development of special software for creating digital twins and interacting with them	41
3.1. Implementation tools	41
3.2. Implementation of the digital twin kernel.....	42
3.3. Implementation of a module for user interaction with a digital twin.....	51
3.4. Development of special software.....	55
3.5. Conclusions to Chapter 3	57
Chapter 4. Verification and experimental operation of the special software for creating digital twins	58

4.1.	Verification of the digital twin kernel.....	58
4.2.	Application of the special software to create the digital twin of the computing infrastructure of the BM@N experiment of the NICA complex	70
4.3.	Application of the special software to create the digital twin of the computing system of the online data filter of the SPD experiment of the NICA complex.....	82
4.4.	Conclusions to Chapter 4	86
	Conclusion.....	88
	Glossary of terms	90
	References	91
Appendix 1.	Description of data model entities	98
Appendix 2.	User instructions for working with the special software	103
Appendix 3.	Certificate of registration of a computer program	112
Appendix 4.	Letters on the application of the results of the PhD thesis research	114
Appendix 5.	Act of implementing the results of the PhD thesis research in the educational process	118

Introduction

Relevance of the research topic

Data centers (DCs) are widely used in various fields: in science for storing and further processing data from experiments; in business for providing IT outsourcing services; in government organizations for automating internal processes, for example, document management; in security systems of the urban environment, transport and industrial enterprises to ensure collection, storage and analysis of access control systems, video surveillance, etc. There are many computing systems of various scales for storing and intensive data processing. In this research, distributed systems (centers) designed to collect, store and process extremely large amounts of data (up to 10^{18} bytes) are considered as the most relevant and applicable for computing infrastructures of large-scale scientific experiments.

Distributed centers are a complex that includes not only powerful computing and data storage resources, but also network equipment responsible for data exchange within the system and communication with external consumers, engineering systems, security systems, monitoring systems, etc. Such complex systems require high-quality design in order to determine as accurately as possible the parameters of all processes that will occur in the system under consideration, to choose the necessary equipment for the system operation, taking into account current needs and future development prospects. In addition, during the operation of distributed centers there is a need for scaling in order to increase the efficiency of equipment use, to speed up data processing, etc. Scaling requirements may vary depending on the field of application and the types of tasks to be solved. Therefore, a tool that will allow analyzing the effectiveness and reliability of possible scenarios for the development of distributed data acquisition, storage and processing centers (DDCs) is needed. At the same time, it is necessary to take into account all processes occurring in the DDC, including job flow management strategies, as well as data flow parameters for storage and processing.

Such a tool is usually a variety of modeling software packages. The main element of such packages is the core, i.e. a computer program that performs the modeling of systems. Some packages use databases (DBs) to store input data and simulation results. Existing modeling tools do not enable to carry out the above-mentioned studies qualitatively for the following reasons:

- modern trends in the construction of distributed centers, including heterogeneous data processing modules, cloud structures and supercomputers, hierarchical memory systems and much more, are not taken into account;
- such an important criterion for the functioning of the system as data loss, depending on the type of equipment, the probability of failures, changes in the performance of computing resources and data storages, is not considered;

- a detailed description of the parameters of computer and network systems in the DB leads to difficulties in making a decision on the choice of a system configuration;
- there is no search for a hardware configuration with specified criteria;
- for each simulated infrastructure, it is necessary to adapt the modeling kernel by changing the program code;
- there is no interface for user interaction with the simulation program.

Therefore, in the process of creating and improving the DDC, there is a problem of solving such important tasks as:

- checking various DDC scaling scenarios, taking into account the requirements for data flows and jobs;
- formation and enhancement of a job flow management strategy for the efficient allocation of DDC resources in data processing;
- analysis of the resources used and assessment of the required amount of resources for specific tasks according to the requirements for the DDC.

Currently, technologies for the elaboration of problem-oriented management and decision-making systems based on digital twins (DTs), which can be applied to solving the above tasks, are actively developing. Thus, it seems relevant to develop a method and implement algorithms for creating digital twins of DDCs to solve the tasks of designing, managing and developing DDCs, including to verify the effectiveness and reliability of their functioning. Using the DT at the design stage of the DDC as a prototype will enable to analyze the operating modes of future systems and increase their reliability. Consequently, the developed methods and algorithms can be applied to a wide class of tasks for the construction and development of DDCs within large scientific experiments and large-scale projects.

The main requirements for DTs are the adequacy of modeling (compliance of the DDC model with a real system according to a list of characteristics), the visibility of initial data and results presentation, as well as the deliverance of a user from participation in the development and maintenance of the DT core, i.e., a computer program that implements algorithms for modeling various DDC processes

Degree of study of the problem

The first mention of digital twins was made in 2002 [1]. Over the past 20 years, this technology has found application worldwide in almost all fields of science and human activity [2]. Due to the progressive trend of creating data centers for various tasks of science and business [3, 4, 5], DT technologies began to be used in this area.

For example, the American companies Future Facilities [6] и Sunbird DCIM [7] are engaged in the software development and creation of DC digital twins. However, firstly, these DTs are virtual copies of DCs, which are geographically located in the same physical space, and secondly, DCs are considered

only from the point of view of engineering infrastructure. At the moment, neither in Russia nor abroad, there is a tool that allows one to create a DDC digital twin, taking into account data flows and job flows, which can be used to evaluate and select the necessary amount of resources for specific tasks according to the requirements for the DC.

The effectiveness of the creation, use and development of DT technology is determined by the quality of the constructed model, therefore, modeling approaches play a decisive role in the construction of DTs [8]. To improve the quality of the DT model, it is necessary to take into account the probabilistic characteristics of processes occurring in the simulated system. The application and use of DTs are provided by various software tools and information technologies. The DT needs to be upgraded during operation based on the results of testing. Thus, it is required to provide criteria for evaluating the quality of the DT based on the results of its work.

It should be noted that it is important to take into account the functionality of the DT, which will allow one to select the configuration of the DDC equipment. It is necessary to use not only the technical characteristics of the equipment as selection criteria, but also others, for example, time indicators. As a result, several potentially competitive options can be selected from a variety of acceptable configurations.

Thus, the relevance of the chosen research topic follows from the above.

Goal and tasks of the research

The goal of the research is to develop a method and algorithms for creating digital twins to describe distributed systems, to make decisions on choosing equipment configurations within the task of scaling and developing distributed data acquisition, storage and processing centers, of managing resources and processes occurring in the DDC.

To achieve this goal, it is necessary to solve the following tasks.

1. To conduct an analysis of the subject area.
2. To develop a method for creating DDC digital twins, which, unlike existing ones, performs the modeling of distributed centers taking into account the characteristics of job flows and data for storage and processing, as well as the probabilities of changes in processes occurring in the DDC.
3. To implement algorithms, the DB structure, and a web user interface for creating and executing a DT, as well as providing graphical information about the results of the DT work. Based on the created models, methods and algorithms, to develop special software that allows one to compare the DDC efficiency depending on different hardware configurations.
4. To carry out the verification and experimental operation of the special software.

Scientific innovation

New scientific results obtained personally by the author are as follows.

1. A new method of creating and using DDC digital twins is proposed and developed. The method differs from existing ones in the ability to simulate such processes as data storage and processing, taking into account the characteristics of data flows and jobs, the probabilities of failures and changes in the equipment performance and other processes occurring in the simulated system.

2. Algorithms for describing the infrastructure of a distributed system and forming its virtual image are developed and implemented.

3. A problem-oriented decision-making system for management and optimization tasks is developed in order to improve the technical characteristics of the DDC based on DT models. Its adequacy is confirmed by the example of the computing infrastructure of the BM@N¹ experiment of the NICA² complex.

4. The configuration of the DDC for the BM@N and SPD³ experiments of the NICA complex is scientifically justified.

Theoretical significance

The theoretical significance of the research lies in the development of a method for modeling distributed computing systems that operate with extremely large amounts of data requiring reliable storage and a complex processing system.

The applicability of the methodology developed in the thesis for the creation of DDC digital twins is proved on the basis of the results of verification of the computational infrastructure model of an existing experiment.

Practical significance

The practical significance lies in the application of the results obtained to enhance the efficiency, quality and reliability of complex data acquisition, storage and processing systems.

The developed special software can be used for a wide class of tasks in the field of design, construction and development of DDCs, including helping to select several potentially competitive options from a variety of acceptable hardware configurations.

The certificate of state registration of the computer program No. 2023667305 “Software Complex for Creating Digital Twins of Distributed Data Acquisition, Storage and Processing Centers” dated 14 August 2023 is received.

The problem of finding an equipment configuration for the data acquisition, storage and processing system of the BM@N experiment of the NICA complex at the Joint Institute for Nuclear Research (JINR) is solved, which is confirmed by the corresponding letter of application.

¹ *Baryonic Matter at Nuclotron* is an experimental facility for the study of baryonic matter.

² *Nuclotron based Ion Collider Facility* is an accelerator complex designed at the Joint Institute for Nuclear Research (Dubna, Russia) to study the properties of dense baryonic matter.

³ *Spin Physics Detector* is an experiment conducted on a spin physics detector.

The results of the research are used in the design of the computing system for the online data filter of the SPD experiment of the NICA complex at JINR, which is confirmed by the corresponding letter of application.

The results of the research are used in the educational process of the Federal State Budgetary Educational Institution of Higher Education “Dubna University” in the course “Distributed Computing and Cloud Technologies” for the preparation of master’s students in the field of 27.04.03 System Analysis and Management in the profile “Digital Platforms and Big Data Analytics”, which is confirmed by the corresponding act of implementation.

Research methodology and methods

The theoretical research is based on the implementation of the principles of a systematic approach and the methods of system analysis in the study of system connections and patterns of the functioning of complex systems, which are modern data acquisition, storage and processing centers, including distributed ones.

The practical results are obtained on the basis of the use of modern architectural solutions and tools for developing software, web applications and DB.

Degree of reliability

The reliability of the thesis results is ensured by the correct application of the principles of a systematic approach and the methods of system analysis in the study of system connections and patterns of the functioning of complex systems.

The proposed solutions are based on the study and critical understanding of scientific works and developments in organizing the distributed storage and reliable transfer of large amounts of data from physical experiments.

The reliability of the recommendations and conclusions based on the results of the thesis research is confirmed by the practice of applying the developed methods in the design and development of computing infrastructures for large-scale experiments in the field of high-energy physics.

Approbation of the results

The methods and special software developed in this work are tested and show their effectiveness in creating a DT for the distributed data acquisition, storage and processing center of the BM@N experiment of the NICA complex, which is confirmed by the corresponding letter of application.

The results were discussed at the following annual meetings of the BM@N experiment collaboration:

1. 5th Collaboration Meeting of the BM@N Experiment at the NICA Facility, 20-21 April 2020 (JINR, Dubna, Russia).
2. 6th Collaboration Meeting of the BM@N Experiment at the NICA Facility, 26-27 October 2020 (JINR, Dubna, Russia).

3. 8th Collaboration Meeting of the BM@N Experiment at the NICA Facility, 03-08 October 2021 (JINR, Dubna, Russia).

4. 9th Collaboration Meeting of the BM@N Experiment at the NICA Facility, 13-16 September 2022 (JINR, Dubna, Russia).

5. 10th Collaboration Meeting of the BM@N Experiment at the NICA Facility, 14-19 May 2023 (SPbU, St. Petersburg, Russia).

6. 11th Collaboration Meeting of the BM@N Experiment at the NICA Facility, 28-30 November 2023 (JINR, Dubna, Russia).

The special software developed in this work is currently being used to design the computing system of the SPD experiment of the NICA complex, in particular for the online data filter, which is confirmed by the corresponding letter of application. The results obtained and plans for further cooperation were discussed at the SPD experiment collaboration meeting: VI SPD Collaboration Meeting and Workshop on Information Technology in Natural Sciences, 23-27 October 2023 (Samara University, Samara, Russia).

The main provisions and results of the work were reported, discussed and approved by specialists at the following Russian and international scientific events:

1. 8th International Conference “Distributed Computing and Grid-technologies in Science and Education” (GRID 2018), 10-14 September 2018 (JINR, Dubna, Russia).

2. All-Russian scientific and practical conference “Nature, Society, Man” of Dubna University, 21-23 November 2018 (Dubna University, Dubna, Russia).

3. International Conference “Mathematical Modeling and Computational Physics” (MMCP2019), 1-4 July 2019 (Stará Lesná, Slovakia).

4. XXVII International Conference “Mathematics. Computer. Education”, symposium with international participation “Biophysics of complex systems: computational and systems biology, molecular modeling”, 27 January – 1 February 2020 (Dubna University, Dubna, Russia).

5. XXIV International Scientific Conference of Young Scientists and Specialists (AYSS-2020), 09-13 November 2020 (JINR, Dubna, Russia). A diploma for the best report of the conference was awarded.

6. 9th International Conference “Distributed Computing and Grid-technologies in Science and Education” (GRID 2021), 05-09 July 2021 (JINR, Dubna, Russia).

7. 56th meeting of the PAC for Particle Physics, 24-25 January 2022 (JINR, Dubna, Russia).

8. XXVI International Scientific Conference of Young Scientists and Specialists (AYSS-2022), 24-28 October 2022 (JINR, Dubna, Russia).

9. 57th meeting of the PAC for Particle Physics, 23 January 2023 (JINR, Dubna, Russia).

10. XIII Conference (with international participation) “Information and Telecommunication Technologies and Mathematical Modeling of High-Tech Systems” (ITTMM 2023), 17-21 April 2023 (RUDN, Moscow, Russia).

11. 10th International Conference “Distributed Computing and Grid-technologies in Science and Education” (GRID 2023), 03-07 July 2023 (JINR, Dubna, Russia).

12. XXVII International Scientific Conference of Young Scientists and Specialists (AYSS-2023); 29 October – 03 November 2023 (JINR, Dubna, Russia). A diploma for the best report of the conference was awarded.

13. 59th meeting of the PAC for Particle Physics, 22 January 2024 (JINR, Dubna, Russia).

In 2017, 2019 and 2022, the research was supported by the Meshcheryakov Scholarship at the JINR Meshcheryakov Laboratory of Information Technologies (MLIT) for work in the field of support for experimental and theoretical physics, and in 2018 and 2020, it was supported by the Govorun Scholarship at MLIT for work in the field of information, computer and network support for JINR activities. In 2021 and 2023, grants for young JINR researchers to develop methods and a complex of programs for modeling data storage and processing centers were received. Winner of the II degree of the JINR Prize for Young Scientists and Specialists for 2023 in the nomination “Scientific and Technical Applied Works”.

Publications

The main scientific results of the thesis are published in the works listed below.

1. Trofimov V.V., Nechaevskiy A.V., Ososkov G.A., Priakhina D.I. Probability-cost approach to optimizing distributed data storage systems for physical experiments // CEUR Workshop Proceedings. 2018. Vol. 226. pp. 393–399 (in Russian).

2. Korenkov V.V., Priakhina D.I., Nechaevskiy A.V., Ososkov G.A., Trofimov V.V. Simulation of data storage and processing centers taking into account economic components // System analysis in science and education. 2018. No. 4. pp. 1–8 (in Russian).

3. Korenkov V., Nechaevskiy A., Ososkov G., Priakhina D., Trofimov V. A probabilistic approach of the simulation of data processing centers // European Physical Journal Web of Conferences. January 2020. Vol. 226. P. 03012.

4. Priakhina D., Trofimov V., Ososkov G., Gertsenberger K. Data center simulation for the BM@N experiment of the NICA project // AIP Conference Proceeding. 2021. Vol. 2377. P. 040007.

5. Priakhina D., Korenkov V., Gertsenberger K., Trofimov V. Simulation of Data Processing for the BM@N Experiment of the NICA Complex // CEUR Workshop Proceedings. 2021. Vol. 3041. pp. 483–487.

6. Priakhina D., Korenkov V., Trofimov V., Gertsenberger K. Simulation Results of BM@N Computing Infrastructure // Physics of Particles and Nuclei Letters. 2023. Vol. 20. No. 5. pp. 1272–

1275.

7. Priakhina D.I., Korenkov V.V. Relevance of creating a digital twin for managing distributed data acquisition, storage and processing centers // *Modern Information Technologies and IT-Education*. 2023. Vol. 19. No. 2. pp. 262–271 (in Russian).

8. Priakhina D.I., Korenkov V.V., Trofimov V.V. Method for creating digital twins to solve the tasks of effective management and development of distributed data acquisition, storage and processing centers // *Modern Information Technologies and IT-Education*. 2023. Vol. 19. No. 2. pp. 272–281 (in Russian).

9. Priakhina D.I., Korenkov V.V., Trofimov V.V., Gertsenberger K.V. Verification of the simulation program for creating digital twins of distributed data acquisition, storage and processing centers // *International Journal of Open Information Technologies*. January 2024. Vol. 12. No. 1. pp. 118–128 (in Russian).

The certificate of state registration of the computer program No. 2023667305 “Software Complex for Creating Digital Twins of Distributed Data Acquisition, Storage and Processing Centers” dated 14 August 2023 is received.

Structure and scope of the thesis

The PhD thesis consists of an introduction, four chapters, a conclusion, references and five appendices.

The introduction reflects the relevance of the work, sets the goal of the research, as well as the tasks for achieving it, substantiates the scientific innovation of the work, formulates the provisions to be defended, and shows the practical significance of the results obtained.

The first chapter analyzes the task of creating digital twins of distributed data storage and processing centers. For this purpose, examples of distributed centers are considered, and the problems of DDC modeling are described. In addition, an overview of the concept of DT technology is provided, application examples are presented, and the possibilities of using DTs of data centers are outlined.

In the second chapter, a new method for constructing DDC digital twins is proposed. A formal description of the method, functional and non-functional requirements for the DT, including a computer program that performs the modeling of systems, and a module for user interaction with the DT are presented. According to the described requirements, algorithms to implement the method of constructing DDC digital twins are developed. Due to the fact that the functioning of the algorithms is impossible without a DB, a logical data model with a detailed description of entities, their attributes and relationships is provided.

The third chapter is devoted to the software implementation of algorithms for creating DDC digital twins. Diagrams that demonstrate the general structure of the hierarchy of the developed classes of computer programs, as well as the life cycle of certain objects, are presented. The architecture of the special

software is shown. It includes all the developed algorithms and modules for the implementation of the method of constructing DDC digital twins. Implementation tools used both to develop individual algorithms and to combine components into a single complex are described. The characteristics of the equipment necessary for the operation of the special software are given.

The fourth chapter discusses the results of the verification of the DT core, which confirm the adequacy of the developed models and prove the possibility of their further use. The results of the application of the developed special software using examples of creating DDC digital twins to solve design tasks, improve the efficiency, quality and reliability of complex systems for acquiring, storing and processing data from existing experiments in the field of high-energy physics are presented.

In conclusion, the results of the research are summarized, the main conclusions and possible prospects for the development of this work are formulated.

The total volume of the thesis is 119 pages, including 94 figures, 2 tables and 14 formulas. The list of references includes 82 titles.

Main scientific results

1. The system connections and patterns of the functioning of complex systems, which are DDCs, are investigated using the principles of a systematic approach and the methods of system analysis. Methods for describing distributed systems, making decisions on the choice of equipment configurations, and managing resources and processes of complex systems are developed. Models, methods and algorithms for creating DDC digital twins are developed. Algorithms, the DB structure and a web user interface are implemented for creating and executing a DT, as well as providing graphical information about the results of its work. The results are presented in the following publications: **Priakhina D.I.**, Korenkov V.V. Relevance of creating a digital twin for managing distributed data acquisition, storage and processing centers // *Modern Information Technologies and IT-Education*. 2023. Vol. 19. No. 2. pp. 262–271 (in Russian on pages 267–268); **Priakhina D.I.**, Korenkov V.V., Trofimov V.V. Method for creating digital twins to solve the tasks of effective management and development of distributed data acquisition, storage and processing centers // *Modern Information Technologies and IT-Education*. 2023. Vol. 19. No. 2. pp. 272–281 (in Russian on pages 275–278); Trofimov V.V., Nechaevskiy A.V., Ososkov G.A., **Priakhina D.I.** Probability-cost approach to optimizing distributed data storage systems for physical experiments // *CEUR Workshop Proceedings*. 2018. Vol. 226. pp. 393–399 (in Russian on pages 394–395); Korenkov V.V., **Priakhina D.I.**, Nechaevskiy A.V., Ososkov G.A., Trofimov V.V. Simulation of data storage and processing centers taking into account economic components // *System analysis in science and education*. 2018. No. 4. pp. 1–8 (in Russian on page 7); Korenkov V., Nechaevskiy A., Ososkov G., **Priakhina D.**, Trofimov V. A probabilistic approach of the simulation of data processing centers // *European Physical Journal Web of Conferences*. January 2020. Vol. 226. P. 03012 (on page 03012-2).

2. Special software is developed on the basis of the created models, methods and algorithms. The software allows comparing the efficiency of the DDC operation depending on different hardware configurations. The certificate of state registration of the computer program No. 2023667305 “Software Complex for Creating Digital Twins of Distributed Data Acquisition, Storage and Processing Centers” dated 14 August 2023 is received.

3. The verification of the DT kernel is performed using the example of the computing infrastructure of the BM@N experiment of the NICA accelerator complex. The adequacy is assessed by several indicators by comparing the DT results with statistical data from monitoring the computational infrastructure of the experiment. The result is presented in the following publication: **Priakhina D.I.**, Korenkov V.V., Trofimov V.V., Gertsenberger K.V. Verification of the simulation program for creating digital twins of distributed data acquisition, storage and processing centers // International Journal of Open Information Technologies. January 2024. Vol. 12. No. 1. pp. 118–128 (in Russian on page 126).

4. Recommendations on the results of the DT functioning are taken into account in the design and development of computing infrastructures for large-scale experiments in the field of high-energy physics. The experimental operation of the special software is performed during the creation of the DT of the computing infrastructure of the BM@N experiment. The most suitable configuration of data processing equipment in the shortest possible time is obtained. A strategy for managing job flows and distributing the load on computing resources is chosen. The experimental operation of the special software is performed during the creation of the DT of the online data filter system of the SPD experiment of the NICA complex. The results enable to evaluate the required parameters of the equipment for storing, processing and transferring online filter data, taking into account the planned characteristics of experimental data flows. The results are presented in the following publications: **Priakhina D.**, Trofimov V., Ososkov G., Gertsenberger K. Data center simulation for the BM@N experiment of the NICA project // AIP Conference Proceeding. 2021. Vol. 2377. P. 040007 (on pages 040007-3–040007-5); **Priakhina D.**, Korenkov V., Gertsenberger K., Trofimov V. Simulation of Data Processing for the BM@N Experiment of the NICA Complex // CEUR Workshop Proceedings. 2021. Vol. 3041. pp. 483–487 (on pages 485–486); **Priakhina D.**, Korenkov V., Trofimov V., Gertsenberger K. Simulation Results of BM@N Computing Infrastructure // Physics of Particles and Nuclei Letters. 2023. Vol. 20. No. 5. pp. 1272–1275 (on pages 1273–1274).

Provisions to be defended

1. A method for constructing a digital twin is developed. The method allows describing distributed data acquisition, storage and processing centers, taking into account data flows and job flows, as well as processes occurring in the DDC. The ability to simulate such processes as data storage and processing, taking into account the characteristics of data flows and jobs, the probabilities of failures and changes in the equipment performance and other processes occurring in the simulated system are distinctive features

of the new method. The developed approach to the creating of a digital twin realizes methods for describing distributed systems, making decisions on the choice of equipment configurations, and managing resources and processes of complex systems.

2. Algorithms, on the basis of which special software used to make decisions on choosing the configuration of equipment for distributed data acquisition, storage and processing centers according to specified requirements is created, are developed. The special software allows comparing the efficiency of the DDC operation depending on different hardware configurations. Algorithms, the database structure and a web user interface are implemented for creating and executing a digital twin, as well as providing graphical information about the results of its work. Modern architectural solutions and tools for developing software, web applications and databases are used.

3. The adequacy of the constructed methods and algorithms is confirmed using the example of the computational infrastructure of the existing experiment. The verification is performed using the example of the computing infrastructure of the BM@N experiment of the NICA accelerator complex. The adequacy is assessed by several indicators. The resulting values prove that the deviations of the digital twin results from the results of real DDC allow to use the constructed methods and algorithms for solving management and development tasks of distributed systems. Results of the experimental operation of the special software during the creation of the digital twins of the computing infrastructure of the BM@N experiment and the online data filter system of the SPD experiment of the NICA complex results enable to evaluate the required parameters of the equipment for storing, processing and transferring data, taking into account the planned characteristics of data flows. Recommendations on the results of the digital twins functioning are taken into account in the design and development of computing infrastructures for large-scale experiments in the field of high-energy physics. The results of the software application prove the efficiency and high quality of the models and algorithms developed in the thesis.

Chapter 1. Analysis of the task of creating digital twins of distributed data acquisition, storage and processing centers

1.1. Distributed data acquisition, storage and processing centers

The development of scientific research in high-energy physics, astrophysics, biology, earth sciences, chemistry, as well as in medicine, nanotechnology, industry, business and other areas of human activity requires the joint work of many organizations to process large amounts of data in a relatively short time. It is for this purpose DDCs are created and developed. DDCs are capable of transmitting and storing huge data arrays, as well as serving as a universal efficient infrastructure for high-performance distributed computing and data processing.

Distributed systems are designed, as a rule, for acquisition, storage and processing of ultra-large amounts of data and represent a geographically distributed infrastructure, combining many resources of different types (processors, long-term and RAM, storage, networks), which can be accessed by the user from any point, regardless of their location. Such a system assumes a collective shared mode of access to resources [9]. This means that distributed data acquisition, storage and processing centers include not only powerful hybrid computing and storage resources, but also network equipment responsible for data exchange within the system and communication between centers and with external consumers, engineering systems, security systems, monitoring systems and others.

An example of an DDC is the infrastructure for processing data from the Large Hadron Collider (LHC), a high-energy physics experiment at CERN (the European Organization for Nuclear Research, Geneva, Switzerland) [10]. This system has several levels of organization (Tier). The essence of the distributed model is that the entire volume of data from the LHC detectors after real-time processing and primary reconstruction should be sent for further processing and analysis to regional centers. It should be noted that the experimental data are also distributed over several data repositories, which have different geographical locations. In Russia, in particular, there are Tiers of this large distributed infrastructure [11].

Systems similar to the LHC computational infrastructure should be created and developed in Russia for existing and planned experiments of the “mega sciences” class. Such Russian experiments may include:

- NICA project consisting of several experiments at the accelerator complex to study properties of dense baryonic matter (Dubna) [12];
- research at the PIK reactor (Gatchina) [13];

- Center for Collective Use “Siberian Ring Photon Source” (Novosibirsk) [14];
- research in the field of high-energy neutrino astrophysics using the Baikal-GVD neutrino telescope (Lake Baikal) [15].

It is also worth paying attention to the experiments realized jointly with other countries, e.g., JUNO (China) — a planned precision experiment with new-generation reactor ante-neutrinos designed to determine the neutrino mass hierarchy [16].

DDCs for such important and complex experiments need to evolve and scale to ensure the quality and efficiency of scientists who strive for rapid scientific results and new discoveries. Depending on the application and the types of problems to be solved, the scaling requirements may be different, so a tool is needed to analyze the efficiency and reliability of the various processes occurring in DDCs, taking into account the data flows for storage and processing, and to improve the strategy for managing job flows. Traditional modeling tools do not allow such researches to be performed qualitatively. Further materials of this chapter are presented in accordance with the published article [17], where the advantages and disadvantages of existing DDC modeling tools are discussed in detail and recommendations for the development of an alternative solution are proposed.

1.2. State of the problem of modeling distributed data acquisition, storage and processing centers

At the stage of designing the DDC, it is important to determine as accurately as possible the parameters of all processes that will occur in the system. For this purpose, various models are created. In the process of modeling it is possible to determine the minimum necessary equipment for the functioning of the system, as well as to choose several variants of equipment taking into account the current needs and future development prospects. The ongoing research considers libraries and program complexes that allow, first of all, to carry out modeling of DDC data flows.

So far, various tools for modeling distributed systems have been created, such as Bricks, OptorSim, and GridSim [18], which had a narrow specialization. With the help of these software packages it was possible to model certain distributed system architectures. It is worth noting that the use of such tools requires knowledge of special programming languages, which significantly reduces the efficiency of their use [19]. But nevertheless, there are software packages for modeling built on the basis of the above tools.

For example, the SyMSim software tool is developed using the GridSim package, which is built on the SimJava library that allows modeling the flow of discrete events in time [20]. It should be noted that the GridSim (the latest version 5.2 was released in 2010 [21]) and SimJava (the latest version 2.0

was released in 2002 [22]) libraries are currently outdated, so they cannot be used for modeling modern RDCs, as they do not allow taking into account new trends in the construction of distributed systems that have, for example, a hybrid structure, i.e., include heterogeneous data processing modules, including cloud structures and supercomputers, as well as hierarchical memory systems and much more.

Nevertheless, SyMSim provides an opportunity to simulate the job flow process of a distributed system with specified resources and rules of their reservation and use. SyMSim is a simulation model that allows to obtain the value of job processing time and to estimate how the structure of the computing system and the performance of its individual parts affect this time. Based on the results of the simulation, we can conclude that the highest value of job (data) flow intensity [23].

The main feature of the SyMSim program is the synthesis of simulation and monitoring processes. Monitoring data of a particular computer center are stored in the database and then used to form the input jobs flow for simulate. If there is no monitoring data for a particular computer center, the statistics of jobs for similar computer centers is analyzed [24]. Then the distributions of jobs by execution time and input file size are created, after which a hypothesis about the number of job types in the flow is hypothesized and tested [25]. Generated data flows and jobs flows regardless of their volume are stored in the DB, which is a disadvantage because it can lead to serious computational costs when working with the DB.

The means of describing the computing infrastructure in the program complex is realized as a DB with an interface presented as a web page [26]. The description is assigned an identifier, which the user must specify in the parameters of the model run. The model receives information from the DB, which is used to build a description of the computational structure [27]. Consequently, the equipment parameters and architecture of the modeled infrastructure are set through the DB, which is a serious disadvantage. Not all potential users of the program complex have skills of working with the DB. Direct interaction of inexperienced users with the DB can lead to violation of its integrity.

The kernel of the program is a module that is responsible for receiving processing jobs and “launching” them on processors, as well as managing queues by storing information about currently occupied and free processors. In operation [28, 29, 30], the SyMSim toolkit is specifically adapted to each simulated infrastructure by modifying the program code. This fact is a serious disadvantage that does not allow SyMSim to be used to solve a wide class of problems in data center design and support.

The result of the simulation program operation is a sequence of DB records reflecting all events occurring in the system. These include, for example, job arrival, beginning and end of job processing, beginning and end of file transfer, tape manipulations, etc. All events are described in a common format. Modeling results are presented in the form of a limited number of graphs in the web-interface, conveying only a part of information [26]: amount of incoming data, free space in data storages, load on data transmission channels, which may be insufficient for users.

Thus, the existing SyMSim software complex, despite its main advantage, which is the synthesis of simulation and monitoring processes, has a number of significant disadvantages:

1. the libraries that form the basis of the complex are outdated and do not allow to take into account modern trends in the construction of DDCs;
2. web-interface of the complex allows to set only parameters of job flows, start/stop the modeling process, as well as to view some results on a limited set of charts (to obtain additional results, the data should be exported from the DB);
3. to change the equipment parameters and the structure of the modeled system it is necessary to interact directly with the DB;
4. the input data flow and jobs flow, regardless of the volume, is stored in the DB;
5. the program kernel is adapted to each simulated infrastructure by changing the program code;
6. such important criteria of system functioning as data losses depending on the type of selected equipment, probability of realization of events related to equipment failures and failures, changes in the performance of computing resources and data storages are not taken into account;
7. there is no possibility to search for the equipment configuration satisfying the given criteria.

It can be concluded that the listed disadvantages do not allow applying SyMSim software complex for solving a wide class of problems in designing and supporting DDCs. Therefore, it is necessary to create a new approach, different from the existing tools and software complexes for modeling, which can be used in the design, construction and development of DDCs, taking into account the characteristics of the equipment, as well as data flows and jobs flows.

1.3. Digital twin: definition, application examples

Digital twin is a concept that has emerged quite recently. The first mention of it was in 2002 during Michael Greaves' presentation called "Conceptual Ideal for PLM". The American scientist, talking about product lifecycle management (PLM), informally presented the concept of DT, which is based on the idea that for every physical system there should be some virtual system. This virtual system is a mirror image (duplication, twin) of the physical system. It is assumed that the twin contains all information about the physical system and is connected with it throughout its life cycle. At the same time, there should be some channel between the virtual and physical systems to ensure data exchange and synchronization [1].

The life cycle of any system begins with design, when the system's goals, behavior, and characteristics are analyzed, i.e., some requirements are formulated. At the same time, undesirable behaviors of the system are identified to develop strategies to prevent their occurrence. At this stage, the DT is a

prototype of the future system and is used to verify whether the design meets the formulated requirements. Once the virtual copy of a particular system with all components is completed and verified, the physical system construction phase begins, where the DT will help to adjust the production or installation of system components, thus preventing possible problems during system commissioning, reducing financial costs and time. During the system operation phase, the DT's predictions regarding the behavior of the system under various external factors are verified. When using the system, as a rule, there is a need to replace the components, there are requirements to improve its characteristics, behavior, adding new functional capabilities. It is at this stage that the connection between physical and virtual systems is necessary. To predict the behavior of the system when changing its configurations, to assess performance and possible failures are used DTs [31].

Of course, first of all, we are talking about complex systems consisting of a large number of components with multiple connections between them. The design, development and operation of such systems are complicated not only by their architecture, but also by the non-trivial processes of transferring and processing large amounts of data, as well as by the various technologies on which they are based. It is obvious that complex systems must fulfill their tasks flawlessly and always give the desired results. The risk of such a system failing should be minimal. But, unfortunately, in real life in the process of operation of complex systems there are acute problems that require a quick solution. Therefore, there is a need for a tool that can warn about the probability of possible failures in the system so that they can be corrected in a timely manner. One of such tools is modeling, which allows you to determine how the system will work in different conditions, to test and evaluate the consequences of changes within the system, to find the cause of possible failures, etc. Despite the fact that modeling is the predecessor of DT, this approach does not provide real-time data exchange between the physical system and the model.

Thus, DTs can be defined as computer models that mimic, mirror or duplicate a physical system and follow the life cycle of its physical twin by monitoring, controlling and improving its processes and functions. The DT continuously predicts future system states (e.g., defects, damage, failures) and enables modeling and testing of new configurations. The physical system and its DT must constantly exchange information. Data obtained, e.g., by monitoring the system itself and its operating conditions, allow the application of computational methods to predict failures, test the results of possible solutions, etc. In this way, a predictive modeling approach is formed, where possible failures in the system can be predicted, any modifications to the system can be modeled to prevent errors or find the best solution. However, it is important to realize that DTs are not always fully autonomous and require a lot of human intervention, especially when they are used to test new functions and modifications of physical systems [32].

Currently, there are many examples of the use of DTs all over the world. DTs are developed for different purposes. For example, in transport engineering for creation and operation of vehicles [33], in aerospace industry for preventive maintenance [34] and for creation of internal systems for aviation [35],

in shipbuilding and operation of water transport for virtual testing [36], in railway transport for track construction [37], in oil and gas industry [38] and electric power industry [39], in agriculture for increasing crop yields and optimizing animal care processes [40]. In medicine, organ DTs are very widespread [41], and DTs of medical components play an important role in healthcare [42]. An interesting example of the application of virtual twins is the DT of a city. Such systems are used both for foreign [43] and Russian cities [44]. There are also examples of DT application in logistics [45], for optimizing life cycles of production [46], hospitals [47] and other organizations.

The above is only a small part of the large number of examples of the application of DT in all areas of human activity. In this paper, the research focuses on the study of new trends in the design and application of DD, namely for the design, construction and development of data centers, in particular distributed ones, taking into account the characteristics of data flows and jobs flows.

1.4. Possibilities of using digital twins of data storage and processing centers

In the modern world, in any field of human activity (science, technology, production, social life, etc.), the need to store and process large amounts of data can be traced. In this regard, everywhere began to use DCs. For example, in science, data centers are used for storage and further processing of data from experiments; in business – for IT outsourcing services; in government organizations – for automation of internal processes, such as document management; in security systems of urban environment, transport and industrial enterprises – to ensure the collection, storage and analysis of access control systems, video surveillance, etc. There are many computing systems of different scales for storage and intensive data processing. To design, create and modify such complex systems one cannot do without DT. Let's consider some examples of application of data storage and processing center DTs.

Future Facilities [6], an American company, develops software and creates DTs for DC management. The DT is implemented as a 3D replica of a DC that can simulate its physical behavior in any operating scenario. The DT covers the entire DC ecosystem, including virtual representations of power, cooling components, etc. Such DTs are based on engineering system simulations and allow predicting the impact of changes in DC components through visualization and quantification of performance [48]. These DTs are used to design next-generation facilities, troubleshoot existing facilities, speed up capacity planning processes, and improve overall efficiency and sustainability.

Another American company Sunbird DCIM [7] specializes in creating a 3D copy of the physical DC, which reflects the actual conditions of the DC in real time and provides remote visualization and monitoring capabilities, thus providing faster DC management. This DT contains all the most important information about the physical infrastructure of the DC. In the form of high-quality, scalable 3D images,

the DC contains servers, networking equipment, storage systems, and cabling. About each of the infrastructure components in the DC contains information about the exact location, measure, model, size, configuration, ports and much more. Even ports and cables that are difficult to see physically, such as those in cable trays, can be visualized in the model. In addition, the DT displays current readings of environmental sensors such as temperature, humidity, water, airflow and pressure drop, providing an accurate view of what is happening in the DC. This DT functionality can increase uptime by providing timely information about potential problems and loads on DC components, increase DC efficiency by analyzing current resource consumption and assessing the impact of allowable changes, increase productivity remotely, and avoid the time and expense of physically being in the DC. With automatic collection, storage and reporting, you can receive and analyze data on DC operations in real time.

There are significant disadvantages in the DTs described above:

- virtual copies of DC that are geographically located in the same physical space are presented;
- DC are considered only in terms of engineering.

However, firstly, distributed DCs are used more often lately, and secondly, there is a necessity to model data flows to form a strategy of job flow management, to analyze the amount of used resources in order to timely update the equipment, to prevent the situation when the available equipment will be insufficient to store all data or fast processing in the required time interval, to estimate the amount of resources required to use the DC in certain tasks, according to the requirements of the DC.

Thus, it seems relevant to develop a method and algorithms for creating digital twins of DDCs in order to conduct research in the field of efficiency and reliability of distributed systems, to test various scaling scenarios, taking into account the requirements for data flows and jobs flows, to form and improve the strategy for managing job flows, to analyze the using resources, to estimate the necessary amount of resources for specific tasks according to the requirements for DDCs.

1.5. Conclusions to Chapter 1

This chapter clarifies the concept of a DT for the goals of this research. A DT is a computer model that:

- reflects the DC architecture, as well as processes taking place in the system under consideration and related to data flows and job flows;
- allows one to simulate and test new hardware configurations, thereby providing the opportunity to predict system failures and evaluate the results of solutions to prevent such situations.

As a result of the analysis of existing DTs and DC modeling tools, it can be concluded that at the moment, neither in Russia nor abroad, there is an instrument that enables to create a DDC digital twin

in order to form and improve a strategy for managing job flows, analyzing used resources, and evaluating the required amount of resources for specific tasks according to the requirements for the DC. Thus, there is an urgent task of developing a method and algorithms for constructing DDC digital twins to conduct research on the effectiveness and reliability of DDCs both during design and during operation, to manage DDCs and check various scaling scenarios, taking into account the requirements for data flows and job flows.

Chapter 2. Models, methods and algorithms for creating digital twins of distributed data acquisition, storage and processing centers

2.1. Preliminary remarks

As a rule, the process of constructing a DT is realized in stages. At the first stage, data on the purpose of the physical object (system), its components and operation parameters are collected [49]. At the second stage, the information necessary for building a virtual image of the object (system) under consideration is extracted from the obtained data [50]. At the third stage, modeling methods are used to build a virtual image. The fourth stage, as a rule, is devoted to the verification of the resulting models, verification and validation with the physical prototype [51]. This is a very important stage, since further study of the considered physical object (system) depends on the adequacy of the model. If necessary, the model is corrected. The last, fifth stage of DT construction provides for the creation of a module for demonstrating the results of DT work, which should be open, accessible and presented in an understandable form (diagrams, charts, schemes, etc.) [52]. Thus, the basis of DT is models, and the functioning of DT is provided by tools that are created with the help of various information technologies.

The efficiency of creation, use and development of DT technology is conditioned by the quality of the kernel (model), for the construction of which various approaches are used. There are several classes of models that are mainly used to create DT: physical model, optimization model, simulation model [44]. A physical model provides computer simulation of physical processes occurring in time. Such a model is based on the laws of physics and computer-aided engineering analysis [53]. Optimization model provides the search for optimal values of the target function in the presence of constraints using mathematical methods. Simulation modeling is a research method in which the system under study is replaced by a computer model in which all or some of the processes occurring in the real system are simulated. In order for a DT to cope with its tasks, it is necessary to take into account the probabilistic characteristics of the processes occurring in the modeled system when developing the kernel [54].

At the same time, the technologies that enable the user to interact with the DT also play an important role. The DT should have the functionality that will allow setting, displaying and editing the input data, as well as visualizing and processing the simulation results. For this purpose, it is necessary to develop a methodology and appropriate software and hardware tools. In addition, it is very important to provide criteria for assessing the quality of DT operation, which will be used in the process of creating and modernizing the DT based on the results of its operation.

Thus, it is necessary to describe functional and non-functional requirements to the created digital twin of DDC.

2.2. Requirements for digital twins of distributed data acquisition, storage and processing centers

2.2.1 Kernel requirements

Within the framework of the considered task of creating a DT, which will help to qualitatively conduct research related to the creation and development of modern DDCs, taking into account the characteristics of data flows and jobs flows, it is advisable to develop a unique Kernel. The uniqueness of the kernel lies in its universality and applicability for modeling the processes of data transfer, storage and processing of any DDC. When building the kernel of DT, in which the process of modeling of DDCs will be implemented, it is necessary to take into account modern trends in building distributed systems and adhere to the principle of developing a universal software package, which does not need to be modified for each modeled infrastructure. During modeling it is also necessary to take into account possible probabilities of changes in the system functioning processes depending on the type of equipment (e.g., data loss, performance changes). This peculiarity will allow to apply DT for solving a wide class of problems, to carry out researches connected with designing, scaling, increase of DDC performance, analysis of efficiency and reliability of various processes occurring in DDC during operation period.

The DT should include functionality that allows for the selection of several potentially competitive options from among the many allowable equipment configurations for DDC.

2.2.2 Requirements to the module for user interaction with the digital twin

Of course, digital twins of DDCs cannot be fully autonomous and require human intervention, for example, to describe the modeled infrastructure, when testing various modifications of the distributed system, when searching for equipment configurations according to the available requirements. During the experiments and operation of the DT the structure of the modeled distributed system may be changed. For example, it may be necessary to add or remove computing components, data storages, communication links between the components of the DDC. This means that the developed DT should be equipped with a well-designed, convenient and accessible interface, where the user can not only set the required parameters of the DDC, but also get the results of the DT operation in a clear and structured form, including graphs.

Thus, we can formulate the following functional requirements for the module for user interaction with the DT:

- interaction should be in dialog mode;
- the user should build the DDC infrastructure by locating the necessary objects on the field, moving them to the required area, connecting them with each other by communication links;
- the system should allow saving the image with the DDC infrastructure to the user's computer;

- the module shall provide an opportunity to set parameters of basic configuration of DDC equipment,
- the module shall provide the ability to specify the characteristics of data flows to be processed at the DDC;
- the module shall provide the ability to specify characteristics of job flows to be executed at the DDC;
- the user should be able to edit the data center infrastructure, equipment parameters, data flow and job flow characteristics;
- the module must contain functionality for launching the DT;
- the user should create computational experiments on the DT, which involve searching and selecting the required hardware configuration for solving specific tasks (e.g., searching for the required number of resources for storing data coming from the experimental facility; searching for the required number of computational resources for processing data for a certain period of time, etc.);
- it should be possible to create scenarios for scaling/modifying DDC parameters;
- the results of DT operation should be graphically illustrated and reflect data volumes and distribution of different types of files in storage, using of computing components, load on communication links during data transfer.

The diagram of variants of use of the described module is presented in the figure 1.

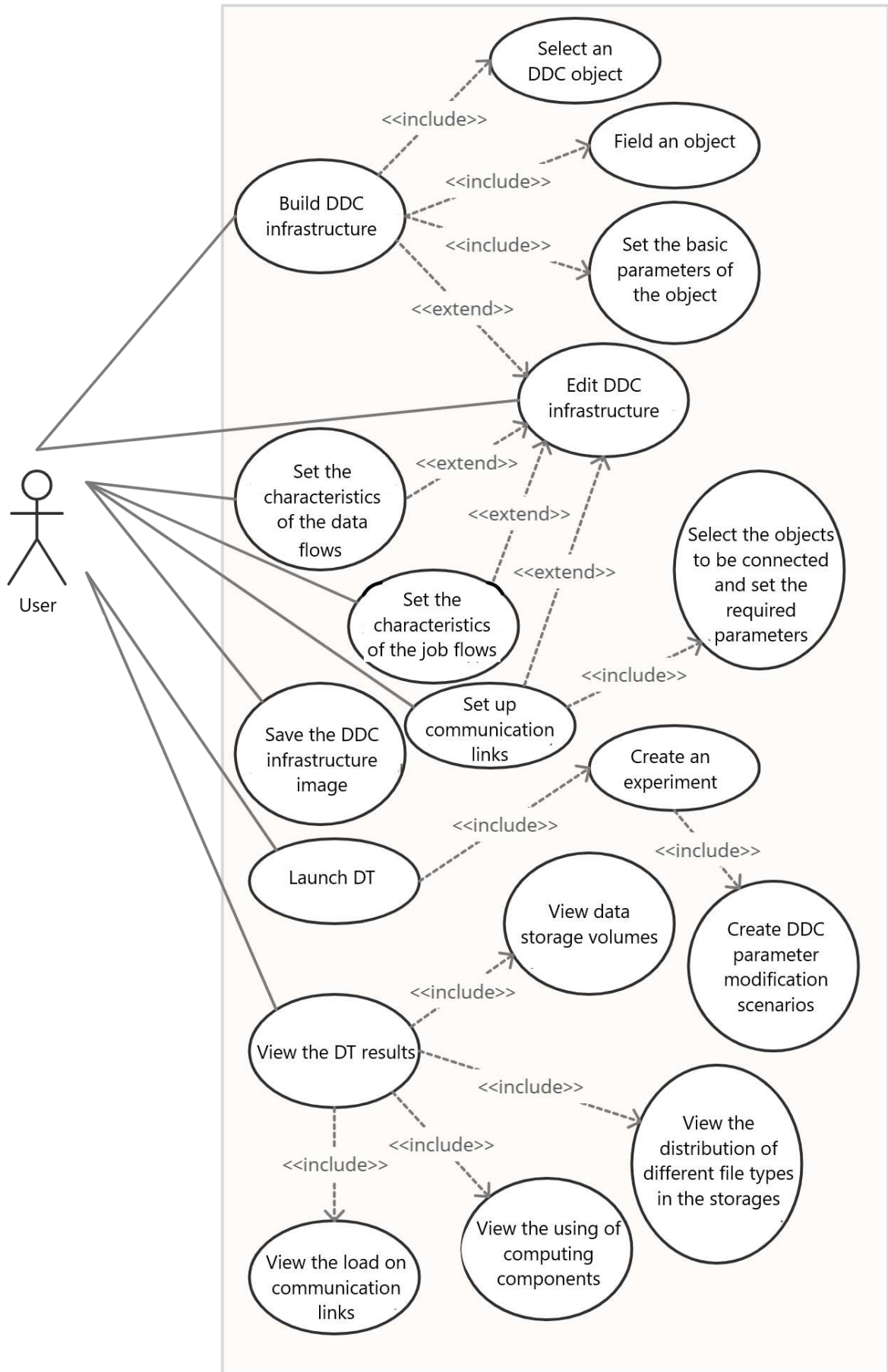


Figure 1. Diagram of variants of using the module for user interaction with the DT

2.2.3 Non-functional requirements for the method of constructing digital twins of DDCs

The following non-functional reliability, security and performance requirements can be attributed to the method of constructing digital twin of DDCs:

- various parameters and results of DT operation should be stored in the DB;
- the speed of operation of the DT from the moment of start-up to the receipt of results should be determined based on the scale of the DDC and the processes occurring in it;
- the adequacy of the DT should be verified by the results of the existing DDC for different parameters available during the system monitoring; the results of the DT should not deviate from the average value of the monitoring data by more than three standard deviations;
- system operability should not depend on third-party software.

2.3. Method of creating digital twins of distributed data acquisition, storage and processing centers

Based on the identified functional requirements, we can formulate a method for constructing digital twins of DDCs. The method consists of the following steps.

1. Obtaining data on DDC: equipment parameters, data flows and job flows, purposes of the DT construction, probabilistic characteristics of the processes occurring in DDC.
2. Description of the DDC structure and the links between the components.
3. Formation of virtual image of DDC on the basis of received data and description of its structure.
4. Modeling of DDC.
5. Graphical presentation of the DT results.

Based on the principles of system analysis [55], the system for implementing the method can be represented as follows:

$$MDT\{D, S, V, M, G, R, Q, W\}, \quad (1)$$

where D — physical objects of DDC, S — data model entities, V — objects of the DDC virtual image, M — objects of modeling processes in DDC (the kernel of DT), G — instances of the module for user interaction with DT, R — relations defining links between physical objects and data model entities, Q — relations defining links between data model entities and objects of the DDC virtual image, W — elations defining links between the results of DT operation and their graphical representations.

Each physical object of an DDC ($d \in D$) has a set of parameters that characterize the components of the distributed centers X , i.e.: $d \in D \subset X_1 \times X_2 \times \dots \times X_N$, where X_i are the properties of physical objects at $i = \overline{1, N}$ ($N \in \mathbb{N}$). Physical objects are capable of changing from one state to another under

the influence of external factors or processes occurring in DDCs, so physical objects are formalized in the following form:

$$D = \{X, \Phi, f\}, \quad (2)$$

where Φ is the set of admissible states of the object, f is the transition function from one state to another, i.e. $f: \Phi \times X \rightarrow \Phi$.

A data model is needed to formally describe the objects of the DDC, where each physical object corresponds to a certain entity ($s \in S$), i.e. $D \xrightarrow{R} S$, where

$$R: \{s \in S | \exists d \in D, s = R(d)\}. \quad (3)$$

The relationships between the components of an DDC are described in the data model as $P: \{s_i P_j s_k\}$, where $i, j, k = \overline{1, N}$ ($N \in \mathbb{N}$). Therefore, the formal description of the data model is as follows:

$$S = \{D, R, P\}. \quad (4)$$

The virtual image of the DDC should be formed based on the parameters of real physical objects and the links between them (P), which is reflected in the data model, i.e. $S \xrightarrow{Q} V$, where

$$Q: \{v \in V | \exists s \in S, v = Q(s)\}. \quad (5)$$

Thus, the virtual image of DDC is formalized as

$$V = \{S, Q, P\}. \quad (6)$$

The main stage of the method of constructing digital twins of DDCs is the modeling process, for the implementation of which objects (data and job generators, data movement scheduler, management object for data processing job, etc.), which together constitute the kernel of the DT ($m \in M$) must be created. The initial data of this stage are the parameters of the virtual objects of the DDC ($v \in V$), each of which has a set of admissible states (U). Each virtual object, similar to its physical image, has a transition function for state change ($h: U \times V \rightarrow U$), which should be taken into account when modeling various processes occurring in DDC. The outputs of the considered stage are formal descriptions of the states of the objects ($z \in Z \subset Y_1 \times Y_2 \times \dots \times Y_N$, where Y_i are the properties of the objects at $i = \overline{1, N}$ ($N \in \mathbb{N}$)) at each moment of time (T). Thus there exists an exit function $k: U \times V \rightarrow Z$. The exit is possible when a given running time is completed, when the job flow process ends, or when resources overflow, which cause the DDC to stop running. Consequently, formally, the kernel of the DT can be represented as follows:

$$M = \{T, V, U, Z, h, k\}. \quad (7)$$

It is worth noting, the uniform (8) or normal (9) distribution of a continuous random variable can be used to generate object parameters such as data volume and job execution time.

$$pr(x) = \frac{1}{b-a}, \quad (8)$$

where $pr(x)$ is the probability density function of a uniform distribution, $[a, b)$ is the interval of acceptable variation of the value of the parameter under consideration.

$$pn(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \quad (9)$$

where $pn(x)$ is the probability density function of the normal distribution, μ is the mean value of the random variable, σ is the standard deviation.

The events of changing the state of DDC objects can include, for example, the transition of equipment to a non-operational state, changes in performance, etc. Events can be described in terms of probability distribution, since the time of their occurrence is a random variable. The flow of events is formed according to Poisson's law, according to which the moments of occurrence of events are distributed. The intervals between the occurrences of events are determined by the formula of exponential distribution:

$$\tau = -\frac{1}{\lambda} * \ln(r), \quad (10)$$

where τ is the interval between random events, λ is the average number of events per unit time, $r \in [0; 1]$ is a uniformly distributed random number.

Characteristics of data flow and job flows, parameters of random events of object state change, types of probability distribution are determined by the user, who is an expert in the field of DDC administration. Evaluating the system reliability and failure probability according to the data of various monitoring systems, the user can change the settings of DT parameters in order to obtain information on data processing time, using resources, etc. to make a decision on the strategy of job flow management or possible redistribution of resources.

Appropriate interfaces ($g \in G$) should be provided for user interaction with the DT in the dialog mode, in particular, for setting various parameters and graphical representation of the results of the DT operation, which can be formally represented as follows:

$$G = \{Z, W\}, \quad (11)$$

where Z are the formal descriptions of the states of the objects, the information about which was stored in the process of DT operation, W are the relations defining the links between the results of DT operation and their further representations ($Z \xrightarrow{W} G$), namely:

$$W: \{g \in G | \exists z \in Z, g = W(z)\}. \quad (12)$$

Based on the results of the DT, the expert can make a decision to select the required equipment configuration from some set of alternatives $\{\psi\}$, based on the selection principle Ω . I.e., the decision making problem can be formalized in the form:

$$\{\{\psi\}, \Omega\} \rightarrow \psi^*, \quad (13)$$

where ψ^* — selected alternative. The principle of selection depends on the goal, which can be the search for equipment configuration and job flow management strategy that ensures processing of all data in the

minimum amount of time, using the least amount of resources, etc. So, it is necessary to find such an alternative that delivers a minimum of one, the most preferred criterion (for example, time), provided that the values of the other criteria are no more than some predetermined values:

$$criteria_1(\psi) \rightarrow \min; criteria_j(\psi) \leq c_j; j = \overline{2, N} (N \in \mathbb{N}), \quad (14)$$

where $criteria_1(\psi)$ is the most preferred criterion, $criteria_j(\psi)$ is the set of criteria, c_j is the given acceptable value of the j -th criterion.

The selected alternative should provide more efficient use of resources and reliable scenarios for scaling and managing the data flows and job flows of DDC. In general, the performance indicator is $\alpha = \langle H_{\text{т}}, T \rangle$, where $H_{\text{т}}$ — target effects, T — time costs.

In order to implement each step of the method, it is necessary to develop algorithms, as well as a data model where the initial data about DDC and the results of the DT will be stored. Further material is presented in accordance with the published article [56].

2.4. Algorithms for implementing of the method of creating digital twins of distributed data acquisition, storage and processing centers

2.4.1 Algorithm of kernel operation

The key element in the method of constructing digital twins of DDCs is the kernel, or the modeling program of distributed systems, which takes into account the parameters of data flows and jobs flows to be processed, as well as probabilistic characteristics of the processes occurring in DDC. Therefore, first of all, it is necessary to describe the algorithm of the simulation program (see Fig. 2).

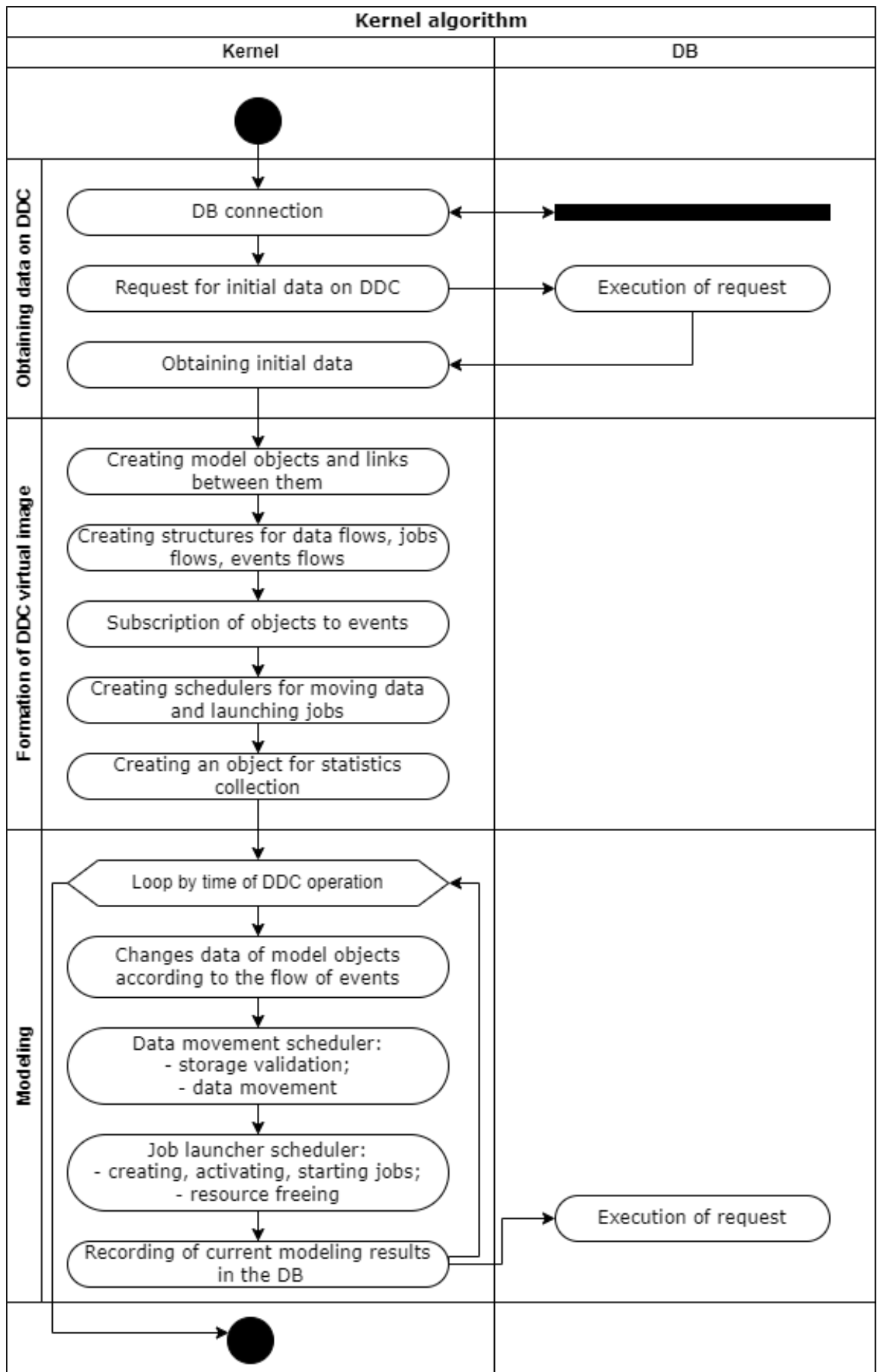


Figure 2. The algorithm of operation of the DDC digital twin kernel

In order for the kernel to be universal and can be used for modeling the processes of data transmission, storage and processing of any DDC, it is necessary to adhere to the principle of developing a universal software package, which does not need to be modified for each modeled infrastructure. In this regard, all input parameters that describe the structure of the DDC, the configuration of its equipment, the characteristics of data flows and job flows should be stored in the DB. In addition, the DB should contain information about possible events occurring in the DDC. An event means, for example, a scheduled equipment shutdown, its temporary cessation of operation, performance changes, etc. Events are not tied to a specific object, because they can occur with any element of the DDC. The event can occur independently of the object, for example, when stopping the whole DDC, or it can be initiated by the object processes, for example, data storage overflow, which occurs when data exceeding the available resources are written. The DB should describe the scenarios of modeling and modification of the DDC equipment parameters, as well as the criteria for collecting statistics about specific objects in the process of operation of the future DT. Thus, the first stage of the algorithm is to set up the connection to the DB and obtain all the necessary information about the modeled system.

The second stage of the algorithm is the formation of a virtual image of the DDC. At this stage, using the initial data obtained from the DB, the creation of objects of the future model and the links between them is carried out. Further, the structures for data flows and jobs flows are defined, the flow of events is generated. Due to the fact that events can occur with any object, it is advisable to introduce the concept of “object subscription to an event”, thanks to which it is possible to control simultaneously all elements of DDC at the moment of occurrence of this or that event. Thus, objects are subscribed to any number of events. At the same stage, objects are created that should be responsible for planning the processes of data and jobs movement between the DDC elements, as well as objects that will collect and record in the DB statistical information about the processes occurring in different DDC elements during the third stage of the algorithm — modeling.

Modeling is carried out by constant time units, the minimum value of which is 1 second. For each model object built at the previous stage of the algorithm, in accordance with the flow of events, changes are made in the data, which correspond to the time unit of DDC operation. The onset and duration of each event is determined according to one of the probability distributions (e.g., Poisson). Data movement and job management schedulers are the key objects that describe the processes occurring in the DDC. At each unit of time, the data movement scheduler checks all storages for new data, which are generated according to the selected probability distribution law. If they are found, the process of data transfer from the current storage to the destinations is started according to the description of the structure of the modeled DDC. The task of the second scheduler is to manage the flow of jobs for processing the available data. At each unit of time, the scheduler creates a certain number of jobs, the execution time of which is

a random variable distributed according to one of the laws (e.g., uniformly), activates and sends previously created jobs for execution. There are two basic models for launching jobs on computational resources. The first model represents sending jobs to queues that are located directly on the counting nodes of computational components. In the second model, there are general queues for each class of jobs that are not assigned to specific computing resources. Therefore, computing components have individual pilots that control the process of launching jobs. A pilot is an algorithm that analyzes the number of free cores on a computational component allocated to a certain type of jobs. If free resources are available, the pilot takes a job from the corresponding queue and sends it for execution. Then the job is removed from the queue, occupies computational resources (slots) and starts to be executed, i.e. processes the input file if it is available in the storage. Each slot includes some subset of cores. The job may require either one or several cores. As a result of job execution, the output file is written to the storage with which the compute resource is associated. When each task is completed, the scheduler frees the computational resources. Throughout the entire modeling stage, data is recorded in the DB, which reflects statistical information about all the ongoing processes.

2.4.2 Algorithm of the module for user interaction with the digital twin

The algorithm of the module for user interaction with the DT is presented in the diagram in Figure 3. User interaction with the DT begins with the construction of the DDC infrastructure. According to the requirements, the user needs to select from a list of objects corresponding to the equipment of the DDC, place it in some area and set up the basic configuration of this device. After that, all information about the object added to the infrastructure is stored in the DB. After all the necessary objects are added, the user must configure the communication links between them. Communication links are logical connections between the components of the DDC, through which the data are transmitted. User should select the objects between which the connection is made and set the required parameters to configure the links. Information about communication links is also added to the DB. Further it is required to create data flows for processing at the DDC and job flows that will process this data. The user can edit the DDC infrastructure, basic equipment configuration, change the parameters of data flows and job flows before proceeding to the next stage, which is to create the DT.

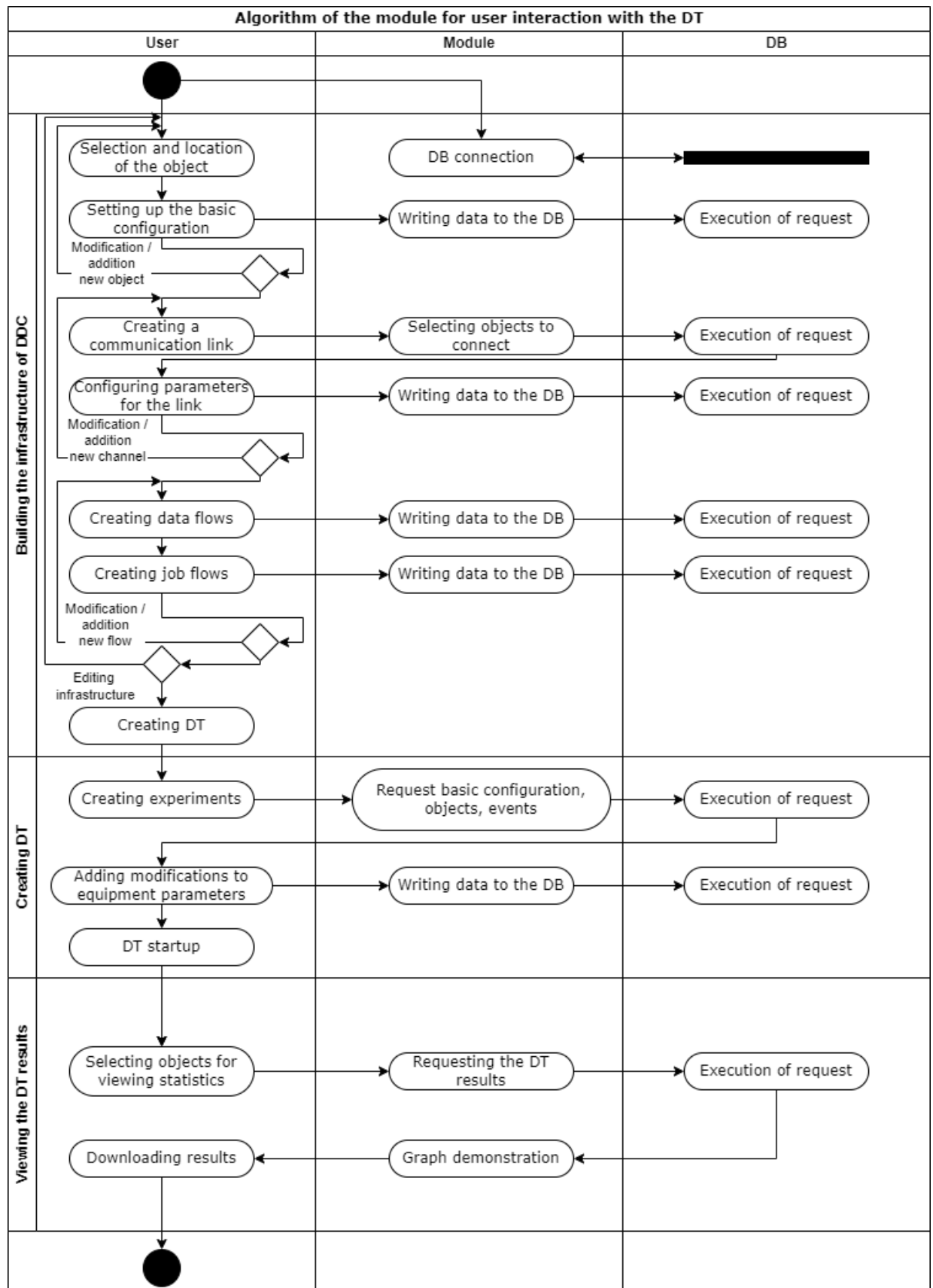


Figure 3. Algorithm of the module for user interaction with the digital twin of DDC

DTs can be created for different tasks, for example, to design an DDC; to check the performance of an existing DDC with current equipment parameters and to find problem areas of its operation; to scale DDC and to find an equipment configuration that will meet certain requirements, etc. In this regard,

the user needs to create computational experiments where the DT will be applied to solve a specific problem. When adding a computational experiment on the DT, it is important to specify objects and events, information about which will be stored in the DB. In each such experiment there will be access to the basic configuration of equipment, which was created when building the DDC infrastructure. At the same time the user can add additional modifications to the equipment parameters, if it is necessary to solve the task at hand. After configuring the computational experiments, the DT is launched. If there are several modifications, the DT for each modification can be launched simultaneously.

In the process or upon completion of the DT operation, the user can view and further analyze the results in order to assess the efficiency of solving the tasks of building and development of DDC. The user should select the type of equipment of interest, after which interactive graphs will be built using data from the DB, which were recorded during the operation of the DT. For example, the graphs can reflect data volumes and distribution of different types of files in storages, using of computing components, load on communication links during data transfer, etc. Interactive graphs imply the ability to zoom and select data for viewing. The user can save the results of the DT operation for any modification in the form of images with graphs, which reflect the changes occurring in the DDC.

2.5. Data model

The DB plays a key role in the development of algorithms for the implementation of the method of creating DTs of DDCs. The DB should store information about the architecture of the DDC, parameters of equipment included in its composition, characteristics of data flows and job flows, events occurring in the DDC, scenarios of possible scaling of the system, and the results of the work of the DT. In this regard, it is necessary to develop a data model, which should include the following entities:

- *DataStorages* — DDC data storages;
- *ComputingComponents* — DDC computing components;
- *Pilots* — pilots, i.e. objects that will launch jobs on DDC computing resources;
- *Slots* — slots that are part of the computing components of the DDC;
- *Links* — DeepL Translate – Самый точный переводчик в мире;
- *DataTags* — types of data stored and processed in DDC;
- *DataFlows* — data flows for storage and processing in the DDC;
- *JobQueues* — job flows that will process the data available in the DDC;
- *TransportJobs* — objects for transferring data flows between DDC storages;
- *Events* — possible events occurring at the DDC;

- *Event_Object* — parameters of probabilities distributions of events occurrence for each object with which this or that event can happen;
- *Sensors* — objects for collecting statistics in the process of DT operation;
- *Experiments* — variants of launching DT for different tasks (computational experiments);
- *Modifications* — modifications (scaling scenarios) of the basic configuration of DDC equipment;
- *SimulationReport* — DT results.

The attributes of the *DataStorages* entity are the parameters of DDC data storages: identifier, name, description, volume, name of the object for statistics collection. For the object of data generation, i.e. the active storage, attributes are required to indicate the fact of activity and the rate of data appearance. For each storage, a priority value should also be set, which indicates the sequence in which devices are viewed during the scheduling of data transfer processes.

The attributes of the *ComputingComponents* entity are the parameters of the DDC computing components: identifier, name, description, number of computing resources (cores), name of the object for statistics collection. If the computing component has a possibility to reduce the time spent on job execution, an attribute containing the value of the speed up factor is required.

The attributes of the *Pilots* entity are the parameters of pilots that will run jobs on the DDC computing resources: identifier, name, description, name of the object for statistics collection. To identify the resource on which the pilot will work, an attribute specifying the name of the computational component is required. According to the kernel algorithm, there are individual pilots for each class of jobs on computational components, so the entity under consideration must contain an attribute that specifies the name of the job flow. Each pilot should have a priority value that indicates the sequence in which entities are viewed during the scheduling of data processing job startup processes. To divide job flows into several computational components, an attribute is introduced that specifies the fraction of jobs from the total flow that should be picked up by the pilot from the queue. Due to the fact that there can be several types of resources for data storage in the DDC, attributes indicating the name of the data storage for reading the initial data, for processing and recording the results of computational jobs execution are added.

The attributes of the *Slots* entity are the parameters of the slots that are part of the DDC computing components: identifier, name, number of cores allocated to a particular pilot, name of the object for statistics collection. It is important to note that the total number of cores in all slots belonging to one computing component must not exceed the total number of cores on the resource. The attribute responsible for the slot activity is necessary to realize the possibility of changing the number of computational resources in the process of DDC operation.

The attributes of the *Links* entity are the parameters of the links between the components of the DDC: identifier, name, description, bandwidth, name of the object for statistics collection. As a rule, a link connects objects of different entities, the names of which are indicated in the corresponding attributes. To realize the possible event of communication link disconnection, an attribute is provided, by means of which the entity becomes inactive.

Attributes of *DataTags* entity are parameters of data types stored and processed in DDC: identifier, name, description. The attributes of the *DataFlows* entity are the parameters of data flows to be stored and processed in the DDC: identifier, name, description, data type, name of the object for statistics collection. Each flow is bound to a specific storage to which data will be written. A certain amount of resources is allocated for each flow in the storage, in case of overflow of which the flow status will change to inactive for recording. It is important to note that the total allowable volume of data flows belonging to one storage must not exceed the total storage resource volume. When creating a DT may be necessary to consider the variant of work DDC, in which the data stores have already written a certain number of files for processing. For the described case, this entity has a corresponding attribute.

The attributes of the *JobQueues* entity are the parameters of the job flows that will process the data available in the DDC: identifier, name, description, name of the object for statistics collection. For each type of tasks such values as: type of input/output data, average volume of input/output data, average time of job execution should be defined. In addition, the number of jobs in flows may differ, some jobs may be started more often, and some jobs may start executing only after others are finished. All these details are reflected in the attributes. It is important to provide for the possibility of random generation of data files and jobs for their processing, therefore, attributes defining permissible limits for changing certain values have been added to the entity.

The attributes of the *TransportJobs* entity are the parameters of the objects for transferring data flows between DDC storages: name, names of storage resources between which data are transferred and the communication link connecting them, type of transferred data. For each object, a priority value must be set, which indicates the sequence of device viewing during the planning of data transfer processes. It is possible to divide data flows into several distributed storages by means of an attribute, where the percentage of data from the common flow to be transferred to certain resources will be specified. The attribute of delayed start of data transfer is introduced to realize the situation of redistribution of storage resources.

The attributes of the *Events* entity are the parameters describing possible events occurring in the DDC: identifier, name, description, name of the object for statistics collection. Such events can be, for example, stopping/starting one of the DDC elements, increasing/decreasing the number of available computing or storage resources and others. The attributes of the *Event_Object* entity are the parameters of event probability distributions for each object with which an event can occur: event identifier, name

and type of object, which can be data storages, computing components, slots, communication links. The main attributes are the distribution type and the value of the event occurrence probability, which can differ not only depending on the object type, but also on its purpose. For realization of dependent events, additional attributes are provided.

The attributes of the *Sensors* entity are the parameters of the objects for collecting statistics during the operation of the DT. For each type of element included in the DDC: data storages, computing components, pilots, slots, communication links, data flows, job flows, events — a special program object of statistics collection is provided, having an identifier, name and description. The collected information should be averaged and saved with a specified frequency, which will speed up the recording process and save space in the DB.

The attributes of the *Experiments* entity are the parameters of variants of the DT launch for different tasks: identifier, name, description, launch parameters, date and time of creation, as well as the list of identifiers of the objects of statistics collection, which can be changed according to the needs of the DT use. The attributes of the *Modifications* entity are the parameters of modifications (scaling scenarios) of the basic configuration of the DDC equipment: identifier, parameters of the DDC equipments, which are changed in accordance with the scenario relative to the basic configuration, status, as well as the date and time of creation, start, and completion of the DT operation. To identify the records with the DT results according to a particular scenario, the corresponding attribute is added, which contains the value of the identifiers of the CD start and the scenario.

Attributes of the *SimulationReport* entity are parameters describing the DT results. Each record contains the identifier, system time of statistics collection, identifier and name of the object, which is part of the DDC, information about the results of work of which is contained in the record. For each type of objects, the data are recorded, which are of the greatest interest for tracking the processes occurring in the DDC in accordance with the description of the objects of statistics collection. The results of all variants of starting the DT with different modifications are stored in one table, therefore an attribute is provided to identify the record of each scenario.

A full description of the attributes of the listed entities is provided in the Appendix 1. Notes are added for each attribute, according to which a physical data model is developed for the database implementation. Figure 4 presents a logical data model showing all described entities, their attributes and relationships.

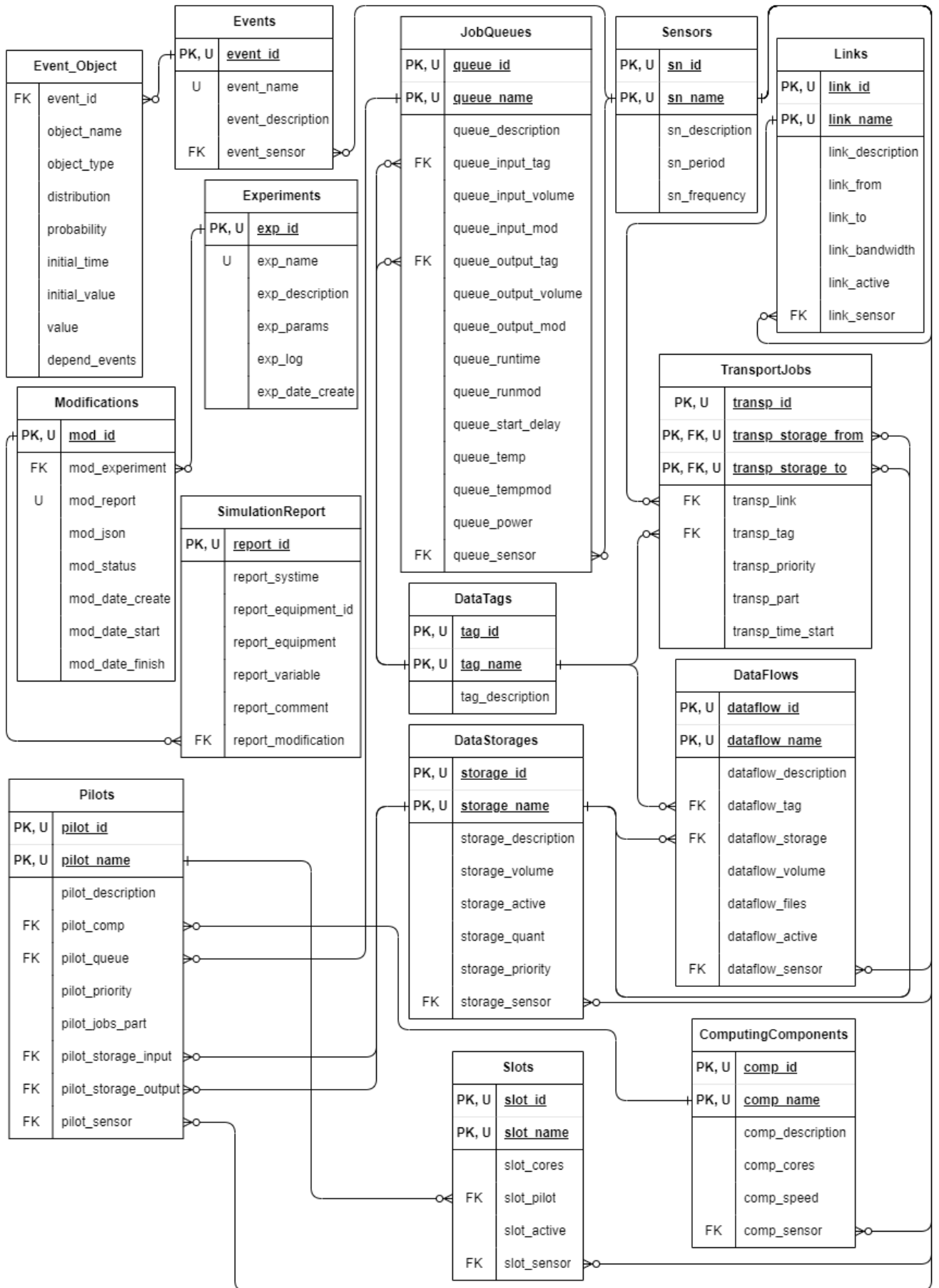


Figure 4. Logical data model

2.6. Conclusions to Chapter 2

Functional and non-functional requirements for DDC digital twins are defined. The requirements include requirements for the core, which is a simulation program, and requirements for the module for user interaction with the DT. The main requirements of the core are universality, adequacy of modeling taking into account the characteristics of data flows and job flows, probabilistic processes of the system functioning, as well as applicability for modeling the processes of data transfer, storage and processing of any DDC. The requirements for the module for user interaction with the DT include the visibility of the presentation of source data, the modeling process and results, as well as the release of a user from participation in the development and maintenance of the DT core. The requirements are formulated in accordance with the use cases of the future DT, the main task of which is to assess the effectiveness of solving the tasks of building and developing a DDC.

As a result, a new method for creating and using DTs is formulated to solve the tasks of managing and developing DDCs, including improving their technical characteristics. A formal description of the method is presented. The proposed method differs from existing ones in the ability to simulate such processes as data storage and processing, taking into account the characteristics of data flows and jobs, the probabilities of failures and changes in the equipment performance and other processes occurring in the simulated system.

To implement the method, algorithms for the core operation and the user module are developed. The algorithm of the core can be divided into three stages: obtaining data about the DDC, forming a virtual image of the DDC, and modeling. The algorithm of the module for user interaction with the DT includes the construction of a DDC infrastructure, the creation of a DT, and viewing the results of the DT work.

A data model is created, entities are presented. The entities describe the architecture of the DDC, the equipment included in it, data flows and job flows, events occurring in the DDC, scenarios for the possible scaling of the system, and the results of the DT work.

The first provision to be defended is proved: “A method for constructing a digital twin is developed. The method allows describing distributed data acquisition, storage and processing centers, taking into account data flows and job flows, as well as processes occurring in the DDC”.

Chapter 3. Implementation of algorithms and development of special software for creating digital twins and interacting with them

3.1. Implementation tools

It is necessary to define a list of technologies to implement the method of building digital twins of DDCs. First of all, it is necessary to choose a programming language that will allow to create a cross-platform software product. Python — an interpreted object-oriented high-level programming language with dynamic typing and automatic memory management — is chosen from the set of existing languages [57]. Its advantages for developing complex algorithms with multiple methods are the availability of a variety of libraries that can be used to process command line arguments, encode and decode data into various formats, work with high-level mathematical functions, and finally interact with database management systems (DBMS). The Python libraries used will be described in more detail in the following sections when the developed algorithms are described in detail.

The Python programming language can be used to create web services that can be conveniently used for user interaction with the DT. For example, the freely distributed open source web framework Django supports Object-Relational Mapping (ORM) technology for accessing database entities [58]. Django's architecture is called Model Template View (MVT, model-template-representation) because models map the data structures needed by the application to tables in the DB, templates are responsible for presenting the data to the user, and the view formats the data from the models into a specific view and passes it to the templates [59]. The user interface in this case is conveniently implemented as adaptive web pages using the JavaScript programming language and the Bootstrap, Cytoscape.js and Plotly libraries. The libraries used will be described in more detail in the following sections when describing the developed algorithms in detail.

The freely distributed object-relational DBMS PostgreSQL [60] was chosen for storing the data required for creating, launching and functioning of the DT. The physical realization of the DB is based on a detailed description of the attributes of the entities of the developed data model, which is presented in the Appendix 1.

3.2. Implementation of the digital twin kernel

Due to the fact that the DT kernel, in which the process of DDC modeling will be implemented, should be universal for modeling the processes of data transmission, storage and processing of any DDC, it is necessary to adhere to the principle of developing a universal software package, which does not need to be modified for each modeled infrastructure. The object-oriented approach is used for the development of the DT kernel.

All objects from the data model are described by the abstract base class *Object_DC* (see Fig. 5) which includes general properties of the DDC elements, such as identifier (*id*), name (*name*), description (*description*), priority (*priority*), active status (*active*) and the list of events (*events*) that can occur. The class implements the constructor and methods of getting statistical information about the object functioning (*GetStatistics*), adding an event (*AddEvent*) and sorting events by the nearest time of occurrence (*SortEventByMinTime*). The methods of editing object parameters (*EditObject*) and processing events (*ProcessEvents*) are abstract and are implemented in the base class inheritors: *DataStorages*, *ComputingComponents*, *Pilots*, *Slots*, *Links*, *DataFlows*, *JobQueues*, *TransportJobs*, *Events*, *Sensor*.

Figure 6 shows a diagram demonstrating the general structure of the hierarchy of kernel classes, which describe the entities of the described data model, as well as additional objects that allow to implement the process of modeling the work of DDCs.

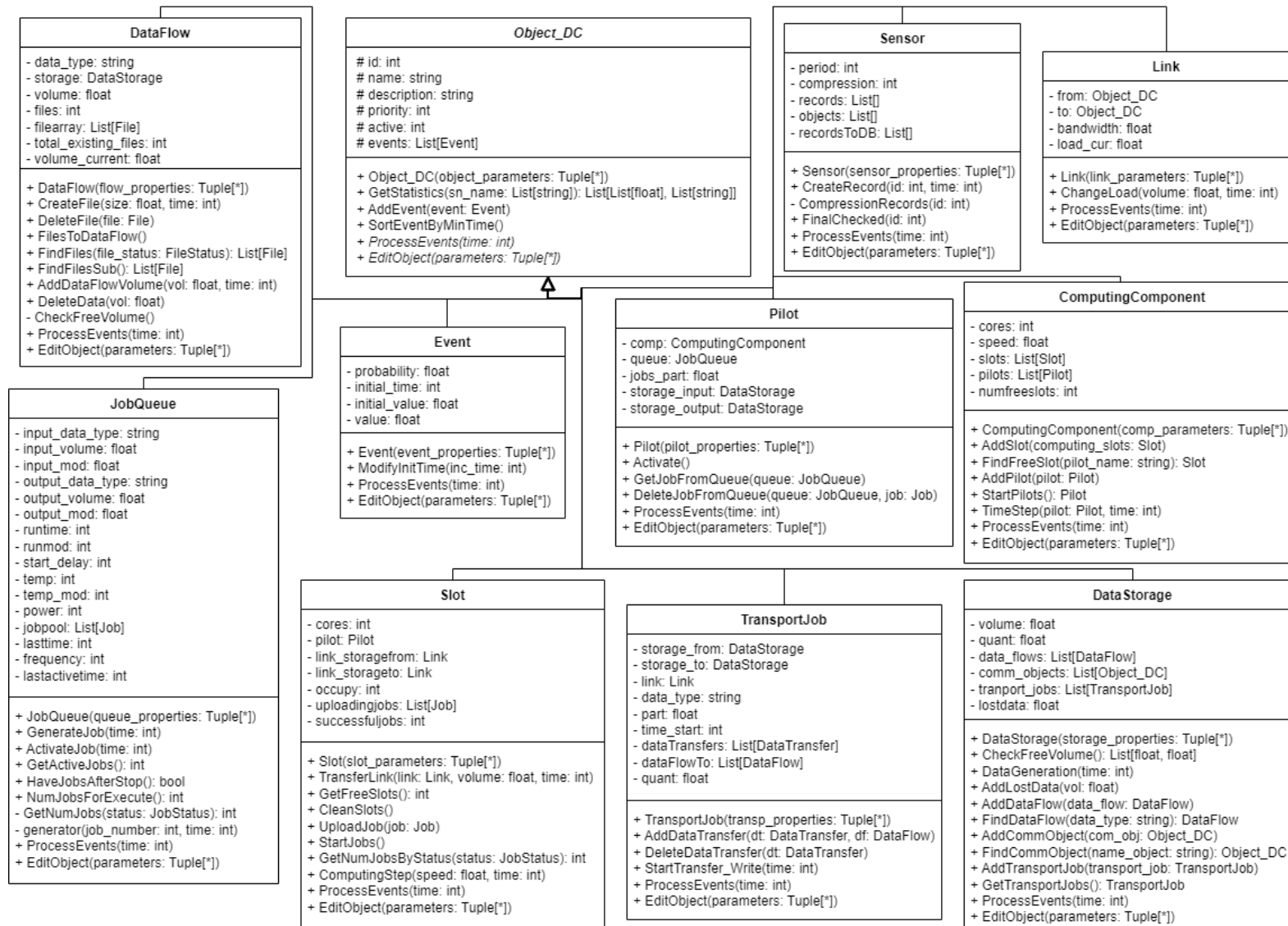


Figure 5. Class diagram describing the inheritance hierarchy

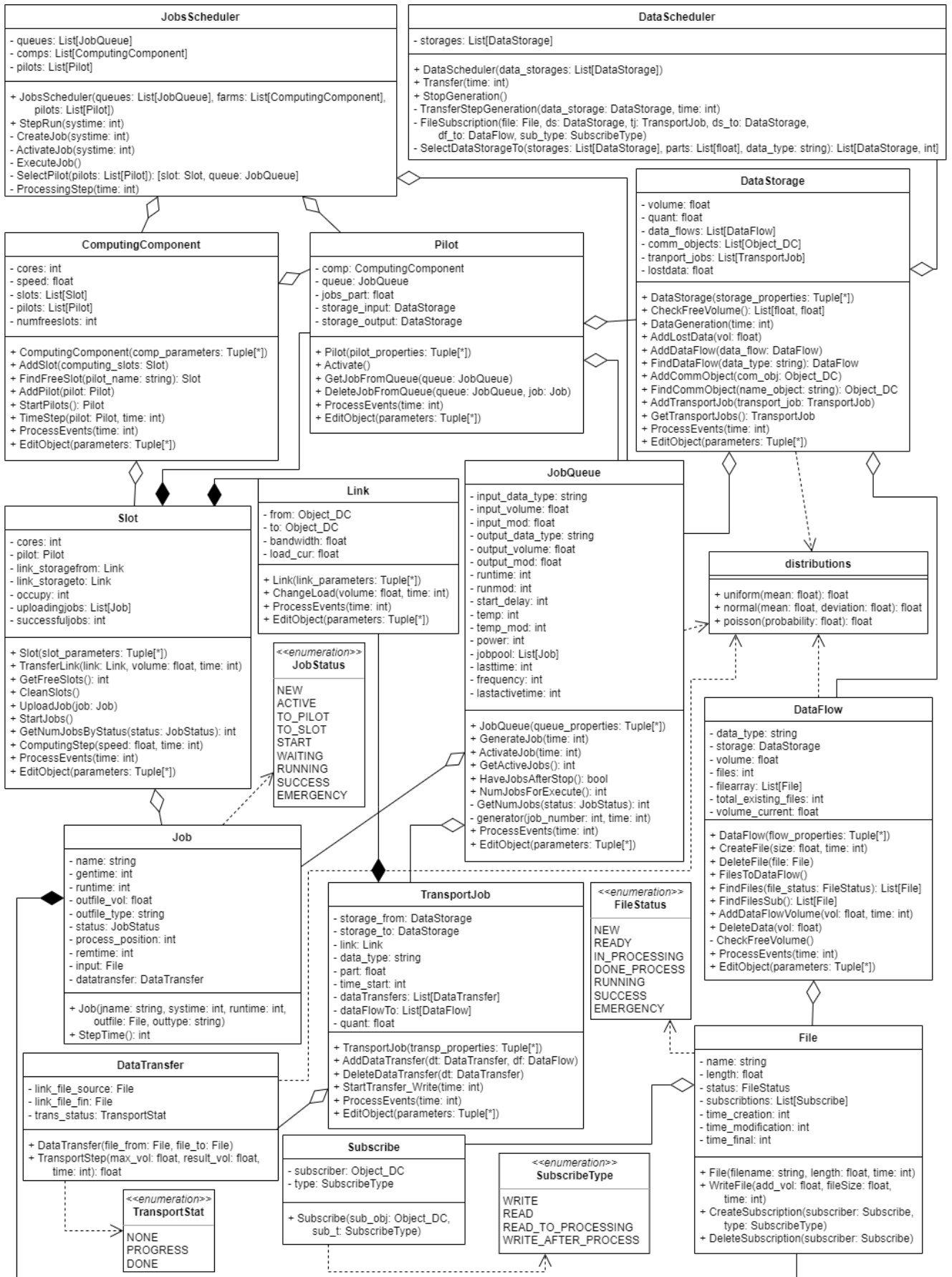


Figure 6. Detailed class diagram for the DT kernel

The *DataStorage* class describes data storages of DDC. Distinctive properties of the object are the maximum storage volume (*volume*) and the rate of data generation per unit of time (*quant*), if the object describes a data generator. A data generator can be subscribed to an event that changes its parameters. The data generator has no memory for storing information, so if by the beginning of the next generation event the data volume of the generator is not equal to zero, the information is considered lost. In this regard, a data loss counter (*lostdata*) is required. Data in storages are managed by flows, which are described in the *DataFlow* class. Each flow is bound to a specific storage (*storage*) to which data of a certain type (*data_type*) will be written. Each flow in the storage is allocated a certain volume of resources (*volume*) for each thread, in case of overflow of which the flow status will change to inactive for writing. Data in a flow are stored as files (*files*).

Data is moved between storages using objects described in the *TransportJob* class. The fields are the type of transferred data (*data_type*), pointers to the storages between which the data are moved (*storage_from* and *storage_to*), and a pointer to the data transfer link (*link*). To implement the process of dividing data flows into several storages, a field (*part*) is created, which contains the value of the percentage of data from the total flow to be transferred to certain resources. There is a field of delayed start of data transfer (*time_start*) to realize the situation of reallocation of storage resources. The *DataTransfer* class was created to detail the process of transferring data files. The fields are pointers to files for data transfer (*link_file_source*) and reception (*link_file_fin*), as well as transfer status (*trans_status*). Possible statuses are listed in the *TransportStat* element (no transmission, transmission in progress, transmission completed). Communication links between DDC objects for data transfer are described in the *Link* class. The characteristics of the link are bandwidth (*bandwidth*) and references to the infrastructure objects between which the link is performed (*from* и *to*).

The *File* class describes data files. Each file has a name (*name*), a size (*length*), a creation time (*time_creation*), a modification time (*time_modification*), and a completion time (*time_final*). Since a file can be used to write or read for processing, a field is added to reflect its status (*status*), as well as a field describing the subscriptions (*subscriptions*) object that requires the file. The *FileStatus* element lists the possible statuses of the file: new, ready for use, in process of transfer, transferred, in process, processed successfully, processed failed. The subscriber object can be either a data storage or a computing component, so the *SubscribeType* element is introduced, which lists the types of acceptable file usage: write, read, read for processing, write after processing.

The *ComputingComponent* class describes the computing components of an DDC. Each computing component contains a certain number of cores (*cores*) on which various jobs will be executed. To possibly reduce the time spent on job execution on a computing component, a field describing the speed up factor (*speed*) is created. The cores of the computational component are combined into slots, which are described in the *Slot* class. It is on these slots that data processing jobs are performed. A certain

number of cores (*cores*) is allocated for jobs of different types. In this case, we consider a model with general queues for each class of jobs, not assigned to specific computing resources, so the process of launching jobs is controlled by pilots, which are described by the *Pilot* class. Each pilot works only within one computational component (*comp*) and controls only a certain type of jobs, which are formed in some queues (*queue*). Since all jobs from the queue can be divided among several pilots, a field (*jobs_part*) is added to the class to specify the percentage of jobs to be taken by a pilot from the queue. Additionally, fields indicating the data storages for reading the input data for processing (*storage_input*) and recording the results of computational jobs (*storage_output*) have been added to the class.

Data processing tasks are described in the *Job* class. The job is characterized by such class fields as name (*name*), generation time (*gentime*), runtime (*runtime*), output data volume (*outfile_vol*), output data type (*outfile_type*), executed operations counter (*process_position*). The job must process a file, so a field that contains a reference to the input data for processing (*input*) and a field that contains a reference to the object detailing the process of file transfer from storage (*datatransfer*) are added. The *JobStatus* element is introduced to track the status of job execution (*status*). Possible variants of job status: new, activated, taken by pilot, sent to execution slot, started, waiting for file for processing, running, completed successfully, completed with error. After creation, all jobs fall into queues (flows), which are described by the *JobQueue* class. For each queue are defined such characteristics as: input/output data type (*input_data_type* and *output_data_type*), average input/output data volume (*input_volume* and *output_volume*), average job execution time (*runtime*). Besides, the number of jobs in flows may differ, some jobs may be started more often, and some jobs may start executing only after others are finished, that's why the corresponding fields (*power*, *temp*, *start_delay*) are added to the class. fields for values of permissible modification of input/output data volumes (*input_mod* and *output_mod*), job execution time (*runtime_mod*) and start frequency (*temp_mod*) have been added to the class.

To describe possible events occurring in DDC, the *Event* class is created. The class fields are probability of event occurrence (*probability*), time of occurrence (*initial_time*) and permissible values for changing object properties (*initial_value*, *value*). Objects of the *Sensor* class are required, which are characterized by the averaging period of the collected information (*period*) and the frequency of writing to the DB (*compression*), to collect statistics in the process of operation of the DB.

The key elements in the implemented kernel are the data movement scheduler (*DataScheduler* class) and the job management scheduler (*JobsScheduler* class). The life cycle of these elements and their main functions are shown in Figures 7 and 8 respectively. A field of the *DataScheduler* class is a list of references to objects (*storages*) that are data stores, since the data movement scheduler must control any operations on files. At each unit of time, the data movement scheduler checks the health of all stores, which in turn performs event processing and then responds with the status of their activity. The data storages are then checked for new data that are generated in the data flows according to the required

probability distribution law. In case of their detection, the process of data transfer from the current storage (*Transfer()*) or from the generator (*TransferStepGeneration()*), if it is provided by the computing infrastructure, to the destinations (*SelectDataStorageTo()*) according to the list of objects requesting a certain file (*FileSubscription()*) is started. The fields of the *JobsScheduler* class are lists of references to objects that contain queues of jobs to be executed (*queues*), and are responsible for launching jobs (*pilots*) on specific computational resources (*comps*). At each time unit, the scheduler creates a certain number of jobs (*CreateJob()*), the execution time of which is a random variable distributed according to a specified law, activates previously created jobs (*ActivateJob()*) and sends them to the computing components for execution in the queue. Next, a pilot is selected to run the job on the computational component (*SelectPilot()*). The pilot analyzes the number of free cores on the computational component. If free resources are available, a job is executed (*ExecuteJob()*): the pilot takes a job from the corresponding queue and sends it for execution on computational resources, in the process of which the input file is processed if it is available in the storage (*ProcessingStep()*). When each job is completed, the scheduler removes it from the queue and frees computing resources.

Generation of all data and jobs, event occurrence times is performed using the *distributions* class, where methods of random values generation according to different distributions (uniform (*uniform()*), normal (*normal()*), Poisson (*poisson()*)) are implemented using the module of work with high-level mathematical functions and multidimensional arrays *NumPy* [61].

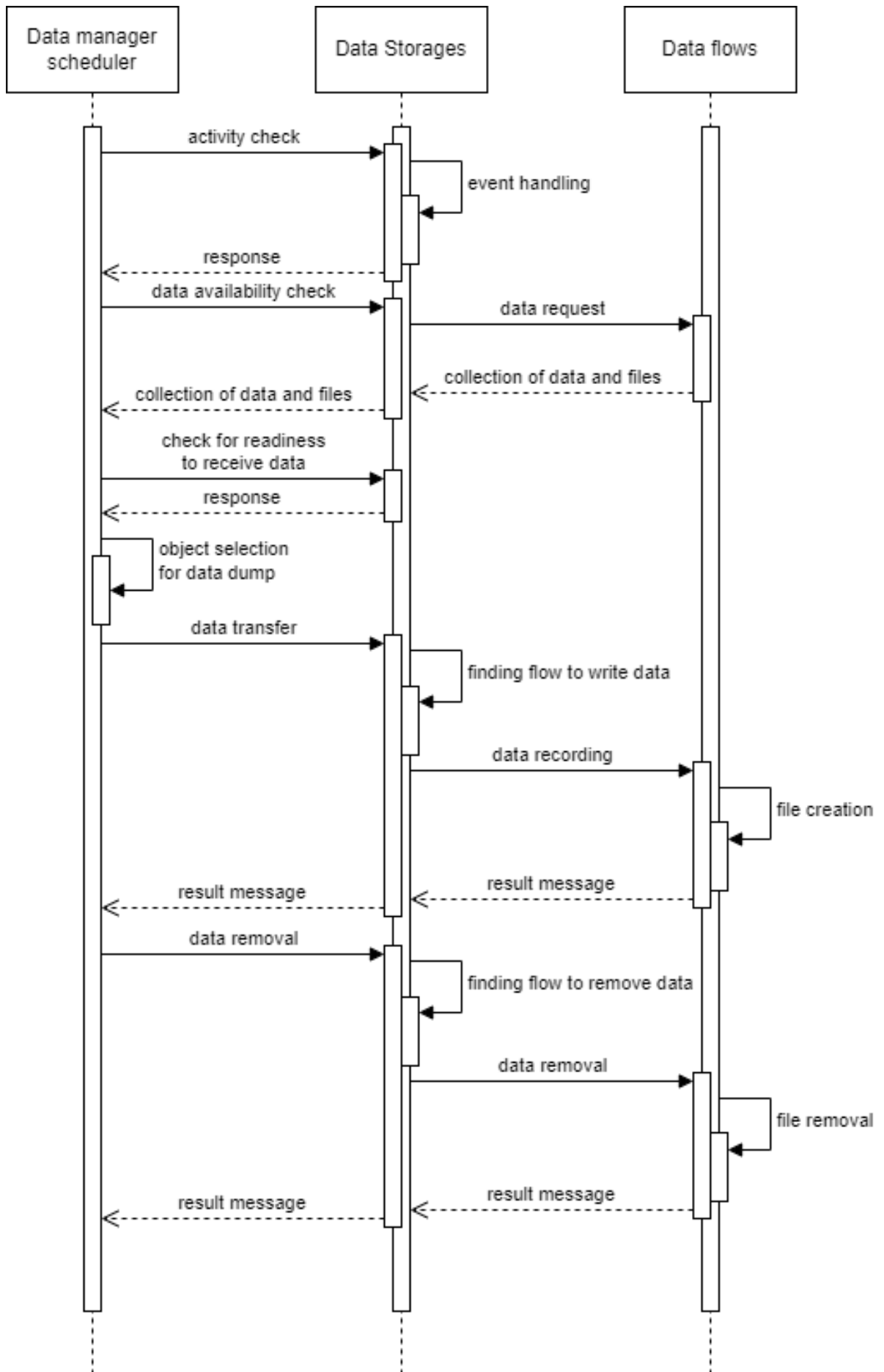


Figure 7. Sequence diagram for the data manager scheduler

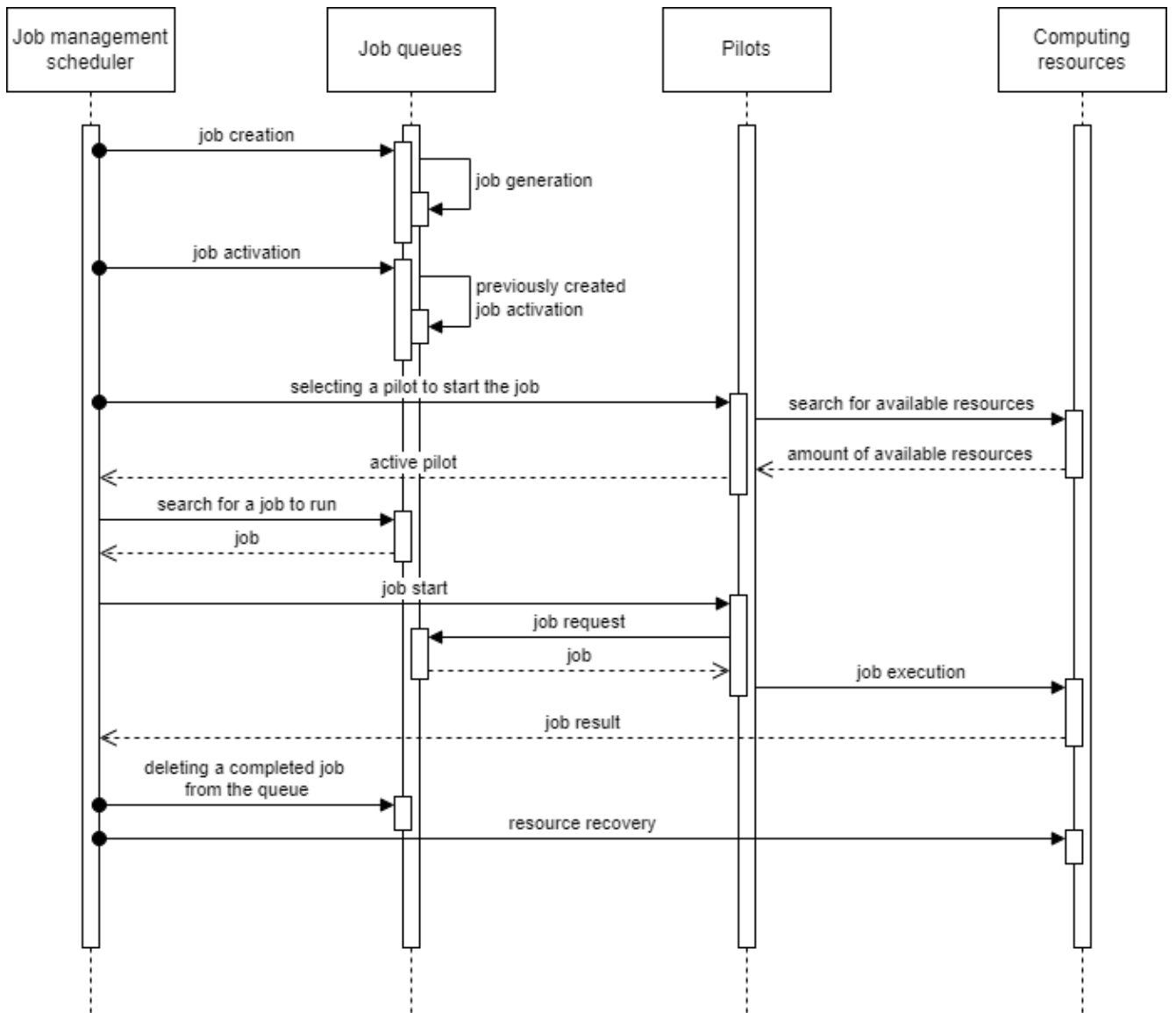


Figure 8. Sequence diagram for the job management scheduler

Now let's consider the general principle of kernel operation, its relations with the described classes and additional parameters that are necessary for its operation, which is reflected in Figure 9.

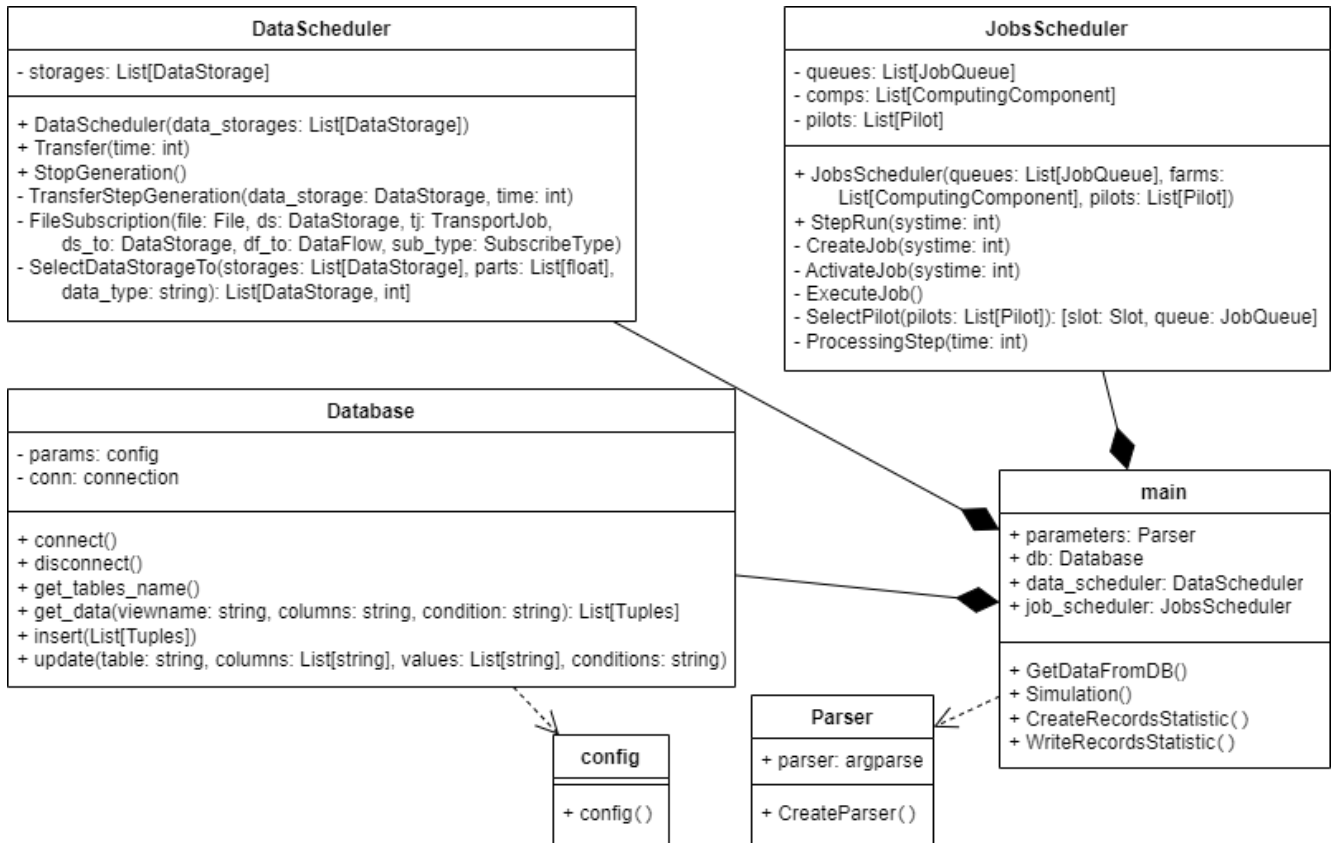


Figure 9. General class diagram for the DT kernel

The developed program should be run using the command line, so the *Parser* class was developed. The only method of the class with the help of the *argparse* library [62] processes the specified arguments, which are necessary for further operation of the program. Such arguments include: the operating time of the experimental facility in hours for which the computational infrastructure is created (*simulation_time*); the time resolution factor (*resolution_factor*), which is necessary to speed up the program and obtain results in a shorter time (calculations are performed not for each second of astronomical time, but for the time indicated by this factor); the identifier of the DT for logging the results of its operation (*log_id*).

The first step after starting the program is to configure the connection to the DB and get all the necessary information about the DDC. The DB connection settings are described in a special file with the extension *.ini*, where such parameters as host, port, user name, password and DB name are specified. A single method of the *config* class parses the parameters of the configuration file using the *configparser* module [63]. The *Database* class describes the connection to the DB and, using the *psycopg2* module, implements the methods of interaction with the PostgreSQL DBMS: connect and disconnect (*connect()* and *disconnect()*), get the list of tables (*get_tables_name()*), get data from a given table (*get_data()*), add new data (*insert()*), update data (*update()*).

The second stage of the algorithm is the formation of a virtual image of the DDC. At this stage, using the initial data obtained from the DB, the creation of objects of the future model and links between

them is carried out. Further, structures for data flows and job flows are defined, a flow of events is generated, to which DDC objects are subscribed. Schedulers of data and job management processes are created, as well as objects that will collect and record in the DB statistical information about the processes occurring in various elements of the DDC.

The third stage is modeling. Modeling is carried out by constant time units, the minimum value of which is 1 second. Since the program is designed to analyze the flow of data and jobs in the system, the methods of the data transfer scheduler (*data_scheduler.Transfer()*) and job management scheduler (*job_scheduler.StepRun(system_time)*) are called. For each model object built in the previous step of the algorithm, changes are made to the data according to the flow of events that correspond to the time unit of the DDC operation. Throughout the entire modeling stage, statistical information about all ongoing processes is obtained (*CreateRecordsStatistic()*) and data are written to the DB (*WriteRecordsStatistic()*).

In this way the kernel of the DT is developed taking into account all the declared requirements.

3.3. Implementation of a module for user interaction with a digital twin

The module for user interaction with the DT functions in a dialog mode. The module is realized in the form of adaptive web pages taking into account all the requirements. HTML, CSS, Bootstrap and JavaScript technologies were used. Each page is described in a separate template for ease of further use when building the project in the special software.

User interaction with the DT begins with building the DDC infrastructure (see Fig. 10). The page contains a list of objects corresponding to possible DDC equipment, as well as a field for locating the required objects and connecting them to the common infrastructure. Moving an object to the field opens a form for entering its parameters. When you click on the “Add device” button, all information about the object will be saved in the DB. This is how the basic configuration is set up. After all the necessary objects are added, the user must configure the communication links between them. Communication links are understood as logical connections between the components of DDC, through which data are transmitted. To configure the links, select the objects between which the connection is made and set the required parameters (see Fig. 11). To create data flows and job flows by analogy, forms for entering the required parameters are created. Information about communication links, data flows and job flows should also be added to the DB. The user can edit the DDC infrastructure, basic equipment configuration, change the parameters of data flows and job flows before proceeding to the next stage, which is to create the DDC.



Figure 10. Page for building DDC infrastructure and setting up basic equipment configuration



Figure 11. Page for building DDC infrastructure and setting up communication links between objects

Building DDC infrastructure is one of the most important functions. The process involves animated arrangement of objects on a form (HTML canvas element) on which they can be moved, connected, selected. Configuration building resembles the process of building a weighted graph, where the added objects are vertices, and the links between them, which describe the communication links with bandwidth — edges with weights. Due to this feature, the Cytoscape.js library, which is focused on graph and network construction, was chosen to implement the functionality. The library has built-in

rendering with gestures and events, and allows manipulating highly customizable and interactive graphs [64].

A corresponding simple HTML page for adding computational experiments on the DT has been created to describe the purpose of creating the DT and entering additional parameters for the modeling process (see Fig. 12). In each computational experiment there is access to the basic configuration of the equipment, which was created during the construction of the DDC infrastructure, also the user can add additional modifications to the equipment parameters and enter the events occurring in the DDC. After configuring the computational experiments, the DT is launched (see Fig. 13).

Добавление эксперимента

Заполните поля формы, чтобы добавить новый эксперимент для поиска оптимальной конфигурации оборудования

* Обязательное поле для заполнения

Название эксперимента *

Описание эксперимента

Параметры моделирования

- Продолжительность работы моделируемой инфраструктуры – ч.
- Ускорение процесса моделирования в раз.

Параметры логирования

Выберите объекты и события, о которых необходимо сохранять информацию во время моделирования

- Объекты моделируемой инфраструктуры
 - Хранилища данных
 - Вычислительные компоненты
 - Каналы связи
- События
 - Генерация данных
 - Потери данных
 - Работа с файлами
 - Генерация, запуск, выполнение задач

Добавить

Очистить

Отмена

Figure 12. Page for adding a computational experiment

Информация об эксперименте
Дата создания: 7 февраля 2023 г., 10:36

Название эксперимента
Test 1

Описание эксперимента
Поиск оптимального количества ресурсов для хранения данных.

Параметры моделирования

- Продолжительность работы моделируемой инфраструктуры – 500 ч.
- Ускорение процесса моделирования в 1000 раз.

Параметры логирования

- Объекты моделируемой инфраструктуры
 - Хранилища данных
 - Вычислительные компоненты
 - Каналы связи
- События
 - Генерация данных
 - Потери данных
 - Работа с файлами
 - Генерация, запуск, выполнение задач

[Посмотреть результаты](#)

[Выбрать другой эксперимент](#)

Базовая конфигурация

Хранилища данных		
Название	Описание	Объем (ТБ)
trigger	Trigger BPH	10000,0
buffer	Data reception buffer	5405,0
eoslhp	Main storage LNER	1000,0
eoslit	Main storage LIT	1000,0
dcach	DP	1000,0

Вычислительные компоненты		
Название	Описание	Количество ядер
tzlit	LIT T2 farm	598
psxlhp	LNER main farm	1298
super	Governor	198

Каналы связи		
Название	Описание	Способность (ГБ/с)
rm0	trigger - buffer	100,0
rm1	buffer - lhp	10,0
rm2	buffer - lit	10,0
compute0	lhp - farm lhp	10,0
compute1	lit - governor	10,0
compute2	lit - farm lit	10,0
dataeoslhpelit	eoslhp - eoslit	10,0

[Добавить модификацию](#)

Список модификаций

#	Статус	Дата обновления	
16	NEW	9 марта 2023 г., 14:52	Просмотр Запуск Результаты
15	DONE	10 марта 2023 г., 10:18	Просмотр Запуск Результаты

Список событий

[Добавить событие](#)

Figure 13. Page to view information about the computational experiment and start the DT

According to the requirements, in the process or upon completion of the DT work, the user should be able to view the DT results. The Plotly library is used to implement this function [65]. On the page, the user is presented with interactive graphs depending on the type of equipment selected (see Fig. 14). Interactive graphs provide the ability to zoom in and select data for viewing, which helps to identify several potentially competitive options from a variety of acceptable equipment configurations. The user can save the DT results for any modification in the form of images with graphs. All of these features are provided by the Plotly library.



Figure 14. Page for viewing the DT results

Thus, a module for user interaction with the DT is realized taking into account all the declared requirements.

3.4. Development of special software

The implemented models, methods, modules and algorithms are combined into a single software package for further use as a part of special software, which is a problem-oriented system of management, decision-making and optimization based on DT models. The special software architecture is presented in Figure 15. Special software contains three main blocks: data storage, modeling, presentation of results. Data storage — accumulation of structured information to support the processes of modeling and analysis of results. Modeling — algorithms of the DT kernel. Results presentation — a module for user interaction with the DT, with the help of which it is possible to set the initial data describing DDC and parameters for modeling, as well as to view the results of the DT operation.

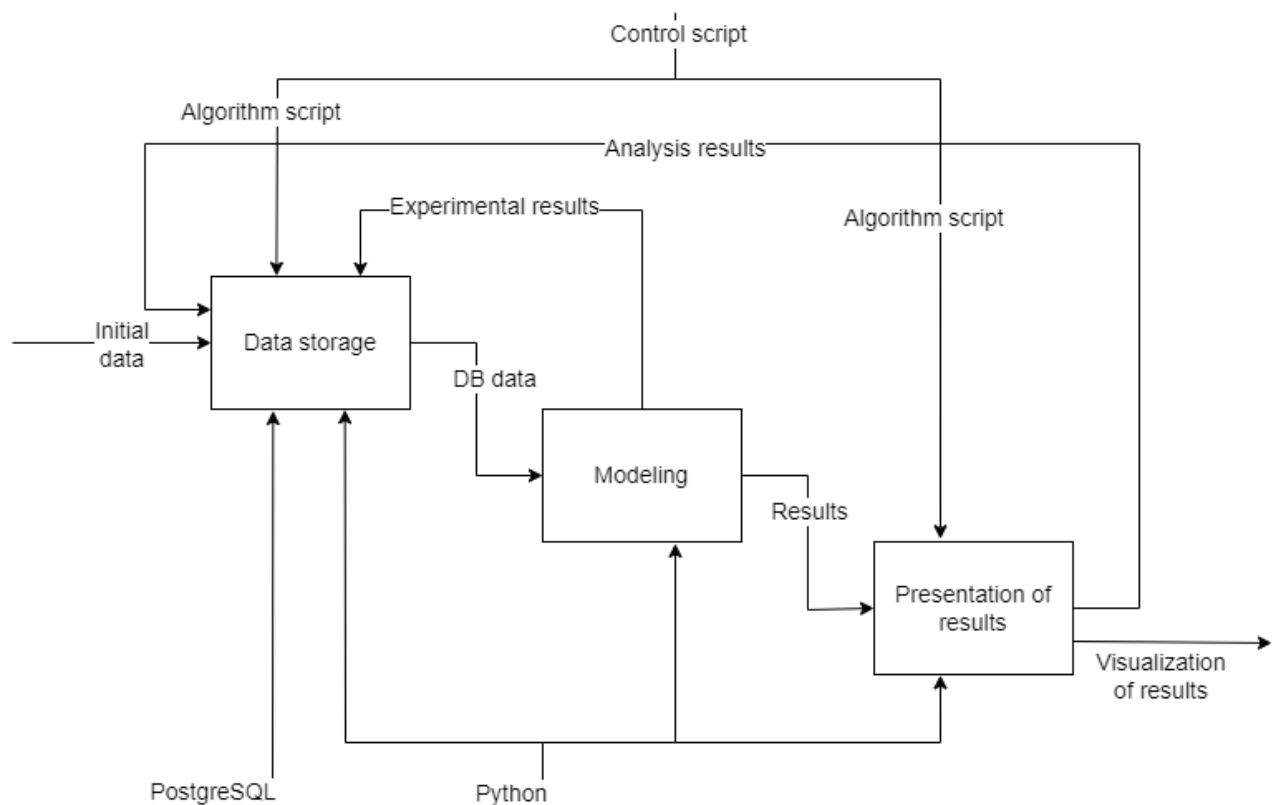


Figure 15. The architecture of the special software

To implement the previously described methods, modules and algorithms, the Python programming language was used, so Django framework was used to combine all components into a single special software as a web service. This framework supports object-relational mapping technology for accessing DB entities, and the architecture features allow the project to be run as a server-side application. A Django project consists of a project directory with general settings and directories of applications to be connected and disconnected from the project. Applications are connected in the file of general project settings.

Each page that is designed for the user interaction module with the DT is given a template that defines how the page will be displayed to the user, the view that accepts the page request, and the url

address where the page is accessible. To define application URLs, the Python URLconf (URL configuration) module is used. This module contains Python code that renders URL patterns using regular expressions, and related Python functions (views). Configurations can reference each other and be created dynamically [66]. As a result, a page has a template that defines how the page will be displayed to the user, its view that accepts a page request, and the address at which the page is accessible.

To work with the DB Django framework has a project settings file (*settings.py*), where you must specify the appropriate driver for the type of DBMS used. The same file specifies the name, host, port and data for connecting to the DB. Each entity, which is necessary for storing data of the implemented system, Django matches classes-models, described in the file *models.py*. A model contains certain fields, each of which has a name, a type, and may have additional properties corresponding to possible DB attribute properties: for example, a property of uniqueness of values or an indication that the field is a foreign key. Django provides some built-in model methods for accessing data in the DB, and also provides the ability to fetch data from the DB using manually written SQL queries.

The server part of the DDC infrastructure building process is of the greatest interest. When selecting a device from the list, a form opens for entering its parameters. This functionality is implemented with the help of AJAX-requests so that the page is not reloaded after adding parameters to the next device. Using JavaScript, data from the form is collected, and an AJAX-request is formed, which sends the received data to the view, where the data is processed and stored in the DB. Then the server replies that the data has been successfully added to the DB, or an error has occurred, which is reported to the user.

A detailed user's manual with a detailed description of all possible actions for working with the special software can be found in the Appendix 2.

Upon completion of development and implementation of the problem-oriented system of management, decision-making and optimization on the basis of DT models, the certificate of state registration of computer program No. 2023667305 "Software package for creating digital twins of distributed data acquisition, storage and processing centers" from August 14, 2023 is received (Appendix 3).

The following minimum system requirements are required for the installation and operation of the special software:

- 32-bit (x86) or 64-bit (x64) processor clocked at 2 GHz;
- 16 GB of RAM;
- 1 GB of free disk space.

Additional disk space at the rate of at least 30 MB per computational experiment will be required to store the DT results.

3.5. Conclusions to Chapter 3

Algorithms for constructing DDC digital twins are implemented according to the described requirements. Modern architectural solutions, object-oriented programming principles and tools for developing software, web applications and DBs are used.

The DT core is a program consisting of a large number of classes describing all kinds of objects that may exist in various DDCs, as well as additional elements that allow one to implement the process of modeling the DDC operation. At the same time, the principle of developing a universal software package is taken into account. The software package does not need to be changed for each simulated infrastructure.

The module for user interaction with the DT is implemented as a web service with adaptive web pages to ensure operation in interactive mode. If necessary, additional pages can be added to the service according to the developed templates. It simplifies the further development of the project.

A relational DB is implemented to store the data necessary for the creation, launch and operation of the DT. Data is accessed using libraries implemented for the Python programming language, which is used to implement all algorithms.

Special software is developed. The software is a problem-oriented management, decision-making and optimization system based on DT models. The special software is used to create DDC digital twins and allows one to compare the efficiency of the DDC operation depending on different hardware configurations. The special software includes a database, the DT core and a module for user interaction with the DT.

The special software implements methods for creating DTs of data storage and processing centers, modeling such centers, generating data flows and job flows, and visualizing. The special software can be used for a wide class of tasks in the field of design, construction and development of DDCs, including helping to select several potentially competitive options from a variety of acceptable hardware configurations.

The certificate of state registration of the computer program No. 2023667305 “Software Complex for Creating Digital Twins of Distributed Data Acquisition, Storage and Processing Centers” dated 14 August 2023 is received (Appendix 3).

The second provision to be defended is proved: “Algorithms, on the basis of which special software used to make decisions on choosing the configuration of equipment for distributed data acquisition, storage and processing centers according to specified requirements is created, are developed”.

Chapter 4. Verification and experimental operation of the special software for creating digital twins

4.1. Verification of the digital twin kernel

4.1.1 Problem statement

According to the presented requirements for digital twins of DDCs, the adequacy of the DT should be verified by the results of the existing DDC for different parameters available during the system monitoring. At the same time, the DT results should not deviate from the average value of monitoring data by more than three standard deviations. Therefore, it is necessary to verify the modeling program, which is the kernel of the DT. Further description of the material is presented in accordance with the published article [67].

Verification of the simulation program was carried out on the example of the computational infrastructure of the BM@N experiment [68] of the NICA accelerator complex [12], which is being built in Russia at the JINR in Dubna, Moscow region. The BM@N experimental facility is one of the elements of the first stage of realization of the NICA complex. After a series of technical sessions of the experiment in the winter of 2022-2023, the first physical session took place, in which more than 550 million events of interaction of a xenon ion beam on a cesium-iodine target were collected, subject to further processing and physical analysis of the experimental data [69]. The computational infrastructure of the experiment includes various resources, namely:

1. the NICA cluster, which is located in the Veksler and Baldin Laboratory of High Energy Physics (LHEP) of JINR;
2. components of the distributed grid infrastructure of the JINR Multifunctional Information and Computing Complex (MICC) [70]: the resource center of the first level Tier1 LIT and the resource center of the second level Tier2 LIT;
3. Govorun supercomputer, part of the HybriLIT heterogeneous platform [71] (JINR MICC);
4. data storage on the EOS distributed file system (JINR LIT).

The DIRAC system is used to integrate infrastructure objects and provide unified access to them for the purpose of launching mass data processing jobs [72]. Monitoring and evaluation of the performance of computing resources as a result of the experiment jobs is also performed using DIRAC Interware software [73].

Thus, as the initial data for verification we used the statistics obtained from the results of monitoring with the help of DIRAC Interware software when running the tasks of converting the obtained «raw»

(unprocessed) experimental data into digit format (hereinafter referred to as *RawToDigit* jobs) and the jobs of converting the digit format data into the data of reconstructed particle collision events in DST format (hereinafter referred to as *DigitToDst* jobs).

To achieve this goal, three stages of modeling were carried out. The task of the first stage is to model the process of acquisition experimental data, determining the amount of resources required for their storage. The task of the second and third stages is to model the process of launching *RawToDigit* and *DigitToDst* jobs, respectively, to measure such indicators as using computational resource, total job execution time, data transfer rate.

4.1.2 Description of experimental data and monitoring results

We consider the physical session of the BM@N experiment, which ran from December 2022 through February 2023. The total data collection time was approximately 720 hours. “Raw” unprocessed data (hereafter raw data) came from the facility at a rate, averaged over the entire session time, of 142 MB/s. According to monitoring results [74], at the end of the session, the total amount of physical raw data was 379 TB (see Fig. 16). During each run, during each experiment data set, the experiment data was written to raw files in the receive and store buffer. The size of an individual file was 15 GB. When ready, the raw files were copied in their entirety to the data storage on the EOS file system. The volume of the resulting experimental raw data recorded in the repository corresponds to 25 800 raw files for processing. The processing of the experimental data is a transformation and subsequent acquisition of reconstructed event data.

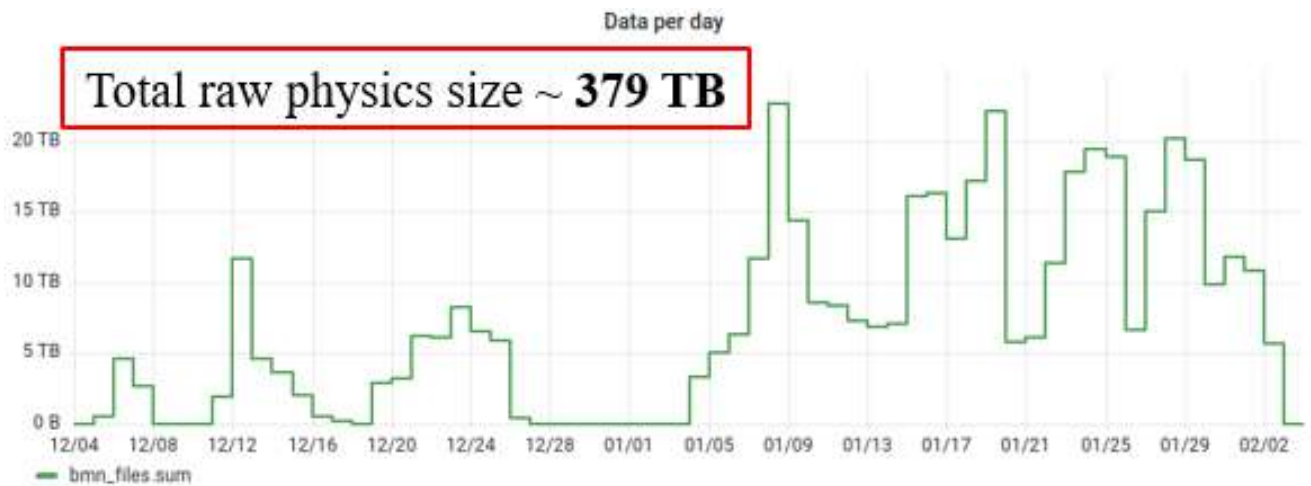


Figure 16. Real volume of incoming raw data from the BM@N experiment

Conversion of raw files into digit files (*RawToDigit* jobs) was performed on the computational resources of the experiment: NICA cluster and LIT Tier1. Each *RawToDigit* job processes 1 file of “raw” experimental data once. The digit file size averages 870 MB. All *RawToDigit* were sent for execution simultaneously. The total processing time for all raw files was approximately 36 hours. Figures 17 and 18 are graphs showing the number of jobs executed on the computational components of the NICA LHEP cluster and LIT Tier1 cluster, respectively, over the specified time period. The graph in Figure 17

shows that the computational resources of the LHEP NICA cluster are loaded uniform, with approximately 100 jobs executed every hour. This allowed us to make the assumption that 100 cores were provided on the LHEP NICA cluster. According to the graph shown in Figure 18, According to the graph shown in Figure 18, we can conclude about the uneven sing of LIT Tier1 resources. The number of executed tasks per hour varies from about 200 to 1 500, which indicates a gradual loading of the provided for processing 1 500 cores. It should be noted that a total of 4 844 jobs were processed on the LHEP NICA cluster, which is approximately 19% of the total, while 20 956 jobs (81%) were processed on the LIT Tier1 cluster. Monitoring of processor performance when executing all *RawToDigit* jobs showed that the average time to complete a single job was approximately 2 500 seconds. The total volume of the resulting digit files in the data store was 23 TB (excluding data mirroring). The graph in Figure 19 shows the data transfer rate. It can be seen that the data transfer rate from the data storage to the computational resources of the LHEP NICA cluster is 0.5 GB/s on average. Due to the fact that the jobs on the LIT Tier1 compute components were received non-uniform, similar data rates vary from 1 GB/s to 8 GB/s.



Figure 17. Number of completed jobs on the LHEP NICA cluster



Figure 18. Number of completed jobs on Tier1 LIT resources

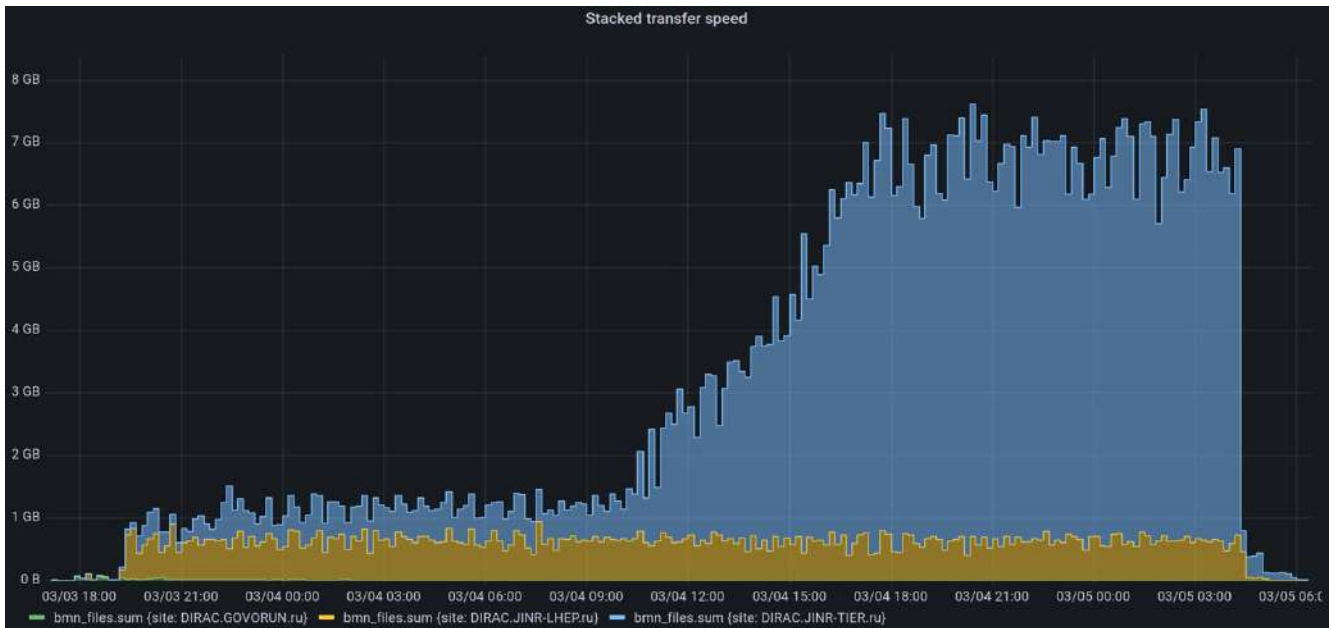


Figure 19. Data rate monitoring: yellow – from data storage to LHEP NICA cluster computing resources; blue – from data storage to Tier1 LIT resources

Obtaining reconstructed data of DST format (*DigitToDst* jobs) — conversion of digit-files into dst-files — was already carried out on a larger number of computing components, namely: LHEP NICA cluster, LIT Tier1, LIT Tier2, Govorun supercomputer. Similarly, each *DigitToDst* job processes 1 digit file once. The size of each dst file averaged 2 000 MB. All *DigitToDst* jobs were sent for execution simultaneously. The total processing time for all digit files was approximately 73 hours. Figures 20-23 show graphs reflecting the number of completed jobs on the LHEP NICA cluster, LIT Tier1, LIT Tier2, Govorun supercomputer, respectively, for the specified period of time. It can be concluded that all computational resources are loaded non-uniform. On the NICA cluster 300 cores are allocated to run *DigitToDst* jobs (see Fig. 20), and there is an interval when fewer resources are used (10 to 200 cores). The LIT Tier1 is allocated 1 500 cores, the compute component is gradually loaded to its maximum and then the number of resources used is reduced to 10 cores (see Fig. 21). A similar situation is seen at LIT Tier2, where 1 000 cores have been allocated (see Fig. 22). Govorun supercomputer, where 500 cores are allocated, is used to run jobs only in the first half of the time interval under consideration (see Fig. 23). Thus a total of 5 315 jobs were processed on the LHEP NICA cluster, which is approximately 21% of the total, 9 289 jobs (36%) on the LIT Tier1, 9 016 jobs (35%) on the LIT Tier2, and 2 180 jobs (8%) on the Govorun supercomputer. CPU performance monitoring during execution of all *RawToDigit* jobs showed that the average execution time of one job is about 10 000 seconds. The total volume of the resulting dst files in the data storage is 53 TB.



Figure 20. Number of completed jobs on the LHEP NICA cluster



Figure 21. Number of completed jobs on the LIT Tier1



Figure 22. Number of completed jobs on the LIT Tier2



Figure 23. Number of completed jobs on the Govorun supercomputer

4.1.3 Modeling the process of acquisition and storage of BM@N experimental data

The simulated system of BM@N experiment data acquisition and storage is presented in Figure 24. The inputs to run the simulation program are the rate of data generation and the bandwidth of communication links between infrastructure objects. The number of resources required to store all incoming data on the buffer and in the EOS data store must be determined.

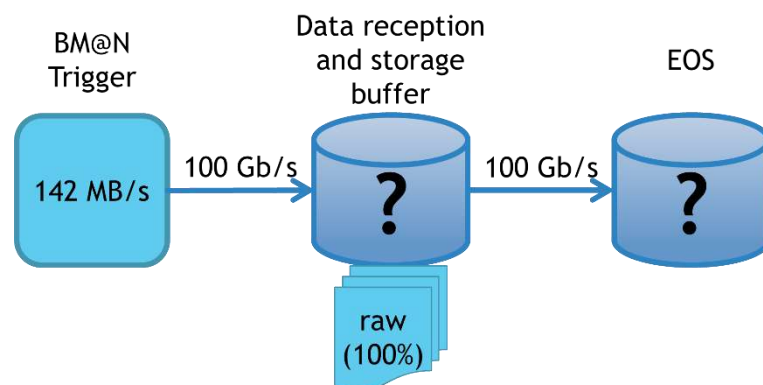


Figure 24. Modeled system for acquisition and storing data of the BM@N experiment

The simulation results showed that the amount of data accumulated over 720 hours would be approximately 363 TB (see Fig. 25). All data packed in raw files will be transferred to the EOS repository (see Fig. 26). A 400 TB buffer will be sufficient for receiving and storing experimental data at a given average generation frequency (not including data mirroring).

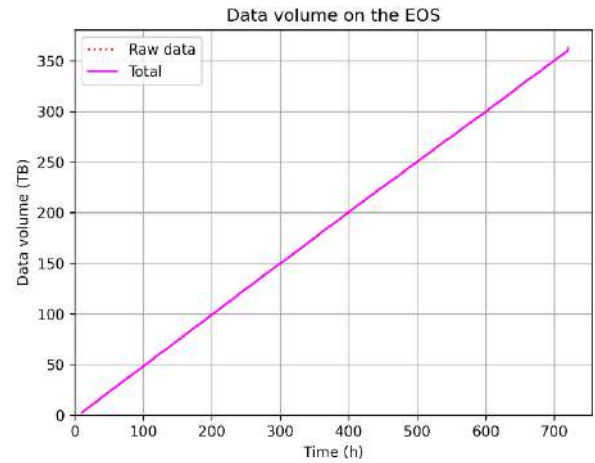
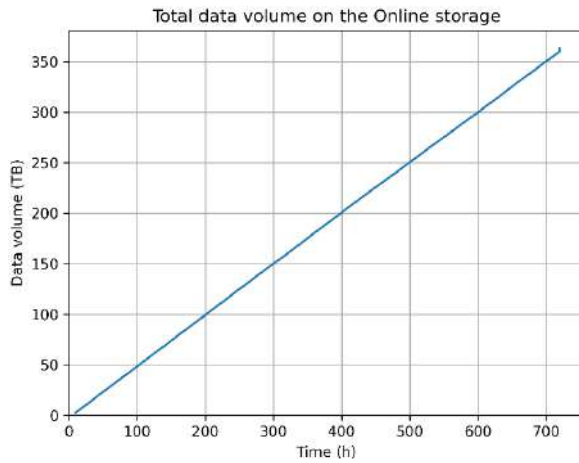


Figure 25. Amount of data reception and storing buffer Figure 26. Volume of raw files in the EOS data storage

4.1.4 Modeling of the process of executing jobs of conversion of experimental data

Based on the monitoring results, the system of experimental data processing was modeled, presented in Figure 27. During the simulation, it is required to calculate the using computational resource during *RawToDigit* jobs and the data transfer rate under the following conditions. There are 100 cores allocated to the LHEP NICA cluster, and 1 500 cores allocated to the LIT Tier1. The number of jobs executed per hour on a LIT Tier1 varies from about 200 to 1 500, so when modeling the *RawToDigit* job execution process, parameters should be set to vary the number of cores used. Such parameters are the probability of occurrence of the event of change in the number of free cores available for job execution, as well as the range of acceptable values. In this case, the probability of a resource increase event occurring is 0.005, with the number of cores varying from 200 to 1 500, on average by 100 units at each event. The following are used for modeling: uniform distribution of job execution time with average value equal to 2 500 seconds, average raw file size equal to 15 GB, average digit file size — 870 MB. Additionally, the total execution time of all *RawToDigit* jobs must be determined.

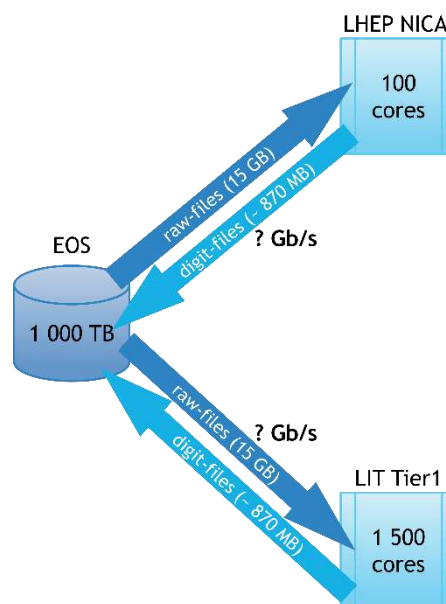


Figure 27. A simulated BM@N experiment computational infrastructure for executing experimental data conversion jobs

All 25 800 *RawToDigit* jobs are generated simultaneously. The process of executing jobs during simulation is controlled by pilots. A pilot is an algorithm that prepares a job for executing, including analyzing the number of free cores on a computational component. If free resources are available, the pilot takes the next job from the queue and sends it for execution. Then the job occupies a processor core and starts executing, i.e. processing the input raw file from the EOS data store. As a result of job execution, the output digit-file is also written to EOS.

Let's consider the obtained results of modeling. Figure 28 presents graphs showing the total number of completed jobs on the LHEP NICA cluster and LIT Tier1, respectively, at each point in time. It can be concluded that all *RawToDigit* jobs are completed in about 30 hours, with 3 875 jobs processed on the LHEP NICA cluster, which is approximately 15% of the total, and 21 924 jobs (85%) processed on the LIT Tier1.

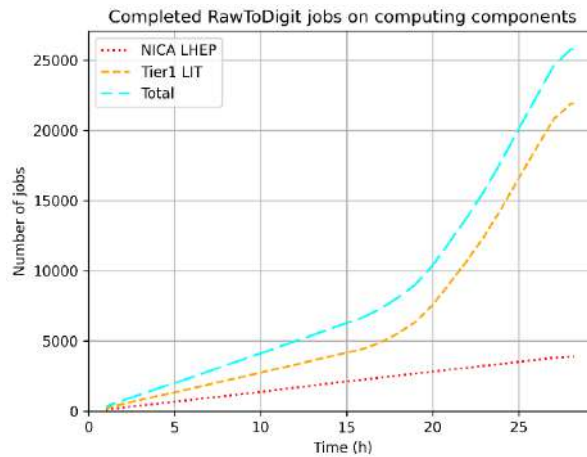


Figure 28. Number of *RawToDigit* jobs completed on computational resources during modeling

Figures 29 and 30 are graphs showing the number of cores in use on the computational components of the LHEP NICA cluster and LIT Tier1, respectively, at each point in time. We conclude that the resources of the LHEP NICA cluster are uniform loaded at 100%, but the LIT Tier1 resources are not fully used, i.e. they are initially loaded at 15% and then gradually begin to fill up to 100%.

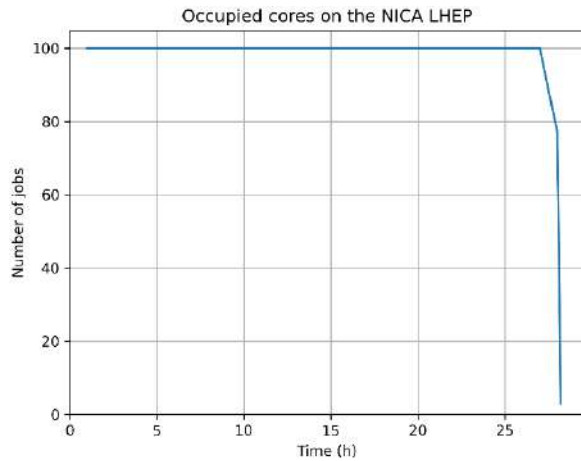


Figure 29. Using of the computational component resource of the LHEP NICA cluster

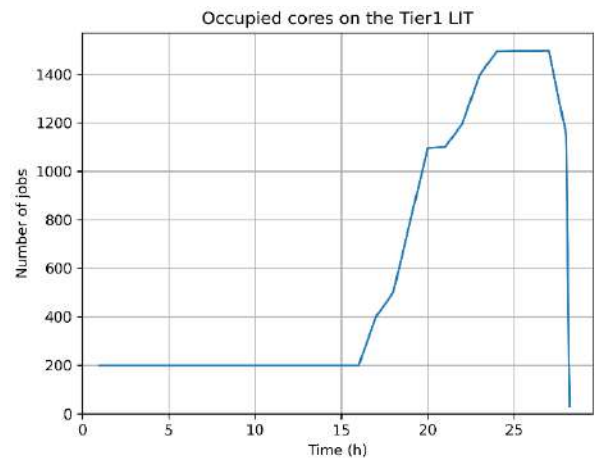


Figure 30. Using of the computational component resource of the LIT Tier1

Graphs showing the load on communication links between the storage and computational components of the LHEP NICA cluster and LIT Tier1 are presented in Figures 31 and 32, respectively. It can be concluded that the average data rate between EOS and the LHEP NICA cluster is $5 \text{ Gb/s} = 0.63 \text{ GB/s}$; between EOS and LIT Tier1, the rate varies from $8 \text{ Gb/s} = 1 \text{ GB/s}$ to $64 \text{ Gb/s} = 8 \text{ GB/s}$. Thus, the change in data rate between EOS and LIT Tier1 corresponds to the change in the number of cores free to run jobs.

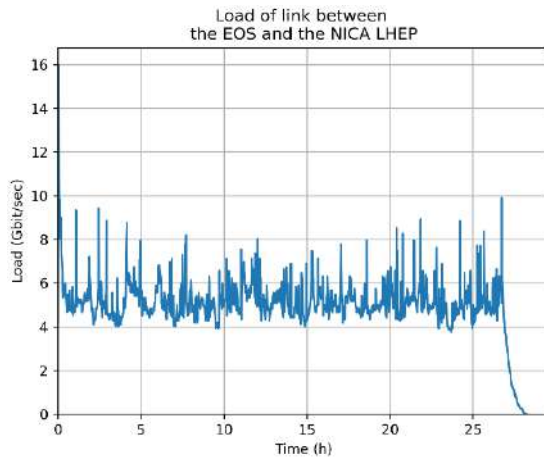


Figure 31. Loading of the communication link between EOS and the LHEP NICA cluster

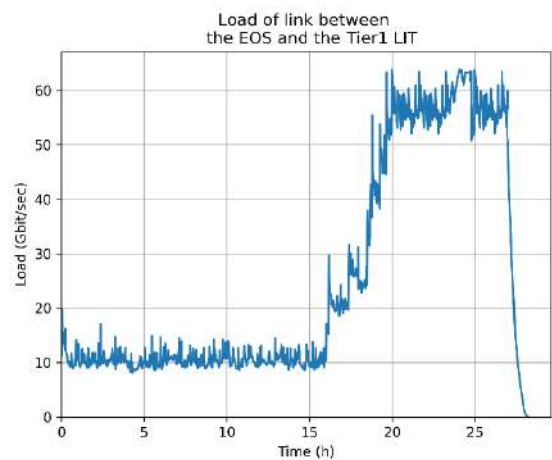


Figure 32. Loading of the communication link between EOS and the LIT Tier1

It is worth noting that the digit file size in the simulation is a uniformly distributed random variable with a mean of 870 MB (see Fig. 33), and the total size of all digit files was approximately 22 TB (see Fig. 34).

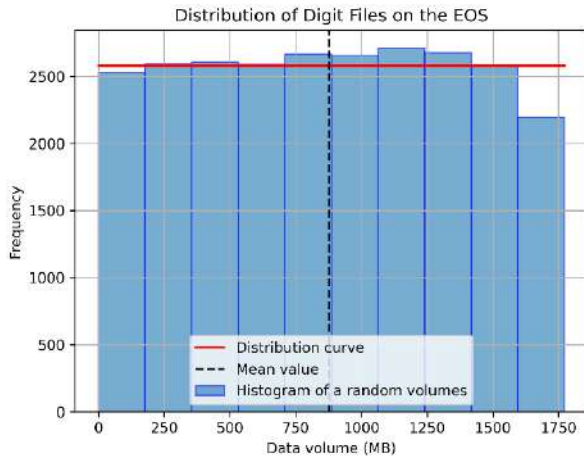


Figure 33. Uniform distribution of digit file sizes in EOS storage

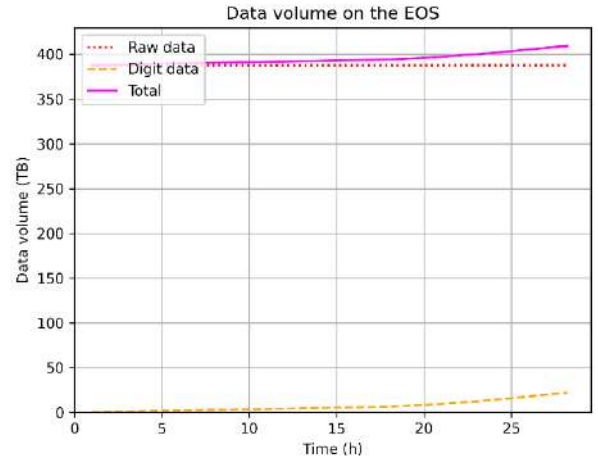


Figure 34. Data volume in the EOS repository after conversion of experimental data

4.1.5 Modeling the process of conversion event reconstruction jobs

The infrastructure that was used to reconstruct particle collision events from the acquired digit data is shown in Figure 35. During modeling it is required to calculate the load of computing resources in the process of *DigitToDst* jobs execution under the following conditions. There are 300 cores allocated to the LHEP NICA cluster, 1 500 cores allocated to the LIT Tier1, 1 000 cores allocated to the LIT Tier2, and 500 cores allocated to the Govorun supercomputer. In this case, the LHEP NICA cluster is fully loaded 60% of the time, with 10 to 200 cores in use the rest of the time. LIT Tier1 is gradually loaded to maximum and then the remaining 40% of the time the number of resources used is reduced to 10 cores. A similar situation is seen on the LIT Tier2. The Govorun supercomputer is used to run jobs half of the time interval under consideration. Thus, when modeling the process of *DigitToDst* job execution, parameters for changing the number of cores used (probabilities and range of changes) for all computational resources are established. In this case, the probability of occurrence of resource increase and decrease events is 0.001, with the number of cores on the LHEP NICA cluster varying from 10 to 300, on the LIT Tier1 from 10 to 1 500, on the LIT Tier2 from 10 to 1 000, and on the Govorun supercomputer from 50 to 500. The following are used for modeling: uniform distribution of job execution time with average value equal to 10 000 seconds, average digit-file size equal to 870 MB, average dst-file size — 2 000 MB. Additionally, the total execution time of all *DigitToDst* jobs must be determined. All 25 800 *DigitToDst* jobs are generated simultaneously. The process of executing *DigitToDst* jobs during simulation is similar to the process of executing *RawToDigit* jobs.

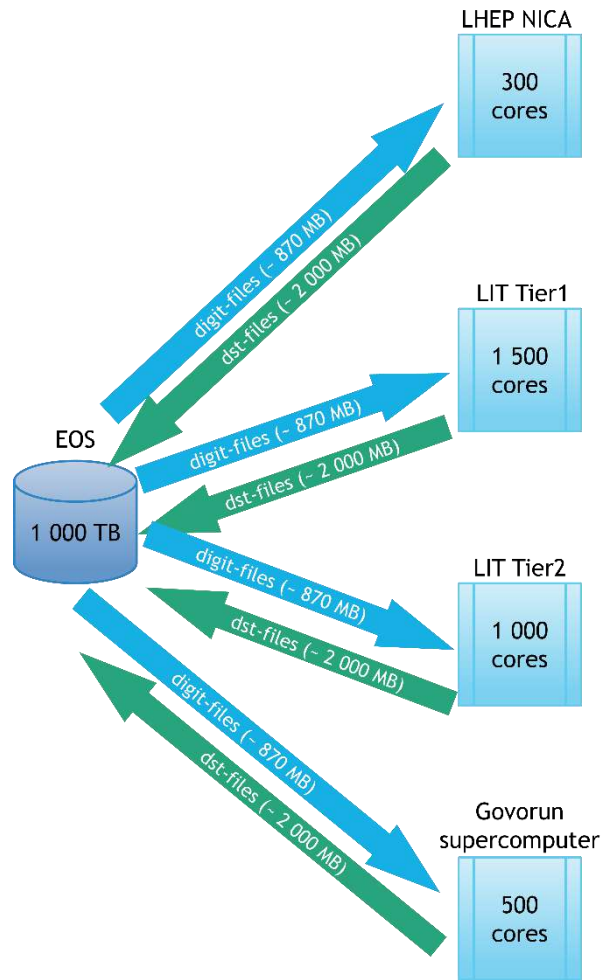


Figure 35. A simulated computational infrastructure of the BM@N experiment for executing event reconstruction jobs

Let's consider the obtained results of modeling. Figure 36 presents graphs showing the total number of completed jobs on the LHEP NICA cluster, LIT Tier1, LIT Tier2, Govorun supercomputer at each point in time. We conclude that all *DigitToDst* jobs are completed in about 80 hours. In this case, 5 906 jobs were processed on the LHEP NICA cluster, which is approximately 24% of the total, 8 872 jobs (34%) on the LIT Tier1, 8 598 jobs (33%) on the LIT Tier2, 2 424 jobs (9%) on the super-computer.

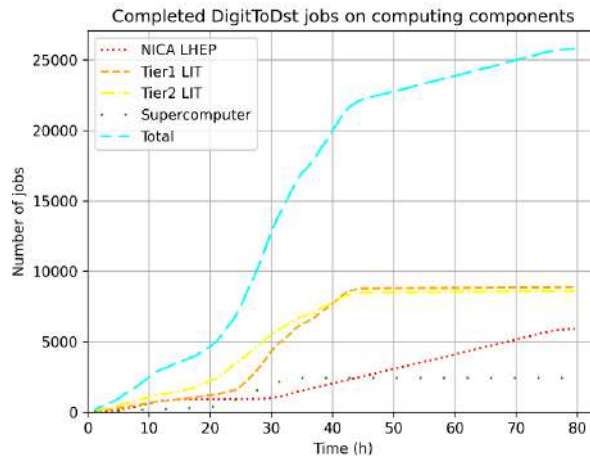


Figure 36. Number of *DigitToDst* jobs completed on computational resources during modeling

Figures 37-40 are graphs showing the number of used cores on the computational components of the LHEP NICA cluster, LIT Tier1, LIT Tier2, and Supercomputer, respectively, at each point in time. We conclude that the resources of the LHEP NICA cluster are uniformly fully used for 60% of the time, the rest of the time 10 to 200 cores are used. LIT Tier1 and LIT Tier2 resources are gradually loaded to maximum, and 40% of the time an average of 10 cores are used. The Govorurun supercomputer runs jobs only in the first half of the time interval under consideration, with 10% of resources being used at first, and then used up to 100% is traced.

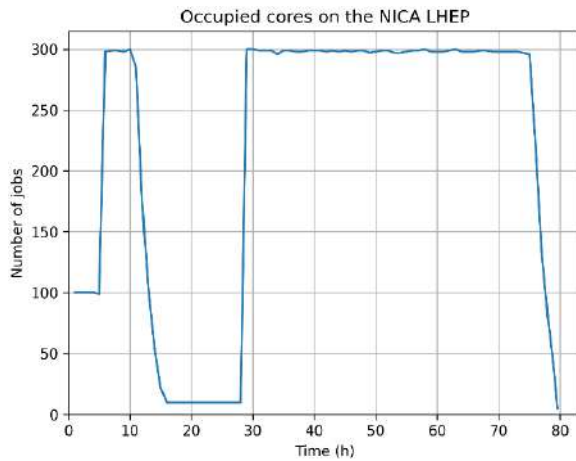


Figure 37. Using resource of the LHEP NICA cluster computational components

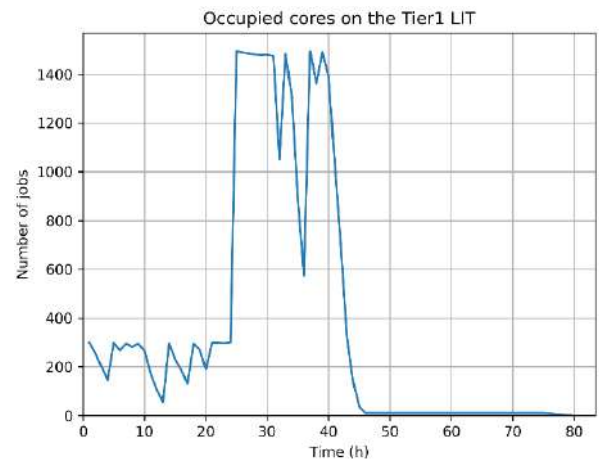


Figure 38. Using resource of the LIT Tier1 computational components

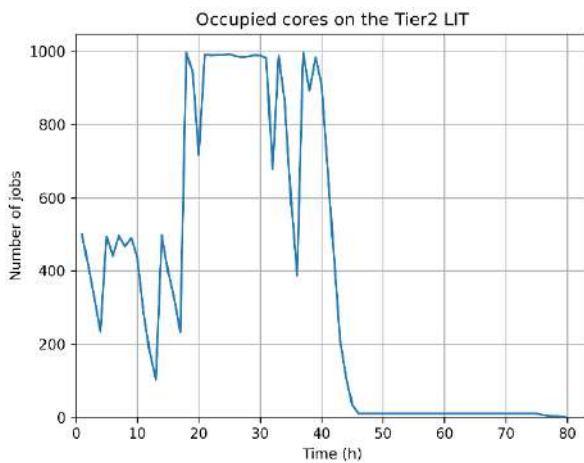


Figure 39. Using resource of the LIT Tier2 computational components

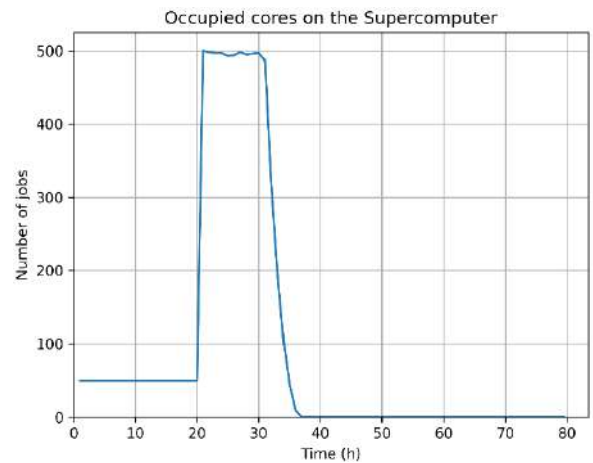


Figure 40. Using resource of the Govorun supercomputer computational components

It is worth noting that the dst-file size in the simulation is a uniformly distributed random variable with a mean of 2 000 MB (see Fig. 41), and the total size of all dst-files was approximately 52 TB (see Fig. 42).

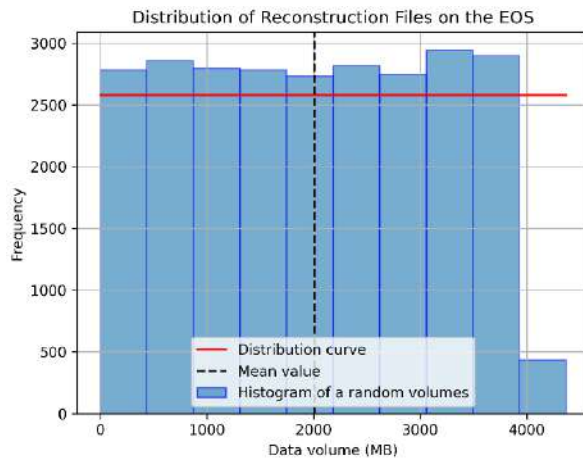


Figure 41. Uniform distribution of digit file sizes in EOS storage

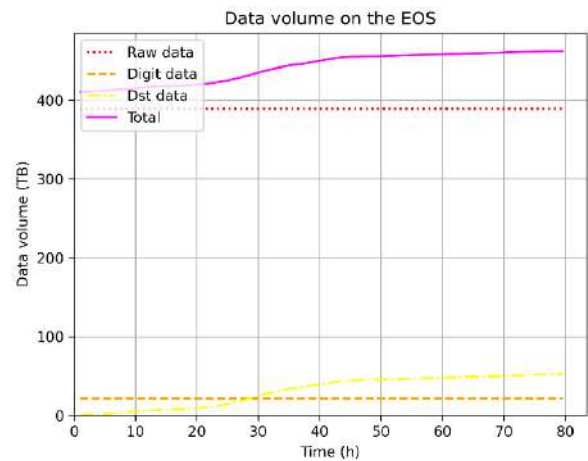


Figure 42. The amount of data in the EOS repository after receiving reconstruction data

4.1.6 Conclusions on verification results

Verification of the simulation program, which is the kernel of the special software for the creation of digital twins of DDCs, was carried out on the example of the computing infrastructure of the BM@N experiment of the NICA project. The considered computing infrastructure was used to acquire, store and process the data of the experiment session, which took place from December 2022 to February 2023. The statistics obtained from the results of monitoring the computing infrastructure of the experiment using the DIRAC Interware software was used as input data for modeling.

In the process of modeling, such indicators as the amount of resources required to store incoming data, the load of computational resources during the transformation of experimental data and obtaining reconstructed event data from them, the total time of job execution, and the speed of data transfer between infrastructure objects were determined. Table 1 shows the results of monitoring and modeling for a more convenient comparison.

The verification results proved the correct operation of the modeling program. Adequacy was assessed by several indicators. It should be noted that deviations of modeling results from the average value obtained by monitoring results **do not exceed three standard deviations** of statistical monitoring data. This value of accuracy is sufficient for further use of the simulation program as part of special software to create digital twins of DDCs, which will be used to solve the problems of design and development of computational infrastructure of scientific experiments of the “megasciences” class.

The simulation program has been previously applied to other BM@N computing infrastructure configurations as well. Some results of the application have been published [75, 76, 77].

Table 1. Comparison of monitoring and modeling results

	Monitor- ing	Modeling	Standard deviation (σ)
Volume of experimental data accumulated during 720 hours of facility operation (TB)	379	363	10
Total execution time of all <i>RawToDigit</i> jobs (h.)	36	30	5
Number of completed jobs on the LHEP NICA cluster / of total number of jobs	4 844 / 19%	3 875 / 15%	500
Number of completed jobs on the LIT Tier1 / % of total number of jobs	20 956 / 81%	21 924 / 85%	500
Data transfer rate between EOS and the LHEP NICA cluster (GB/s)	0.5	0.63	0,1
Data transfer rate between EOS and the LIT Tier1 (GB/s)	1 to 8	1 to 8	1
Total digit file volume (TB)	23	22	2
Общее время выполнения всех <i>DigitToDst</i> задач (ч.)	73	80	5
Number of completed jobs on the LHEP NICA cluster / % от общего числа задач	5 315 / 21%	5 906 / 24%	500
Number of completed jobs on the LIT Tier1 / % of total number of jobs	9 289 / 36%	8 872 / 34%	500
Number of completed jobs on the LIT Tier2 / % of total number of jobs	9 016 / 35%	8 598 / 33%	500
Number of completed jobs on the supercomputer / % of total number of jobs	2 180 / 8%	2 424 / 9%	500
Total dst file volume (TB)	53	52	2

4.2. Application of the special software to create the digital twin of the computing infrastructure of the BM@N experiment of the NICA complex

The results of the first physical session of the BM@N experiment of the NICA complex, which took place from December 2022 to February 2023, showed that the computing infrastructure of the experiment needs to be modified. In this regard, it is required to create a DT with several types of architecture of the system of data acquisition, storage and processing of the experiment, as well as with different parameters of equipment.

The main goal of the DT creation is to select the equipment configuration that will ensure data storage and processing taking into account the planned parameters of data flows of future BM@N experiment sessions. The most preferable selection criterion is the time of processing all data, which should be the minimum of all possible alternatives.

The distributed system of BM@N experiment data acquisition, storage and processing includes the following components:

1. data generator (Trigger) — average data generation rate 140 MB/sec;
2. intermediate data storage (Buffer) — storing “raw” unprocessed experimental data and recording them in raw files, the average volume of which is 15 GB;
3. permanent data storage on a distributed file system (EOS) — 1 000 TB to store all files, which contain not only experimental data, but also data of modeled events, as well as the results of each stage of processing of the listed data types;
4. computational cluster for processing experimental data during the operation of the facility (Online farm) — 1 000 cores can be used to process the data after the session is over;
5. JINR LHEP computing cluster (NICA LHEP) — 1 000 cores can be used for data processing;
6. the resource center of the first level of the JINR LIT MICC (Tier1 LIT) — 750 cores can be used for data processing;
7. the resource center of the second level of the JINR LIT MICC (Tier2 LIT) — 500 cores can be used for data processing;
8. Govorun supercomputer JINR LIT (Govorun) — 200 cores can be used for data processing.

The process of building the DDC infrastructure is presented in Figure 43.

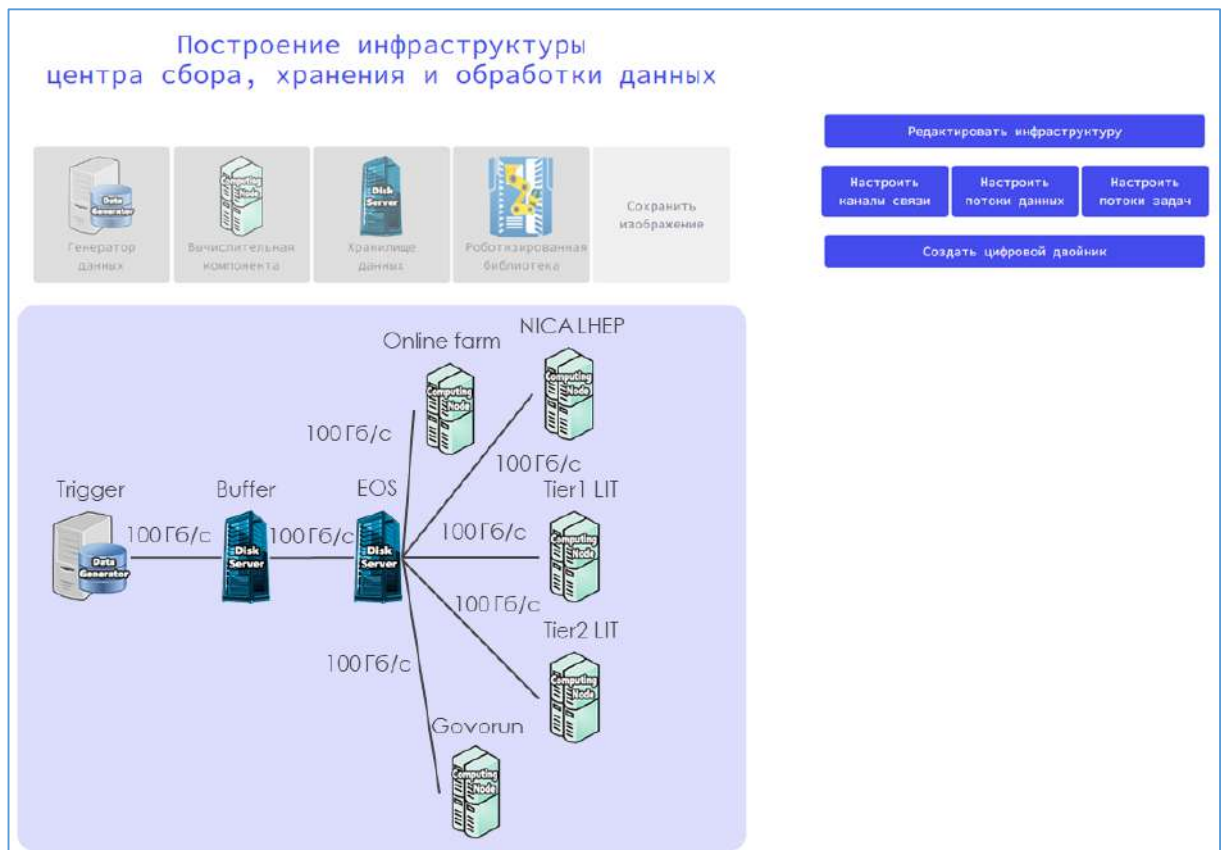


Figure 43. Distributed system for data acquisition, storage and processing of BM@N experiment

It should be noted that the bandwidth of communication links is 100 Gbit/s. We consider the case of DDC operation at uninterrupted functioning of the equipment after the end of the experiment session, i.e. computing jobs were not started during the data set period.

The process of experimental data processing includes two steps:

1. conversion of data from raw files (25 800 pcs.) в формат digit, into digit format, average size of the resulting digit file is 870 MB — *RawToDigit* jobs, average execution time of one job is 2 500 s;
2. conversion of digit format data into reconstructed particle collision event data in DST format, average size of the resulting dst file is 2 000 MB — *DigitToDst* jobs, average execution time of one job is 86 400 s.

The process of processing the generated model data includes two steps:

1. obtaining from the generated data in generation files, or gen files (60 000 pcs.), the average size of which is 4 MB, modeled events after collisions, the average size of the resulting sim file is 300 MB — *GenToSim* jobs, average execution time of one job is 5 400 s;
2. conversion of sim format data into reconstructed particle collision event data of DST format, average size of the resulting dst file is 300 MB — *SimToDst* jobs, average execution time of one job is 5 400 s.

The characteristics specified in the enumeration of data processing stages are used to configure the parameters of data flows and job flows. To analyze the workload of storage resources and computational

components when processing only experimental data with different number of free cores, as well as when processing experimental and modeled data in parallel, we added the corresponding computational experiments on DT.

In the first computational experiment, the required amount of data in the EOS storage is calculated on the DT and the workload of computational resources is analyzed in the process of raw data processing, i.e. during the execution of *RawToDigit* and *DigitToDst* jobs. In this case, *RawToDigit* jobs are executed only on LIT Tier1 resources, while *DigitToDst* jobs are executed on all available computational components. Two configurations of computing equipment are considered:

- a. Online farm — 0 cores, NICA LHEP — 250 cores, Tier1 LIT — 750 cores, Tier2 LIT — 500 cores, Govorun — 200 cores;
- b. Online farm — 1 000 cores, NICA LHEP — 1 000 cores, Tier1 LIT — 750 cores, Tier2 LIT — 500 cores, Govorun — 200 cores.

The second computational experiment on DT is devoted to search for the required amount of data in EOS storage and analyze the workload of computational resources. But in this case both the process of experimental data processing (execution of *RawToDigit* and *DigitToDst* jobs), and the process of model data processing (execution of *GenToSim* и *SimToDst* jobs) are considered. Similar to the first computational experiment, *RawToDigit* jobs are executed only on Tier1 LIT resources, while *DigitToDst* jobs are executed on all available computational components. *GenToSim* and *SimToDst* jobs are executed only on Online farm and NICA LHEP resources. In this case, only the basic configuration of equipment is used, which is set at the stage of building infrastructure before the creation of the DT (Online farm — 1 000 cores, NICA LHEP — 1 000 cores, Tier1 LIT — 750 cores, Tier2 LIT — 500 cores, Govorun — 200 cores).

The DTs for both computational experiments are run simultaneously. After 20 hours, all results are obtained. Let us consider the results in more detail on the images that are exported from the web service.

The results of the first computational experiment show that with the hardware configuration shown in Figure 44, all experimental data will be fully processed in about 432 hours (see Fig. 45), which is 18 days. At the same time, the *RawToDigit* jobs will be completed in 60 hours (see Fig. 46). The graph of completed *DigitToDst* jobs is shown in Figure 47. The graphs in Figures 48-51 represent the full using of computing components. The amount of data in the EOS storage at the end of experimental data processing is 460 TB (see Fig. 52).

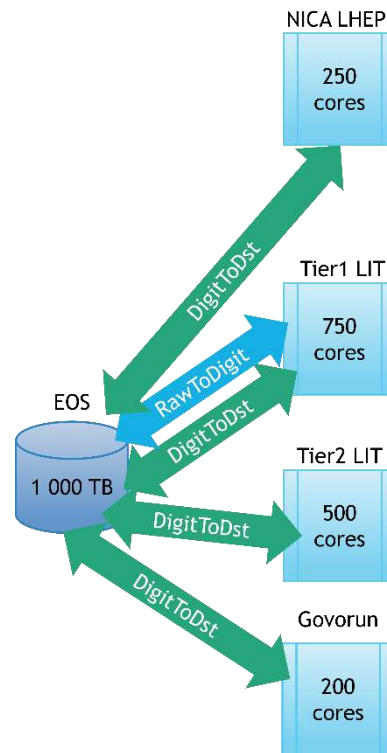


Figure 44. Configuration diagram (a) of the first computational experiment on the DT

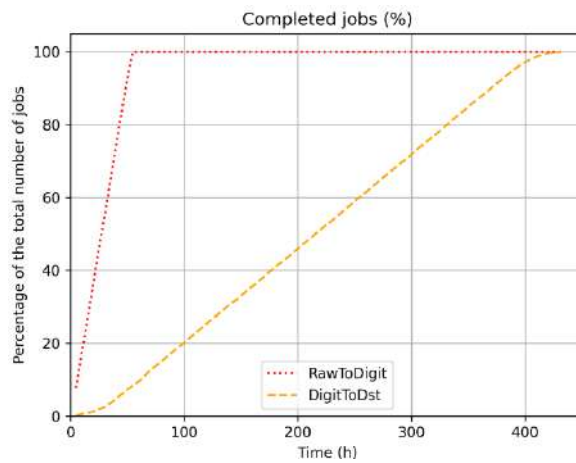


Figure 45. Number of experimental data processing jobs completed on computational resources

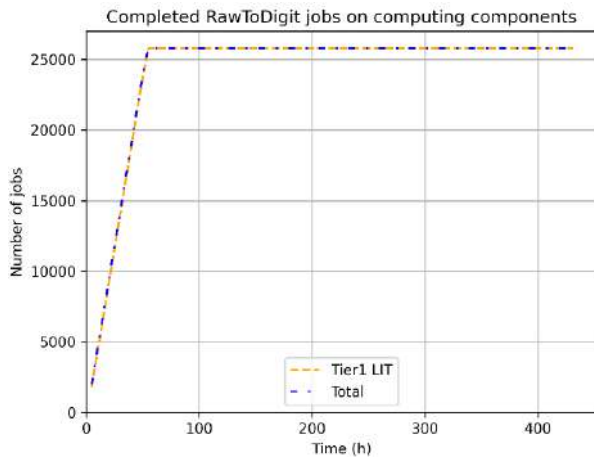


Figure 46. Number of *RawToDigit* jobs completed on computational resources

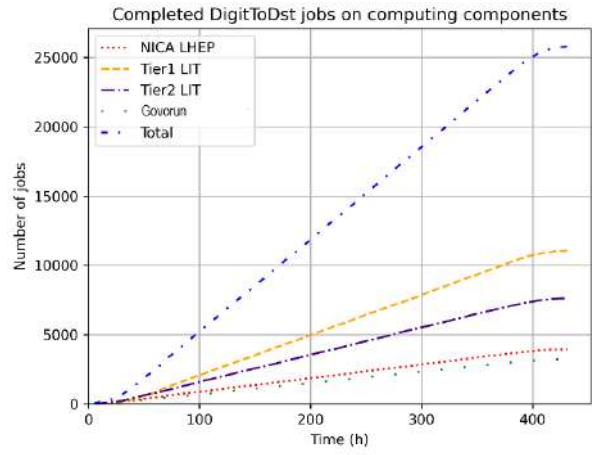


Figure 47. Number of *DigitToDst* jobs completed on computational resources

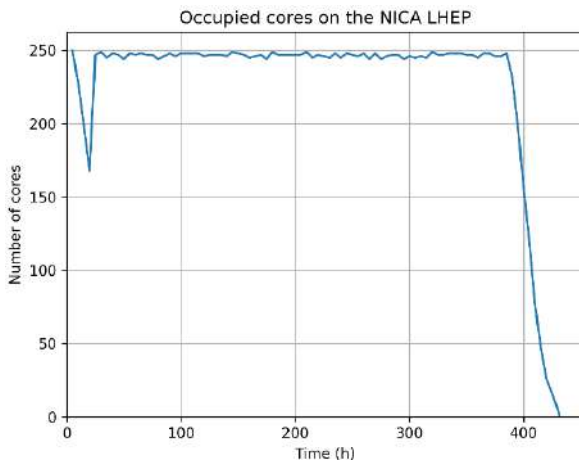


Figure 48. Usage of resources of the NICA LHEP computing component

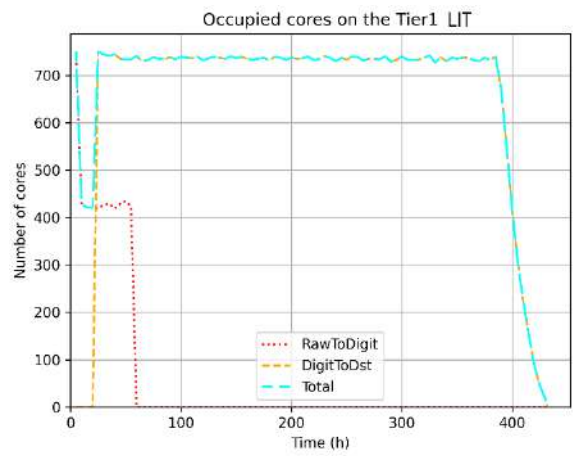


Figure 49. Usage of resources of the Tier1 LIT computing component

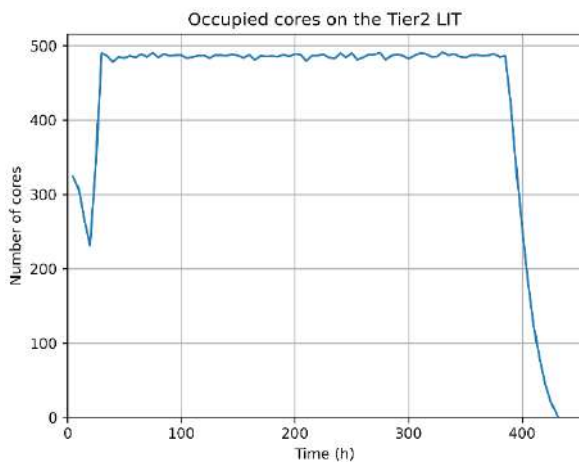


Figure 50. Usage of resources of the Tier2 LIT computing component

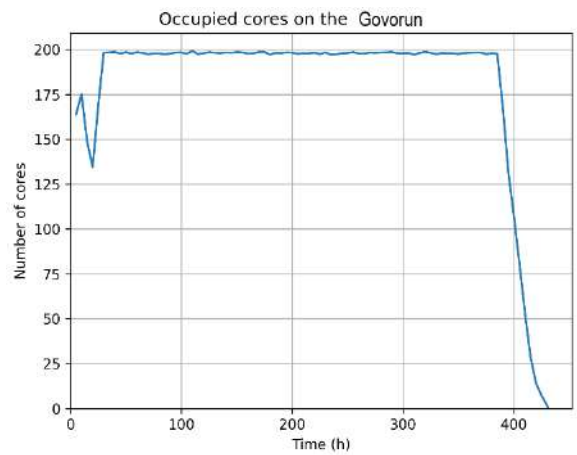


Figure 51. Usage of resources of the Govorun computing component

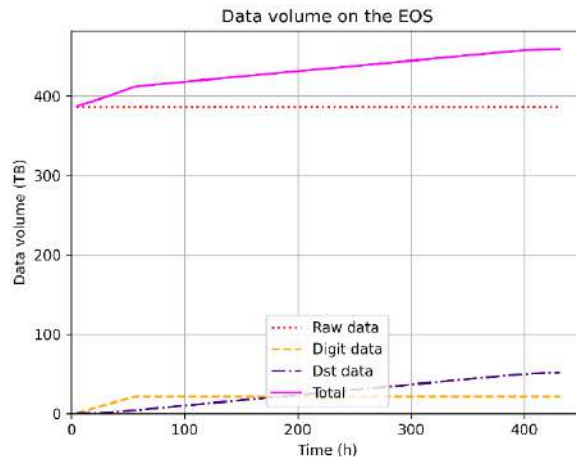


Figure 52. Data volume in the EOS repository after experimental data processing

The results of the DT with the updated hardware configuration (see Fig. 53) showed that all experimental data will be fully processed in about 240 hours (see Fig. 54), which is 10 days. At the same time, the *RawToDigit* will be completed in 28 hours (see Fig. 55). The graph of completed *DigitToDst* jobs is shown in Figure 56. The graphs in Figures 57-61 represent the full using of computing components. It should be noted that this configuration allows processing experimental data almost 2 times faster. The increase in the number of computing resources does not affect the accumulated amount of data in EOS storage.

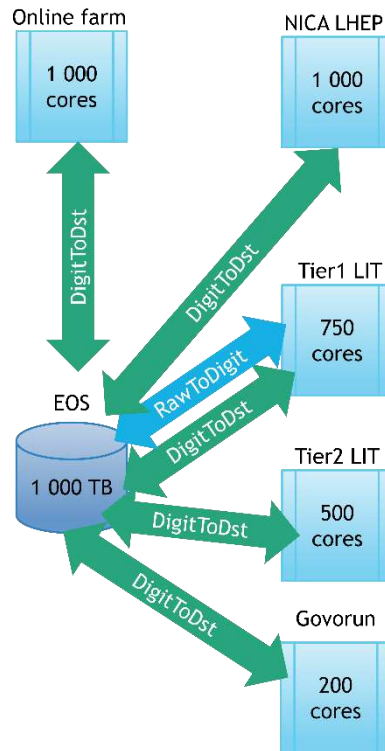


Figure 53. Configuration diagram (b) of the first computational experiment on the DT

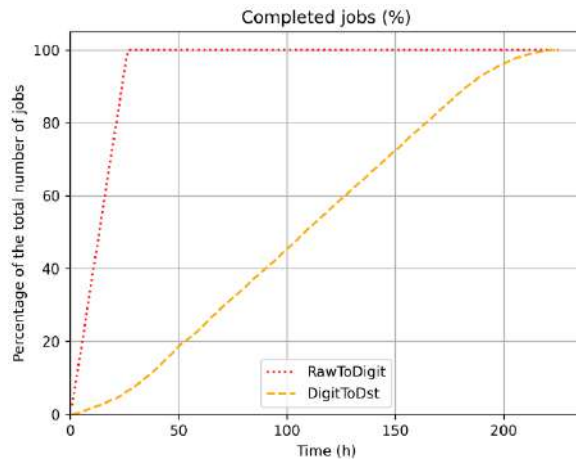


Figure 54. Number of experimental data processing jobs completed on computational resources

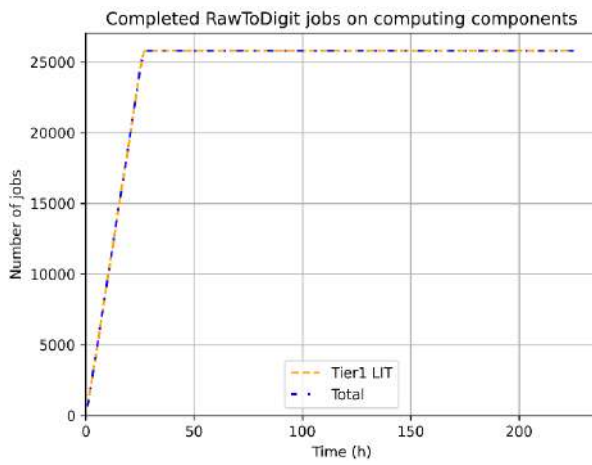


Figure 55. Number of *RawToDigit* jobs completed on computational resources

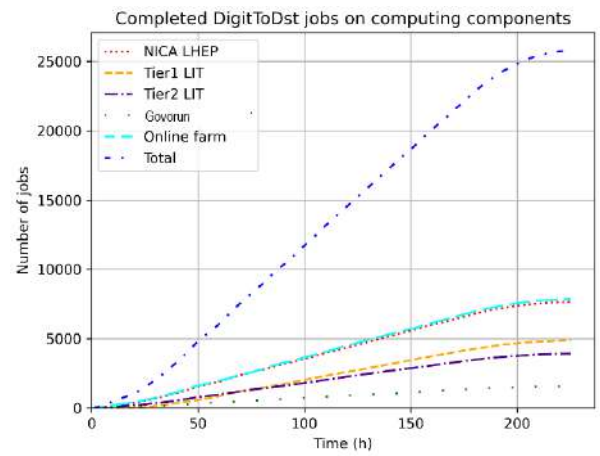


Figure 56. Number of *DigitToDst* jobs completed on computational resources

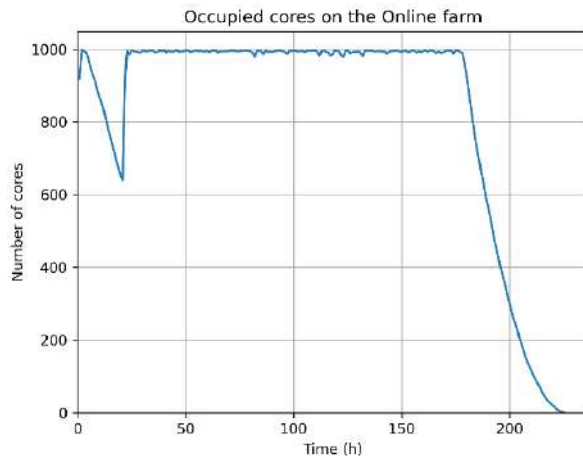


Figure 57. Usage of resources of the Online farm computing component

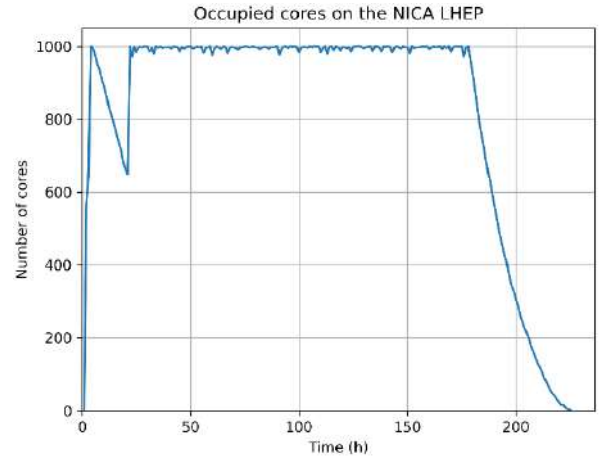


Figure 58. Usage of resources of the NICA LHEP computing component

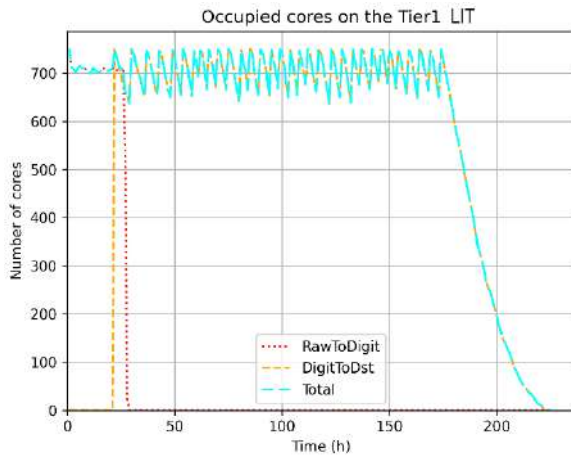


Figure 59. Usage of resources of the Tier1 LIT computing component

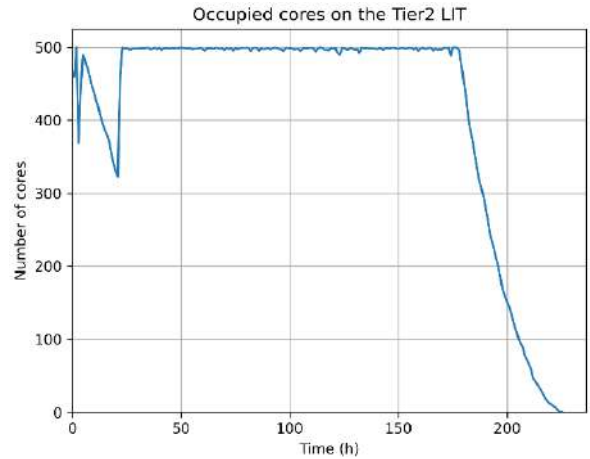


Figure 60. Usage of resources of the Tier2 LIT computing component

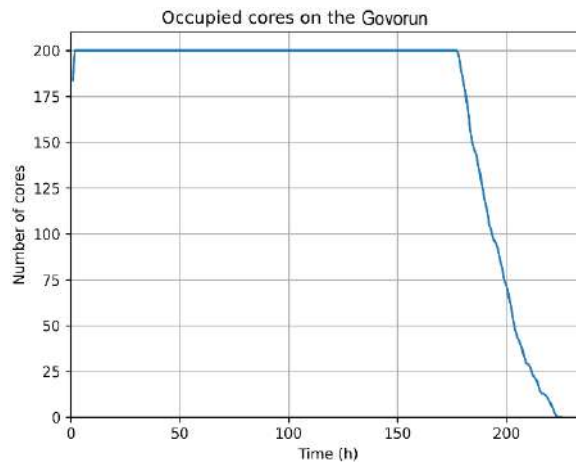


Figure 61. Usage of resources of the Govorun computing component

The results of the second computational experiment on DT to process all types of data on the computing infrastructure shown in Figure 62 showed that the experimental data would be fully processed in approximately 367 hours (15 days) and the modeled data in 130 hours (6 days) (see Fig. 63). The process of executing *RawToDigit* jobs is similar to the same process in the first computational experiment with configuration (b). The execution graphs of *DigitToDst*, *GenToSim* and *SimToDst* jobs are shown in Figures 64-66, respectively. The graphs in Figures 67-71 show the full usage of the computational components. It is worth noting that although the experimental data takes slightly longer to process compared to configuration (b) of the first computational experiment, it takes only 6 days to process the modeled data. The amount of data in the EOS storage at the end of the processing of experimental and modeled data is 500 TB (see Fig. 72).

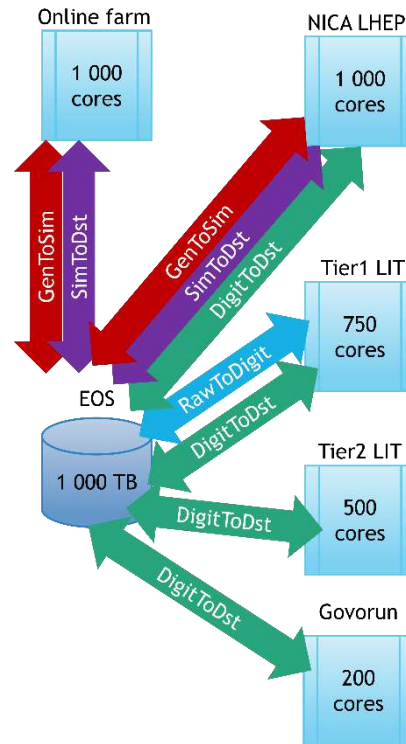


Figure 62. Configuration diagram of the second computational experiment on the DT

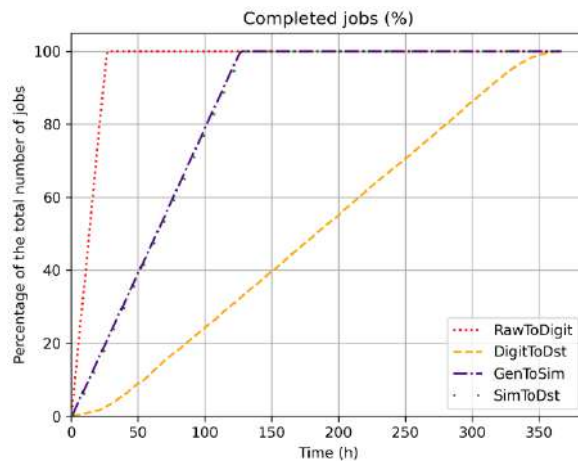


Figure 63. Number of experimental data and modeled data processing jobs completed on computational resources

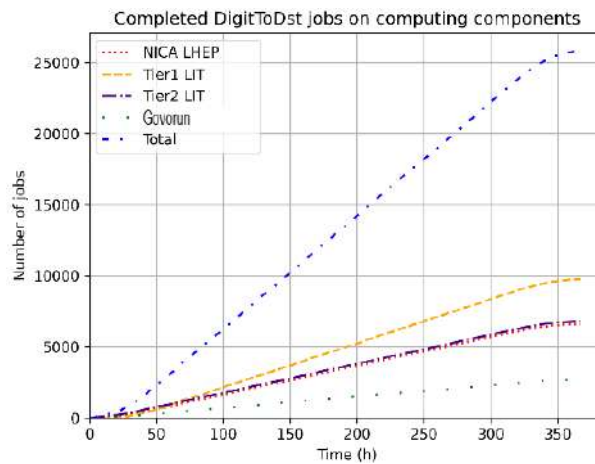


Figure 64. Number of *DigitToDst* jobs completed on computational resources

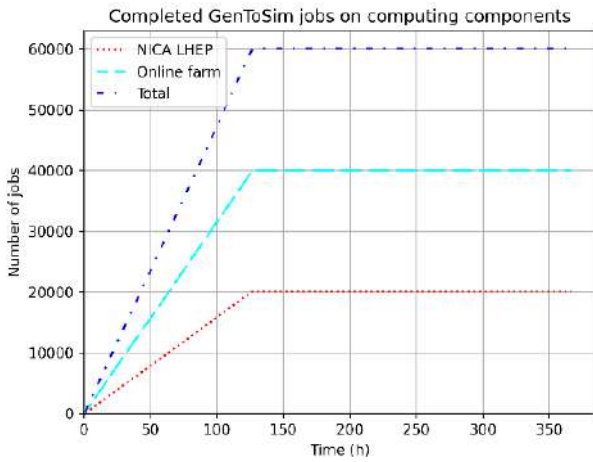


Figure 65. Number of *GenToSim* jobs completed on computational resources

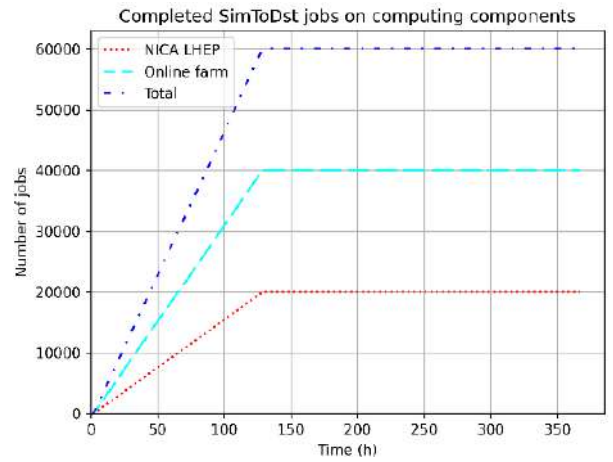


Figure 66. Number of *SimToDst* jobs completed on computational resources

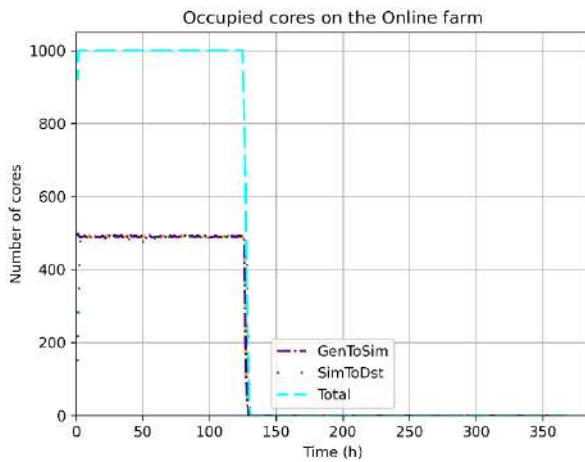


Figure 67. Usage of resources of the Online farm computing component

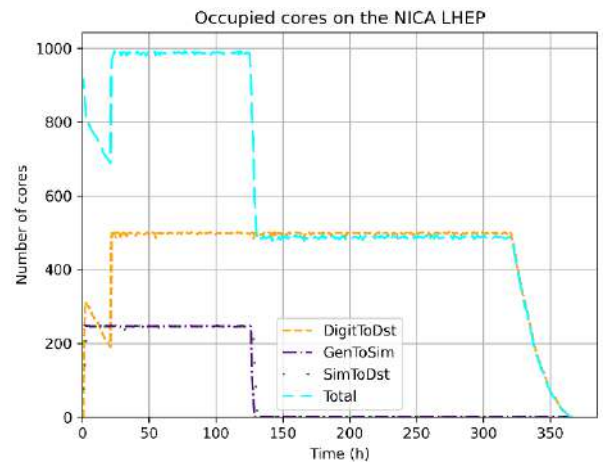


Figure 68. Usage of resources of the NICA LHEP computing component

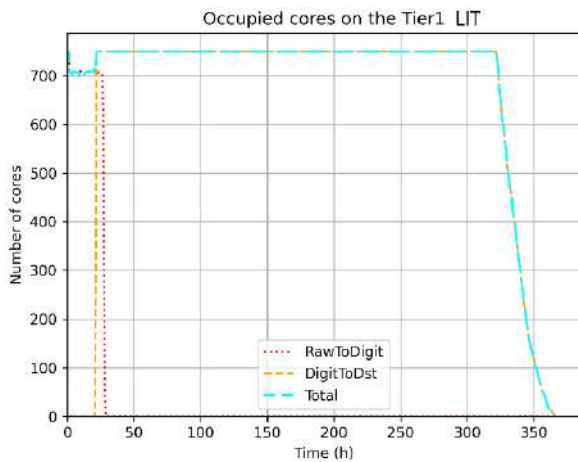


Figure 69. Usage of resources of the Tier1 LIT computing component

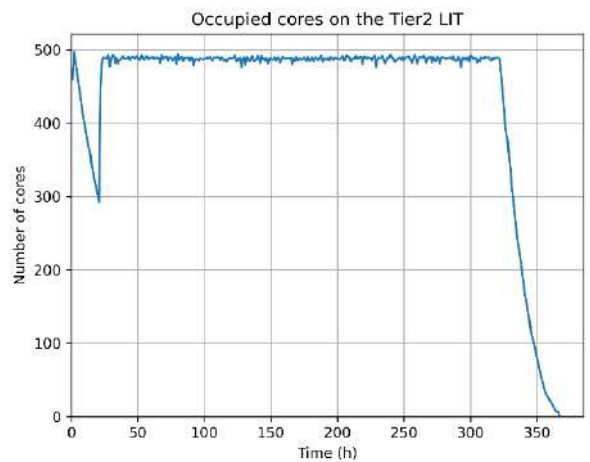


Figure 70. Usage of resources of the Tier2 LIT computing component

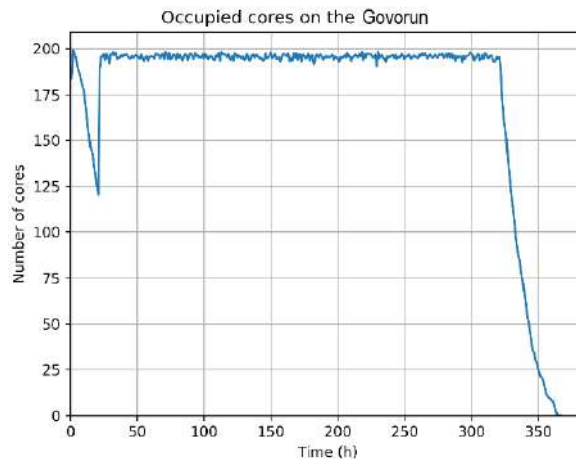


Figure 71. Usage of resources of the Govorun computing component

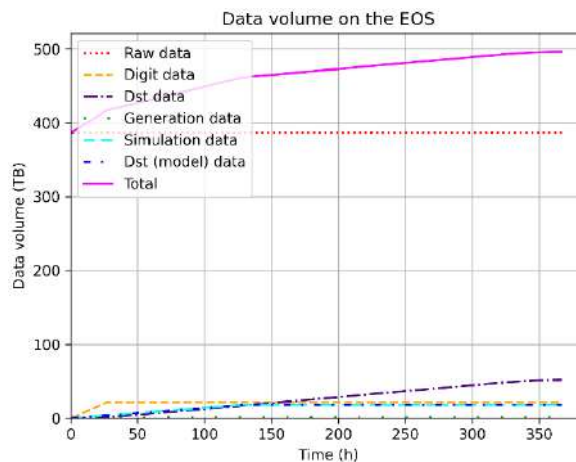


Figure 72. Data volume in the EOS storage after processing of experimental and modeled data

Table 2 summarizes the results of the computational experiments performed on the DT for easier comparison. The construction of the DT was performed to select the hardware configuration that will ensure data storage and processing with regard to the planned parameters of data flows of future BM@N experiment sessions. The most preferred selection criterion is the processing time of all data, which should be the shortest possible.

Based on the DT results, it is concluded that the best equipment configuration is the one that will allow for the least amount of time to process all experimental (367 hours) and modeled data (130 hours) using all available resources. The configuration should include the following components: Online farm — 1 000 cores, NICA LHEP — 1 000 cores, Tier1 LIT — 750 cores, Tier2 LIT — 500 cores, Govorun — 200 cores, EOS — at least 500 TB. It is reasonable to run *RawToDigit* jobs only on Tier1 LIT resources, *DigitToDst* jobs — on all available computational components, *GenToSim* and *SimToDst* jobs — only on Online farm and NICA LHEP. The selected alternative, taking into account the full use of available resources, will provide a **2 times speed up** of the raw data conversion process. **The proposed tactics of managing the jobs flow** with different types when distributing them to computing resources will allow to process not only experimental but also model data in the same time (3 weeks

after the end of the experiment). Therefore, more efficient resource usage is achieved, robust scaling scenarios are selected and data flows and job flows are managed.

Table 2. Comparison of the DT results under different equipment configurations of the BM@N experiment computing infrastructure

	Configurations of the first computational experiment		Configurations of the second computational experiment
	a	b	
	Convert all raw data to digit format (<i>RawToDigit</i> jobs)	60 hours (2,5 days)	28 hours (1 day)
Time for complete processing of all experimental data	432 hours (18 days)	240 hours (10 дней)	367 hours (15 days)
Time for complete processing of all model data	--	--	130 hours (6 days)

In the considered example of the special software application for the creating of the DT of the computing infrastructure of the BM@N experiment of the NICA complex, the problem of searching for the equipment configuration for the system of data acquisition, storage and processing has been solved. The importance of the conducted research is confirmed by the letter of application of the results in the BM@N experiment (Appendix 4). The results of the work are of particular practical importance, as they allowed us to estimate the required amount of resources for data storage and processing taking into account the planned parameters of data flows of future sessions of the BM@N experiment. It was found how much time will be required to process experimental and model data at the end of the experiment session. The obtained results helped to make the right decision in the process of job flow management and more correctly distribute the load on computational resources. The total operation time of the DT for all computational experiments (from the moment of running to the results) was 20 hours, which not only meets the specified requirements, but also allows to test a large number of options for modernization of the DDC in a short time. This, of course, contributes to more rapid decision-making on the development of infrastructure, on which new achievements of scientists directly depend.

4.3. Application of the special software to create the digital twin of the computing system of the online data filter of the SPD experiment of the NICA complex

The NICA complex has components that are under construction, such as the SPD detector. The SPD experiment is designed to study the proton-deuteron spin structure and other spin phenomena with polarized proton and deuteron beams at collision energies up to 27 GeV and luminosities up to

$10^{32} \text{ cm}^{-2}\text{s}^{-1}$ [78]. According to the technical documentation, the new facility will receive data at a rate of 20 GB/sec, which corresponds to 200 PB/year [79]. Designing a large-scale system for storing and processing such a huge amount of experimental data requires special attention. In this regard, it was proposed to use the developed special software to create a DT of the SPD experiment computing infrastructure in order to test subsystems, in particular the online data filter, with different variants of hardware parameters, data flows, and job flows.

The main goal of the online data filter is to quickly reconstruct events arising from particle collisions and suppress background events by a factor of at least 20 in real time. Given that the data arrival rate is planned to be 20 GB/sec, fast reconstruction and filtering of SPD detector data cannot be performed on a single computational node (processor or core). This means that the online filter computing system should be a specialized high-performance cluster including several data storage systems (for receiving detector data and for intermediate storage of filtered data before transferring them to long-term storage) and a large number of identical data processing worker nodes [79].

The online filter will work as follows. Unprocessed raw data of the detector at a rate of 20 GB/sec will be received into the data reception buffer, where the recording of experimental data into raw files is planned. The size of an individual raw file is 4 GB. The background event suppression process, which is performed on some set of computing resources in real time at a rate of 1 000 events/sec, includes three processing steps: decoding, partial reconstruction, and data filtering. Each stage generates the resulting files: dec, prec and filtered, respectively, which are written to intermediate data storage resources. The size of a separate dec-file is 4 GB, prec-file — 8 GB, filtered-file — 450 MB [79].

Thus, in order to ensure fast reconstruction of events and real-time data filtering, it is necessary to design an efficient computing system, providing opportunities for its further development and optimization. Let us consider the results of using the developed special software to create a DT of system of data acquisition, storage and processing for online filtering of SPD experiment.

The first application of the program complex for creation of the online filtering computer system DT is caused by the necessity to calculate the required parameters of the equipment for data storage and processing, as well as to estimate the load of the data transmission network. In the example under consideration, the task of creating a DT for acquisition and filtering the data of the experiment, which will operate for 24 hours with the following periodicity: 1 hour of work and 3 hours of break. An additional requirement is put forward to the time of raw-file processing: all three stages until the filtered file is received should last no more than 10 minutes. Files in data storage systems are not deleted.

Figure 73 shows an element of the special software web-interface, which allows describing the infrastructure of the considered computing system and the parameters of the equipment included in it.

The infrastructure includes such objects as a data generator (Trigger), a computing component (Computing), and two data storages (Buffer and Intermediate). Data flows and job flows were configured in accordance with the presented description of the online filter operation process.



Figure 73. Computational system for online data filtering of SPD experiment

After building the infrastructure, we can proceed to the creation of the DT, for the start of which it is necessary to configure some parameters: set the duration of operation of the computing system, add probabilistic events that can occur in the system, as well as specify objects and events for logging. It is important to note that with the specified duration of the experiment (24 hours) there is a limitation on the time of active data acquisition (1 hour of work and 3 hours of break), in this regard, the efficiency of data generation (20%) is specified in the DT settings.

The final results of the DT were obtained 40 minutes after its run. Let us consider the results in more detail on the images exported from the web service.

At a generation rate of 20%, the SPD facility will produce approximately 400 TB of experimental data in 24 hours (see Fig. 74). Processing all raw files in real time will require approximately 1 400 compute nodes (see Fig. 75), 120 of which will be dedicated to decoding the experimental data (*RawToDec*), 430 to partially reconstructing the decoded data (*DecToPrec*), and 850 to directly filtering the reconstructed data (*PrecToFilt*). The required data volume for the intermediate storage, where the resulting files of the three processing steps are located, is approximately 1 250 TB (see Fig. 76). In this case, 400 TB will occupy decoded data (Dec), 800 TB — partial reconstruction data (Prec) and 50 TB — filtered data (Filt). The load of communication links during data transmission is shown in Figures 77-

79. We conclude that a communication link with a bandwidth of at least 50 Gbps should be provided between the SPD facility and the data reception buffer (see Fig. 77), between the data reception buffer and computing resources — 40 Gbps (see Fig. 78), between computing resources and intermediate data storage — 250 Gbps (see Fig. 79).

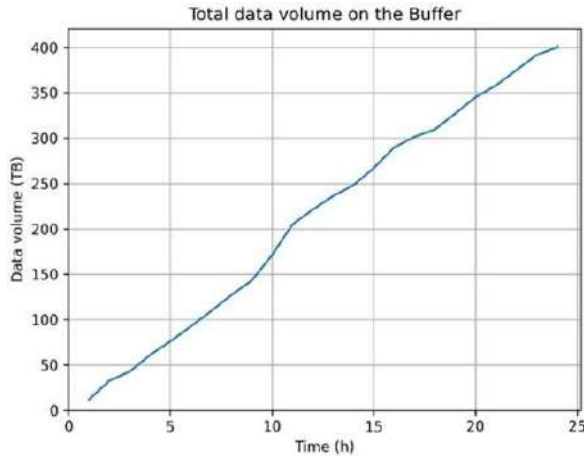


Figure 74. Amount of accumulated experimental data for 24 hours

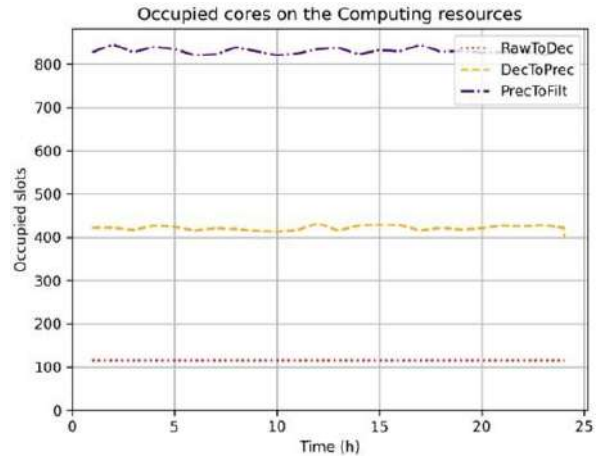


Figure 75. Using computational resources to process data to suppress background events

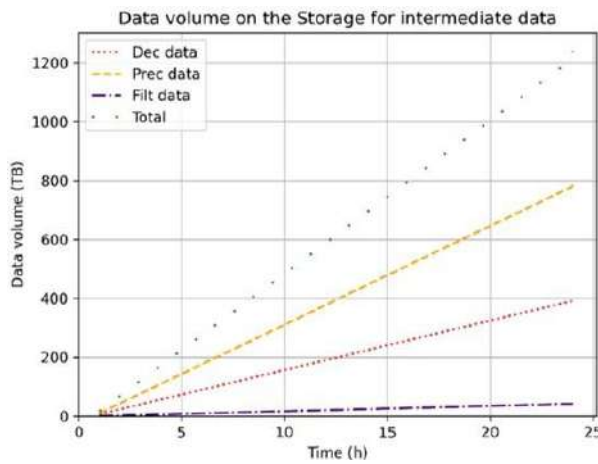


Figure 76. Volume of resulting data in intermediate storage

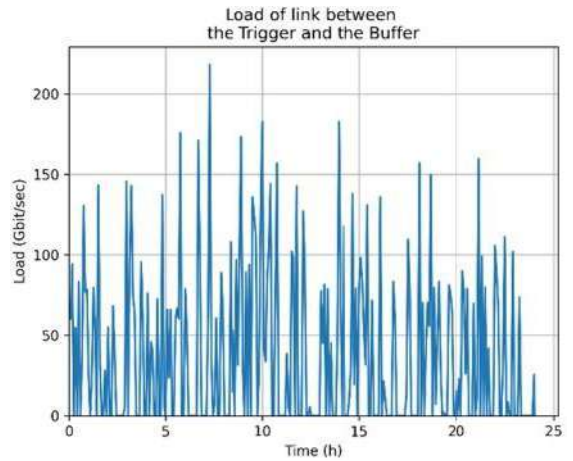


Figure 77. Loading the data link between the SPD facility and the data receive buffer

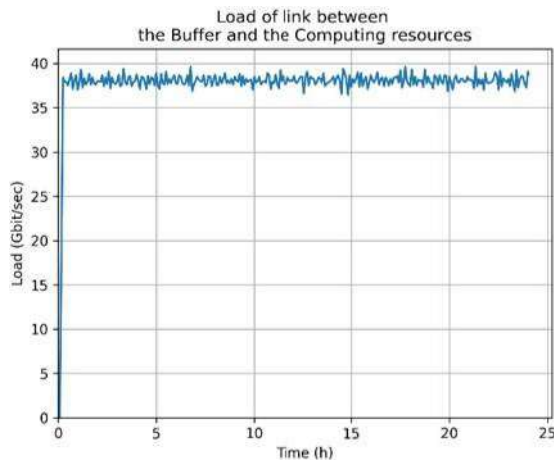


Figure 78. Loading of the data link between the data reception buffer and computing resources

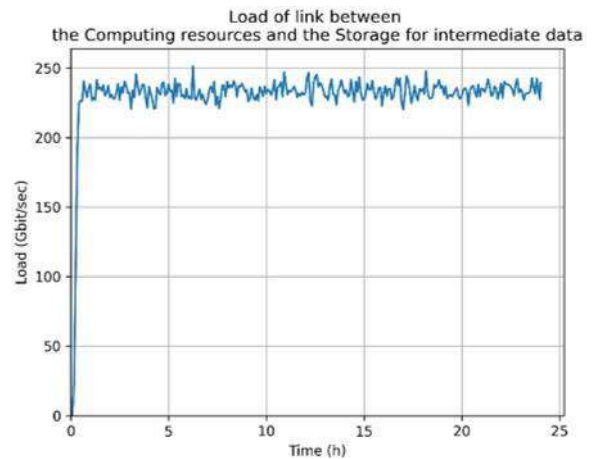


Figure 79. Loading of the data link between computing resources and intermediate data storage

The following conclusions are drawn from the results of the work performed. Taking into account 20% of data generation efficiency, 400 TB of data will be generated during 24 hours of the experiment operation, the processing of which requires at least 1 500 computational nodes to ensure fast reconstruction of events and real-time data filtering, provided that the bandwidth of communication channels is provided at the level of 50 Gbps. The intermediate storage uses 1 250 TB to temporarily store the filtered data before transferring it to the long-term storage. The data transfer rate from computing resources to intermediate storage is 250 Gbps.

The results of the application of special software to the create of the DT of the online data filtering system of the SPD experiment of the NICA complex proved the possibility of using the development to solve the problem of designing the DDC. The high practical significance of the conducted work is confirmed by the letter of application of the research results in the SPD experiment (Appendix 4). The results of the research allowed us to estimate the required parameters of the equipment for storage, processing and transmission of online filter data, taking into account the planned characteristics of the SPD experiment data flows. The operation time of the DT (from the moment of startup to obtaining the results) amounted to 40 minutes, which not only meets the specified requirements, but also will allow testing the online filter computing system with different variants of equipment parameters, data flows and job flows in a short time. This will undoubtedly ensure quality development and support of the computing infrastructure. All of the above confirms that the obtained results allow scaling and transformation of the system for different tasks and requirements.

4.4. Conclusions to Chapter 4

The verification and experimental operation of the special software are performed. The adequacy of the developed DT models of the data acquisition, storage and processing center of the BM@N experiment of the NICA complex is experimentally confirmed by the verification. The computing infrastructure was used to receive, store and process data from the latest session of the experiment, which took place from December 2022 to February 2023. The verification results prove the correct operation of the simulation program. The adequacy is assessed by several indicators. The obtained values proved that the deviations of the DT results from the results of real DDC will not be more than **three standard deviations** of the statistical monitoring data.

The reliability of the recommendations and conclusions based on the results of the research is confirmed by the practice of using the developed methods in the design and development of computing infrastructures for large-scale experiments in the field of high-energy physics.

The experimental operation of the special software is performed using examples of creating a digital twin for the computing infrastructure of the BM@N experiment and a digital twin for the computing system of the online data filter of the SPD experiment of the NICA complex. The applicability of the special software and the possibility of using the DT for the design, improvement of the operation efficiency, quality and reliability of complex data acquisition, storage and processing systems are shown.

DTs are successfully created with the help of the developed special software. The characteristics of the equipment necessary for storing, processing and transferring data, taking into account the planned parameters of data flows of future sessions of the NICA complex experiments, are found on the basis of the DT results. The main criterion is the minimization of the data processing time. The obtained results provide a qualitative check of the DDC functioning, as well as help to make the right decision in the process of managing job flows and more correctly distribute the load on computing resources. The scientific justification of the choice of DDC configurations confirms the importance of the conducted research. The application of the research results to solve the task of finding an equipment configuration for the computing infrastructures of the BM@N and SPD experiments of the NICA complex at JINR is confirmed by the corresponding letters of application (Appendix 4).

The time required to receive the DT results (from the moment of launch to the receipt of the results), which depends on the scale of the RDC and the processes occurring in it, does not exceed 24 hours. This is an additional advantage since it contributes to more prompt decision-making on the development of the DDC.

The third provision to be defended is proved: “The adequacy of the constructed methods and algorithms is confirmed using the example of the computational infrastructure of the existing experiment”.

Conclusion

A new method of creating and using DDC digital twins is developed in the PhD thesis in order to improve DDC technical characteristics. The proposed method differs from existing ones in the ability to simulate such processes as data storage and processing, taking into account the characteristics of data flows and jobs, the probabilities of failures and changes in the equipment performance and other processes occurring in the simulated system.

The main results of the PhD thesis are as follows.

1. The system connections and patterns of the functioning of complex systems, which are DDCs, are investigated using the principles of a systematic approach and the methods of system analysis.

2. Methods for describing distributed systems, making decisions on the choice of equipment configurations, and managing resources and processes of complex systems are developed.

3. Models, methods and algorithms for creating DDC digital twins are developed.

4. Algorithms, the DB structure and a web user interface are implemented for creating and executing a DT, as well as providing graphical information about the results of its work. Modern architectural solutions and tools for developing software, web applications and DBs are used.

5. Special software is developed on the basis of the created models, methods and algorithms. The software allows comparing the efficiency of the DDC operation depending on different hardware configurations.

6. The certificate of state registration of the computer program No. 2023667305 “Software Complex for Creating Digital Twins of Distributed Data Acquisition, Storage and Processing Centers” dated 14 August 2023 is received (Appendix 3).

7. The verification of the DT kernel is performed using the example of the computing infrastructure of the BM@N experiment of the NICA accelerator complex. The adequacy is assessed by several indicators. The resulting values prove that the deviations of the DT results from the results of real DDC will not be more than **three standard deviations** of statistical monitoring data.

8. The experimental operation of the special software is performed during the creation of the DT of the computing infrastructure of the BM@N experiment. The most suitable configuration of data processing equipment in the shortest possible time is obtained. A strategy for managing job flows and distributing the load on computing resources is chosen. The chosen alternative will provide a **speed-up by 2 times** of the raw data conversion process. **The proposed tactics for managing job flows** will allow processing not only experimental, but also model data at the same time.

9. The experimental operation of the special software is performed during the creation of the DT of the online data filter system of the SPD experiment of the NICA complex. The results enable to

evaluate the required parameters of the equipment for storing, processing and transferring online filter data, taking into account the planned characteristics of experimental data flows.

10. The time required to receive the DT results (from the moment of launch to the receipt of the results) has been determined, taking into account the scale of the DDC and the processes taking place in it, which **does not exceed 24 hours**. This is an additional advantage, as it facilitates faster decision-making on the development of DDCs.

11. Recommendations on the results of the DT functioning are taken into account in the design and development of computing infrastructures for large-scale experiments in the field of high-energy physics. The results of the software application prove the efficiency and high quality of the models and algorithms developed in the thesis, which is confirmed by the corresponding letters of application (Appendix 4).

12. The results of the research are used in the educational process of the Federal State Budgetary Educational Institution of Higher Education “Dubna University” in the course “Distributed Computing and Cloud Technologies” for the preparation of master’s students in the field of 27.04.03 System Analysis and Management in the profile “Digital Platforms and Big Data Analytics”, which is confirmed by the corresponding act of implementation (Appendix 5).

The results of the PhD thesis are an important contribution to the development of scientific research, which provides significant assistance in designing, creating, supporting, as well as solving management and development tasks of data acquisition, storage and processing centers for large scientific projects. The developed method of creating DTs allows one to build a prototype of a computing infrastructure, which helps to assess the efficiency of the DDC operation, contributes to prompt decision-making on scaling a distributed system and changing the characteristics of its constituent equipment, which leads to an increase in the quality and reliability of complex data acquisition, storage and processing systems.

In the future, the developed software [80] can be used for a wide class of tasks in the field of design, construction and development of DDCs for large scientific experiments and large-scale projects.

In the long-term development of this work, it is planned to improve the developed method and add a multi-criteria optimization function in choosing an equipment configuration for a DDC. Not only the technical, but also cost parameters of the equipment included in the DDC [81, 82] will have to be taken into account as criteria. It is necessary to add a user’s personal account, and to ensure security, a module for registration, i.e., provide access to the special software for creating DTs only after passing the authorization procedure. It will allow enhancing the convenience of using the special software.

Glossary of terms

1. **adequacy**: Compliance of the model of a distributed data acquisition, storage and processing center with a real system according to a certain list of characteristics.

2. **verification**: A procedure for checking a computer program for the correctness of the implementation of the task by evaluating the results of its work according to specified criteria; in this paper, the results of the work of the distributed system modeling program are compared with the results of monitoring this system.

3. **computational experiment**: An experiment on a digital twin of a distributed data acquisition, storage and processing system to study its behavior under a certain scenario with specified parameters of equipment, data flows and jobs flows.

4. **special software**: A set of programs that implements a method for solving specific tasks, namely in this work: a new method for creating, including building, configuring, executing digital twins of distributed data acquisition, storage and processing centers, as well as demonstrating the results of its work.

5. **kernel of the digital twin**: a computer program that implements algorithms for modeling various processes occurring in the system, in particular, a distributed data acquisition, storage and processing center.

References

1. Grieves M. Digital Twin: Manufacturing Excellence Through Virtual Factory Replication. Digital Twin White Paper // ResearchGate. 2014. URL: https://www.researchgate.net/publication/275211047_Digital_Twin_Manufacturing_Excellence_through_Virtual_Factory_Replication (date of access: 15.03.2023).
2. Prokhorov A., Lysachev M. Digital twin. Analysis, trends, world experience. 1st ed. M.: LLC “AlliancePrint”, 2020. 401 pp. (in Russian).
3. Kornilov V.V., Isaev V.A., Isaev K.A. Prospects of using data processing centers in solving problems of mathematical biology and bioinformatics // Mathematical Biology and Bioinformatics. February 2015. Vol. 10. No. 1. pp. 60–71 (in Russian).
4. Lyashenko M.A. Content of the data center development strategy // Internet journal “Naukovedenie”. July – August 2015. Vol. 7. No. 4 (in Russian).
5. Business Ecosystems. Data Centers. Increase scalability, flexibility and business security // Business Ecosystems. URL: https://becsys.ru/uploads/files/solutions/technological-solutions/3/Business_Ecosystems_Data_Centers.pdf (date of access: 15.03.2023) (in Russian).
6. The Digital Twin Company [Electronic resource] // Future Facilities: [сайт]. URL: <https://www.futurefacilities.com/> (date of access: 20.12.2022).
7. Sunbird Software, Inc. DCIM - Data Center Infrastructure Management Software System, Cable Management, Infrastructure Design & Optimization Companies [Electronic resource] // Sunbird DCIM: [сайт]. URL: <https://www.sunbirdcim.com/> (date of access: 20.12.2022).
8. Petrov A.V. Simulation modeling as the basis of digital twin technology // Bulletin of Irkutsk State Technical University. 2018. Vol. 22. No. 10. pp. 56–66 (in Russian).
9. Foster I., Kesselman C. The grid: blueprint for a new computing infrastructure. San Francisco (CA): Morgan Kaufmann Publishers Inc., 1999. 593 pp.
10. CERN. Welcome to the Worldwide LHC Computing Grid [Electronic resource] // WLCG: [сайт]. URL: <https://wlcg.web.cern.ch/> (date of access: 10.12.2022).
11. Berezhnaya A., Dolbilov A., Ilyin V., Korenkov V., Lazin Y., Lyalin I., Mitsyn V., Ryabinkin E., Shmatov S., Strizh T., et al. LHC Grid Computing in Russia: present and future // Journal of Physics: Conference Series. 2014. Vol. 513. No. 6. P. 062041.
12. Kekelidze V., Kovalenko A., Lednicky R., Matveev V., Meshkov I., Sorin A., Trubnikov G. Status of the NICA project at JINR // European Physical Journal Web of Conferences. March 2017. Vol. 138. P. 01027.

13. Serebrov A.P., Vassiljev A.V., Varlamov V.E., Geltenbort P., Gridnev K.A., Dmitriev S.P., Dovator N.A., Egorov A.I., Ezhov V., Zherebtsov O.M., et al. Program for studying fundamental interactions at the PIK reactor facilities // *Physics of Atomic Nuclei*. June 2016. Vol. 79. No. 3. pp. 293–303.
14. Baranov G.N., Bogomyakov A.V., Levichev E.B., Sinyatkin S.V. Optimization of the magnetic structure of the synchrotron radiation source of the fourth generation SKIF in Novosibirsk // *Siberian Physical Journal*. 2020. Vol. 15. No. 1. C. 5–23 (in Russian).
15. Avrorin A.D., Avrorin A.V., Aynutdinov V.M., Bannash R., Belolaptikov I.A., Brudanin V.B., Budnev N.M., Doroshenko A.A., Domogatsky G.V., Dvornický R., et al. Baikal-GVD: status and prospects // *European Physical Journal Web of Conferences*. October 2018. Vol. 191. P. 01006.
16. Fengpeng A., et al. Neutrino physics with JUNO // *Journal of Physics G: Nuclear and Particle Physics*. February 2016. Vol. 43. No. 3. P. 030401.
17. Priakhina D.I., Korenkov V.V. Relevance of creating a digital twin for managing distributed data acquisition, storage and processing centers // *Modern Information Technologies and IT-Education*. 2023. Vol. 19. No. 2. pp. 262–271 (in Russian).
18. Nechaevskiy A.V., Korenkov V.V.. DataGrid modeling packages // *System analysis in science and education*. 2009. No. 1. pp. 21–35 (in Russian).
19. Samovarov O.I., Kuzurin N.N., Grushin D.A., Avetisyan A.I., Mikhailov G.M., Rogov Y.P. Problems of modeling GRID-systems and their implementation // *Scientific service in the Internet: solving big problems*. Moscow. 2008. pp. 83–88 (in Russian).
20. Korenkov V.V., Nechaevskiy A.V., Ososkov G.A., Priakhina D.I., Trofimov V.V., Uzhinsky A.V. Modeling of grid and cloud services as a means to improve the efficiency of their development // *CEUR Workshop Proceedings*. 2014. Vol. 1297. pp. 13–19 (in Russian).
21. GridSim: A Grid Simulation Toolkit 5.2 [Electronic resource] // *Soft 112: [сайт]*. [2010]. URL: <https://gridsim-a-grid-simulation-toolkit.soft112.com/> (date of access: 10.10.2022).
22. The University of Edinburgh. API Specification [Electronic resource] // *SimJava v2.0: [сайт]*. [2002]. URL: <https://www.icsa.inf.ed.ac.uk/research/groups/hase/simjava/doc/index.html> (date of access: 23.12.2022).
23. Korenkov V.V., Nechaevskiy A.V., Ososkov G.A., Priakhina D.I., Trofimov V.V., Uzhinsky A.V. Synthesis of modeling and monitoring processes for the development of systems for storage and processing of large data arrays in physical experiments // *Computer Research and Modeling*. 2015. Vol. 7. No. 3. pp. 691–698 (in Russian).
24. Kadochnikov I., Korenkov V., Mitsyn V., Pelevanyuk I., Strizh T. Service monitoring system for JINR Tier-1 // *The European Physical Journal Conferences*. Sep 2019. Vol. 214. P. 08016.

25. Korenkov V.V., Nechaevskiy A.V., Ososkov G.A., Priakhina D.I., Trofimov V.V., Uzhinsky A.V. Modeling of grid and cloud services as an important stage of their development // *Systems and Means of Informatics // Systems and Means of Informatics*. 2015. Vol. 25. No. 1. pp. 4–19 (in Russian).
26. Nechaevskiy A.V., Priakhina D.I., Uzhinsky A.V. Development of web-service for modeling of systems of storage and processing of physical experiments data // *System Analysis in Science and Education*. 2015. No. 4. pp. 1–7 (in Russian).
27. Korenkov V., Nechaevskiy A., Ososkov G., Pryahina D., Trofimov V., Uzhinskiy A., Balashov N. Web-Service Development of the Grid-Cloud Simulation Tools // *Procedia Computer Science*. 2015. Vol. 66. pp. 533–539.
28. Korenkov V., Nechaevskiy A., Ososkov G., Pryahina D., Trofimov V., Uzhinskiy A. Simulation concept of NICA-MPD-SPD Tier0-Tier1 computing facilities // *Particles and Nuclei Letters*. 2016. Vol. 13. No. 5. pp. 1074–1083.
29. Kutovsky N.A., Nechaevskiy A.V., Ososkov G.A., Priakhina D.I., Trofimov V.V.. Modeling of interprocessor interaction when executing MPI-applications in the cloud // *Computer Research and Modeling*. 2017. Vol. 9. No. 6. pp. 955–963 (in Russian).
30. Nechaevskiy A., Ososkov G., Pryahina D., Trofimov V., Li W. Simulation approach for improving the computing network topology and performance of the China IHEP Data Center // *European Physical Journal Web of Conferences*. 2019. Vol. 214. P. 08018.
31. Grieves M., Vickers J. Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems // In: *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches*. Springer, Cham, 2017. pp. 85-113.
32. Barricelli B.R., Casiraghi E., Fogli D. Definitions, Characteristics, Applications, and Design Implications // *IEEE Access*. November 2019. Vol. 7. pp. 167653–167671.
33. Denisov A.S., Kuverin I.Yu. Digital twins as the basis of digital transformation of technical operation of vehicles within the fourth technological revolution // *Technical regulation in transport construction*. 2020. Vol. 3. No. 42. pp. 165–168 (in Russian).
34. Yang J., Zhang W., Liu Y. Subcycle fatigue crack growth mechanism investigation for aluminum alloys and steel // *54th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. Boston. 2013. P. 1499.
35. Shalumov A.S., Shalumova N.A., Shalumov M.A. Digital twin of avionics: modeling of physical processes during the formation of electronic model // *Automation. Modern Technologies*, Vol. 75, No. 9, 2021. pp. 403–415 (in Russian).

36. Filatov A.R. Digital twin of the ship hull. Purpose and Basic Principles of Construction // Proceedings of the Krylov State Research Center. 2021. Vol. 4. No. 398. pp. 87–92 (in Russian).
37. Shevchenko D.V. Methodology of building digital twins on the railway transport // Bulletin of the Railway Transport Research Institute. 2021. Vol. 80. No. 2. pp. 91–99 (in Russian).
38. Bykova V.N., Kim E., Gadzhialiev M.R., Musienko V.O., Orudzhev A.O., Turovskaya E.A. Application of digital twin in the oil and gas industry // Actual Problems of Oil and Gas. 2020. Vol. 28. No. 1. pp. 8 (in Russian).
39. Tikhonov A.I., Stulov A.V., Karzhevin A.A., Podobnyi A.V. Development of a nonlinear model of a three-phase transformer for investigation of the influence of the magnetic system asymmetry on the device operation in arbitrary modes // Bulletin of Ivanovo State Power Engineering University. 2020. No. 1. pp. 22–31 (in Russian).
40. Abramov V.I., Stolyarov A.D. Digital twins in agriculture: opportunities and prospects // AIC of Russia: education, science, production. Collection of articles of the II All-Russian (national) scientific-practical conference. Penza. 2021. pp. 3–9 (in Russian).
41. Bruynseels K., Santoni de Sio F., Hoven J. Digital Twins in Health Care: Ethical Implications of an Emerging Engineering Paradigm // Frontiers in genetics. February 2018. Vol. 9. No. 31.
42. Menshutina N.V. Multilevel modeling of aerogels and their production // Supercritical Fluids, Fundamentals, Technologies, Innovations. Sochi. 2017. pp. 124–126 (in Russian).
43. Lehtola V.V., Koeva M., Elberink S.O., Raposo P., Virtanen J.P., Vahdatikhaki F., Borsci S. Digital twin of a city: Review of technology serving city needs // International Journal of Applied Earth Observation and Geoinformation. 2022. Vol. 114. P. 102915.
44. Ivanov S.A., Nikolskaya K.Yu., Radchenko G.I., Sokolinsky L.B., Tsymler M.L. Concept of building a digital twin of the city // Bulletin of SUSU. Series: Computational Mathematics and Informatics. 2020. Vol. 9. No. 4. pp. 5–23 (in Russian).
45. Amirkhanyan A.G. Digital twins in logistics // Modern science. 2020. No 1–2. pp. 37–40 (in Russian).
46. Rosen R., Von Wichert G., Bettenhausen K.D. About the importance of autonomy and digital twins for the future of manufacturing // IFAC-PapersOnLine. 2015. Vol. 48. No. 3. pp. 567–572.
47. Polyniak K., Matthews J. The Johns Hopkins Hospital Launches Capacity Command Center to Enhance Hospital Operations // John Hopkins Medicine. 2016. URL: https://www.hopkinsmedicine.org/news/media/releases/the_johns_hopkins_hospital_launches_capacity_command_center_to_enhance_hospital_operations (date of access: 18.12.2022).

48. Nemati K., Zabalegui A., Bana M., Seymour M.J. Quantifying data center performance // 34th Thermal Measurement, Modeling & Management Symposium. 2018. pp. 141–147.
49. Batty M. Digital twins. Environment and Planning B // Urban Analytics and City Science. 2018. Vol. 5. No. 45. pp. 817–820.
50. Boschert S., Rosen R. Digital Twin — The Simulation Aspect // In: Mechatronic Futures. Springer, Cham, 2016. pp. 59–74.
51. Cimino C., Negri E., Fumagalli L. Review of digital twin applications in manufacturing // Computers in Industry. December 2019. Vol. 113. P. 103130.
52. Halenar I. 20th International Carpathian Control Conference // Virtualization of Production Using Digital Twin Technology. Krakow-Wieliczka, Poland. 2019. pp. 1–5.
53. Khitrykh D.P. Digital twins: past, present and future // CADFEM REVIEW - Scientific and Technical Journal from the company CADFEM. 2021. Vol. 8. No. 31. pp. 30–32 (in Russian).
54. Korenkov V., Nechaevskiy A., Ososkov G., Priakhina D., Trofimov V. A probabilistic approach of the simulation of data processing centers // European Physical Journal Web of Conferences. January 2020. Vol. 226. P. 03012.
55. Romanov V.N. Fundamentals of system analysis / educational and methodical complex. St. Petersburg: NWTU, 2011. 298 pp. (in Russian).
56. Priakhina D.I., Korenkov V.V., Trofimov V.V. Method for creating digital twins to solve the tasks of effective management and development of distributed data acquisition, storage and processing centers // Modern Information Technologies and IT-Education. 2023. Vol. 19. No. 2. pp. 272–281 (in Russian).
57. Ramalho L. Fluent Python: Clear, Concise, and Effective Programming. 1st ed. O'Reilly Media, 2015.
58. Rubio D. Beginning Django: Web Application Development and Deployment with Python. Mexico: Ensenada, 2017.
59. Dronov V.A. Django 2.1. Practice of creating websites in Python. St. Petersburg: BHV-Peterburg, 2019 (in Russian).
60. Novikov B.A. Fundamentals of database technologies: textbook. 2nd ed. Moscow: DMK Press, 2020 (in Russian).
61. Führer C., Solem J.E., Verdier O. Scientific Computing with Python. 2nd ed. Packt Publishing, 2021.
62. Argparse Tutorial [Electronic resource] // Python 3.12.1 documentation: [сайт]. URL: <https://docs.python.org/3/howto/argparse.html> (date of access: 17.07.2021).

63. configparser — Configuration file parser [Electronic resource] // Python 3.12.1 documentation: [website]. URL: <https://docs.python.org/3/library/configparser.html> (date of access: 17.07.2021).
64. Graph theory (network) library for visualisation and analysis [Electronic resource] // Cytoscape.js: [website]. URL: <https://js.cytoscape.org/> (date of access: 20.05.2022).
65. Dabbas E. Interactive dashboards and applications with PLOTLY and DASH. DMK Press, 2023 (in Russian).
66. Django documentation [Electronic resource] // Django: [сайт]. URL: <https://docs.djangoproject.com/en/5.0/> (date of access: 17.05.2022).
67. Priakhina D.I., Korenkov V.V., Trofimov V.V., Gertsenberger K.V. Verification of the simulation program for creating digital twins of distributed data acquisition, storage and processing centers // International Journal of Open Information Technologies. January 2024. Vol. 12. No. 1. pp. 118–128 (in Russian).
68. Kapishin M., BM@N Collaboration. Studies of baryonic matter at the BM@N experiment (JINR) // Nuclear Physics A. 2019. Vol. 982. P. 967.
69. JINR. NICA Bulletin // LABORATORY OF HIGH ENERGY PHYSICS - named after V.I.Vexler and A.M.Baldin. 2023. URL: https://lhep.jinr.ru/wp-content/uploads/2023/03/nica_vypusk9.pdf (date of access: 20.04.2023).
70. Baginyan A., Balandin A., Balashov N., Dolbilov A., Gavrish A., Golunov A., Gromova N., Kashunin I., Korenkov V., Kutovskiy N., et al. Current status of the MICC: an overview // CEUR Workshop Proceeding. 2021. Vol. 3041. pp. 1–8.
71. Adam G., Bashashin M., Belyakov D., Kirakosyan M., Matveev M., Podgainy D., Sapozhnikova T., Streltsova O., Torosyan S., Vala M., et al. IT ecosystem of the HybriLIT heterogeneous platform for high performance computing and training of IT specialists // CEUR Workshop Proceeding. 2018. Vol. 2267. pp. 638–644.
72. Korenkov V., Pelevanyuk I., Tsaregorodtsev A. DIRAC at JINR as a general purpose system for massive computations // Journal of Physics: Conference Series. 2023. Vol. 2438. P. 012029.
73. Pelevanyuk I. Performance evaluation of computing resources with DIRAC interware // AIP Conference Proceedings. September 2021. Vol. 2377. No. 1. P. 040006.
74. Gertsenberger K., Pelevanyuk I. BM@N Mass Data Production on distributed infrastructure for Run 8 using DIRAC, 10th Collaboration Meeting of the BM@N Experiment at the NICA Facility , St. Petersburg, Presentation 2023.
75. Priakhina D., Trofimov V., Ososkov G., Gertsenberger K. Data center simulation for the BM@N experiment of the NICA project // AIP Conference Proceeding. 2021. Vol. 2377. P. 040007.

76. Priakhina D., Korenkov V., Gertsenberger K., Trofimov V. Simulation of Data Processing for the BM@N Experiment of the NICA Complex // CEUR Workshop Proceedings. 2021. Vol. 3041. pp. 483–487.
77. Priakhina D., Korenkov V., Trofimov V., Gertsenberger K. Simulation Results of BM@N Computing Infrastructure // Physics of Particles and Nuclei Letters. 2023. Vol. 20. No. 5. pp. 1272–1275.
78. The SPD proto-collaboration. Conceptual design of the Spin Physics Detector // International spin physics collaboration at the collider NICA. 2021. URL: <http://spd.jinr.ru/wp-content/uploads/2021/04/2102.00442.pdf> (date of access: 25.09.2023).
79. The SPD collaboration. Technical Design Report of the Spin Physics Detector // International spin physics collaboration at the collider NICA. 2022. URL: http://spd.jinr.ru/wp-content/uploads/2023/03/TechnicalDesignReport_SPD2023.pdf (date of access: 25.09.2023).
80. Korenkov V.V., Priakhina D.I., Trofimov V.V. Software Complex for Creating Digital Twins of Distributed Data Acquisition, Storage and Processing Centers, The certificate of state registration of the computer program No. 2023667305, August 14, 2023 (in Russian).
81. Trofimov V.V., Nechaevskiy A.V., Ososkov G.A., Priakhina D.I. Probability-cost approach to optimizing distributed data storage systems for physical experiments // CEUR Workshop Proceedings. 2018. Vol. 226. pp. 393–399 (in Russian).
82. Korenkov V.V., Priakhina D.I., Nechaevskiy A.V., Ososkov G.A., Trofimov V.V. Simulation of data storage and processing centers taking into account economic components // System analysis in science and education. 2018. No. 4. pp. 1–8 (in Russian).

Appendix 1. Description of data model entities

№	Attribute name	Attribute description	Type	Note
<i>DataStorages</i> entity				
1	<i>storage_id</i>	Storage identifier	integer	primary key; unique
2	<i>storage_name</i>	Storage name	character string	
3	<i>storage_description</i>	Storage description	character string	
4	<i>storage_volume</i>	Maximum storage volume (TB)	double precision	
5	<i>storage_active</i>	Storage activity (1 – active; 0 – inactive)	integer	
6	<i>storage_quant</i>	Data generation rate in the active storage (TB/s)	double precision	
7	<i>storage_priority</i>	Storage priority value	integer	
8	<i>storage_sensor</i>	Name of the statistics collection object	character string	foreign key
<i>ComputingComponents</i> entity				
1	<i>comp_id</i>	Computing component identifier	integer	primary key; unique
2	<i>comp_name</i>	Computing component name	character string	
3	<i>comp_description</i>	Computing component description	character string	
4	<i>comp_cores</i>	Total number of cores	integer	
5	<i>comp_speed</i>	Speed up factor	double precision	
6	<i>comp_sensor</i>	Name of the statistics collection object	character string	foreign key
<i>Pilots</i> entity				
1	<i>pilot_id</i>	Pilot identifier	integer	primary key; unique
2	<i>pilot_name</i>	Pilot name	character string	
3	<i>pilot_description</i>	Pilot description	character string	
4	<i>pilot_comp</i>	Computing component name	character string	foreign key
5	<i>pilot_queue</i>	Job flow name	character string	foreign key
6	<i>pilot_priority</i>	Pilot priority value	integer	
7	<i>pilot_jobs_part</i>	Percentage of jobs to be processed	double precision	
8	<i>pilot_storage_input</i>	Name of storage with data for processing	character string	foreign key
9	<i>pilot_storage_output</i>	Name of storage for saving data after processing	character string	foreign key
10	<i>pilot_sensor</i>	Name of the statistics collection object	character string	foreign key
<i>Slots</i> entity				
1	<i>slot_id</i>	Slot identifier	integer	

№	Attribute name	Attribute description	Type	Note
2	<i>slot_name</i>	Slot name	character string	primary key; unique
3	<i>slot_cores</i>	Number of cores in slot	integer	
4	<i>slot_pilot</i>	Pilot name	character string	foreign key
5	<i>slot_active</i>	Slot activity (1 – active; 0 – inactive)	integer	
6	<i>slot_sensor</i>	Name of the statistics collection object	character string	foreign key
<i>Links entity</i>				
1	<i>link_id</i>	Communication link identifier	integer	primary key; unique
2	<i>link_name</i>	Communication link name	character string	
3	<i>link_description</i>	Communication link description	character string	
4	<i>link_from</i>	Name of the object from which the data are transmitted	character string	
5	<i>link_to</i>	Name of the object to which the data are transmitted	character string	
6	<i>link_bandwidth</i>	Communication link bandwidth (TB/s)	double precision	
7	<i>link_active</i>	Link activity (1 – active; 0 – inactive)	integer	
8	<i>link_sensor</i>	Name of the statistics collection object	character string	foreign key
<i>DataTags entity</i>				
1	<i>tag_id</i>	Data type identifier	integer	primary key; unique
2	<i>tag_name</i>	Data type name	character string	
3	<i>tag_description</i>	Data type description	character string	
<i>DataFlows entity</i>				
1	<i>dataflow_id</i>	Data flow identifier	integer	primary key; unique
2	<i>dataflow_name</i>	Data flow name	character string	
3	<i>dataflow_description</i>	Data flow description	character string	
4	<i>dataflow_tag</i>	Data type name	character string	foreign key
5	<i>dataflow_storage</i>	Storage name	character string	foreign key
6	<i>dataflow_volume</i>	Maximum data flow size (TB)	double precision	
7	<i>dataflow_files</i>	Number of files in the data flow	integer	
8	<i>dataflow_active</i>	Data flow activity (1 – active; 0 – inactive)	integer	
9	<i>dataflow_sensor</i>	Name of the statistics collection object	character string	foreign key
<i>JobQueues entity</i>				
1	<i>queue_id</i>	Job flow identifier	integer	primary key; unique
2	<i>queue_name</i>	Job flow name	character string	

№	Attribute name	Attribute description	Type	Note
3	<i>queue_description</i>	Job flow description	character string	
4	<i>queue_input_tag</i>	Input data type name	character string	foreign key
5	<i>queue_input_volume</i>	Average input data volume (TB)	double precision	
6	<i>queue_input_mod</i>	Allowable value of input data volume variation (TB)	double precision	
7	<i>queue_output_tag</i>	Output data type name	character string	foreign key
8	<i>queue_output_vol</i>	Average output data volume (TB)	double precision	
9	<i>queue_output_mod</i>	Allowable value of output data volume variation (TB)	double precision	
10	<i>queue_runtime</i>	Average time of task execution (s)	integer	
11	<i>queue_runmod</i>	Permissible value of job execution time variation (s)	integer	
12	<i>queue_start_delay</i>	Delay of job start process (s)	integer	
13	<i>queue_temp</i>	Average time of job occurrence (s)	integer	
14	<i>queue_temp_mod</i>	Permissible value of job occurrence time variation (s)	integer	
15	<i>queue_power</i>	Total number of jobs	integer	
16	<i>queue_sensor</i>	Name of the statistics collection object	character string	foreign key
<i>TransportJobs</i> entity				
1	<i>transp_id</i>	Transfer object identifier	integer	primary key; unique
2	<i>transp_storage_from</i>	Name of the storage from which the data are transferred	character string	primary key;
3	<i>transp_storage_to</i>	Name of the storage to which the data are transferred	character string	foreign key; unique
4	<i>transp_link</i>	Name of communication link	character string	foreign key
5	<i>transp_tag</i>	Data type name	character string	foreign key
6	<i>transp_priority</i>	Priority value of the transmission object	integer	
7	<i>transp_part</i>	Percentage of data to be transferred	double precision	
8	<i>transp_time_start</i>	Transmission start time (s)	integer	
<i>Events</i> entity				
1	<i>event_id</i>	Event identifier	integer	primary key; unique
2	<i>event_name</i>	Event name	character string	unique

№	Attribute name	Attribute description	Type	Note
3	<i>event_description</i>	Event description	character string	
4	<i>event_sensor</i>	Name of the statistics collection object	character string	foreign key
<i>Event_Object</i> entity				
1	<i>event_id</i>	Event identifier	integer	foreign key
2	<i>object_name</i>	Object name	character string	
3	<i>object_type</i>	Object type	character string	
4	<i>distribution</i>	Distribution type	character string	
5	<i>probability</i>	Probability of event occurrence	double precision	
6	<i>initial_time</i>	Time of occurrence of the first event	integer	
7	<i>initial_value</i>	Initial value of the parameter of the object with which the event occurs	double precision	
8	<i>value</i>	Permissible value of change of the parameter of the object with which the event occurs	double precision	
9	<i>depend_events</i>	List of dependent events	character string	
<i>Sensors</i> entity				
1	<i>sn_id</i>	Statistics collection object identifier	integer	primary key; unique
2	<i>sn_name</i>	Statistics collection object name	character string	
3	<i>sn_description</i>	Statistics collection object description	character string	
4	<i>sn_period</i>	Averaging period of collected information	integer	
5	<i>sn_frequency</i>	Frequency of recording the collected information in the DB	integer	
<i>Experiments</i> entity				
1	<i>exp_id</i>	DT experiment identifier	integer	primary key; unique
2	<i>exp_name</i>	DT experiment name	character string	unique
3	<i>exp_description</i>	DT experiment description	character string	
4	<i>exp_params</i>	DT experiment parameters	character string	
5	<i>exp_log</i>	List of identifiers of the statistics collection object	character string	
6	<i>exp_date_create</i>	Date and time of creation	timestamp	
<i>Modifications</i> entity				
1	<i>mod_id</i>	Scenario identifier	integer	primary key; unique
2	<i>mod_experiment</i>	DT experiment identifier	integer	foreign key

№	Attribute name	Attribute description	Type	Note
3	<i>mod_report</i>	DT experiment scenario identifier	integer	unique
4	<i>mod_json</i>	DDC equipment parameters	текстовые данные <i>JSON</i>	
5	<i>mod_status</i>	Status of work (0 – created; 1 – in operation; 2 – completed)	integer	
6	<i>mod_date_create</i>	Date and time of creation	timestamp	
7	<i>mod_date_start</i>	Date and time of startup	timestamp	
8	<i>mod_date_finish</i>	Date and time of work completion	timestamp	
<i>SimulationReport</i> entity				
1	<i>report_id</i>	Record identifier	integer	primary key; unique
2	<i>report_systime</i>	System time of statistics collection	integer	
3	<i>report_equipment_id</i>	Identifier of the object, which is a part of the DDC, the information on the results of which is contained in the record	integer	
4	<i>report_equipment</i>	Name of the object included in the DDC, the information on the results of which is contained in the record	character string	
5	<i>report_variable</i>	Value describing the state of the object included in the DDC	double precision	
6	<i>report_comment</i>	Comment	character string	
7	<i>report_modification</i>	DT experiment scenario identifier	integer	foreign key

Appendix 2. User instructions for working with the special software

1. When entering the system, a page opens that invites the user to build an DDC infrastructure (see Fig. 80). The page has “active” buttons that show objects available for adding to the DDC infrastructure, namely: data generator, computing component, data storage, robotic library. It is necessary to select an object from the list (1) and place it in the area selected for drawing (2), after which a form for setting up the basic configuration of the device will open (3). It is required to enter all the necessary data and click on the “Add Device” button. Figure 81 shows an example of adding a data storage to the DDC infrastructure. When the infrastructure is completed, you should click the corresponding button, after which the buttons with infrastructure objects will become “inactive” (see Fig. 82). The resulting infrastructure image can be saved to a local device.

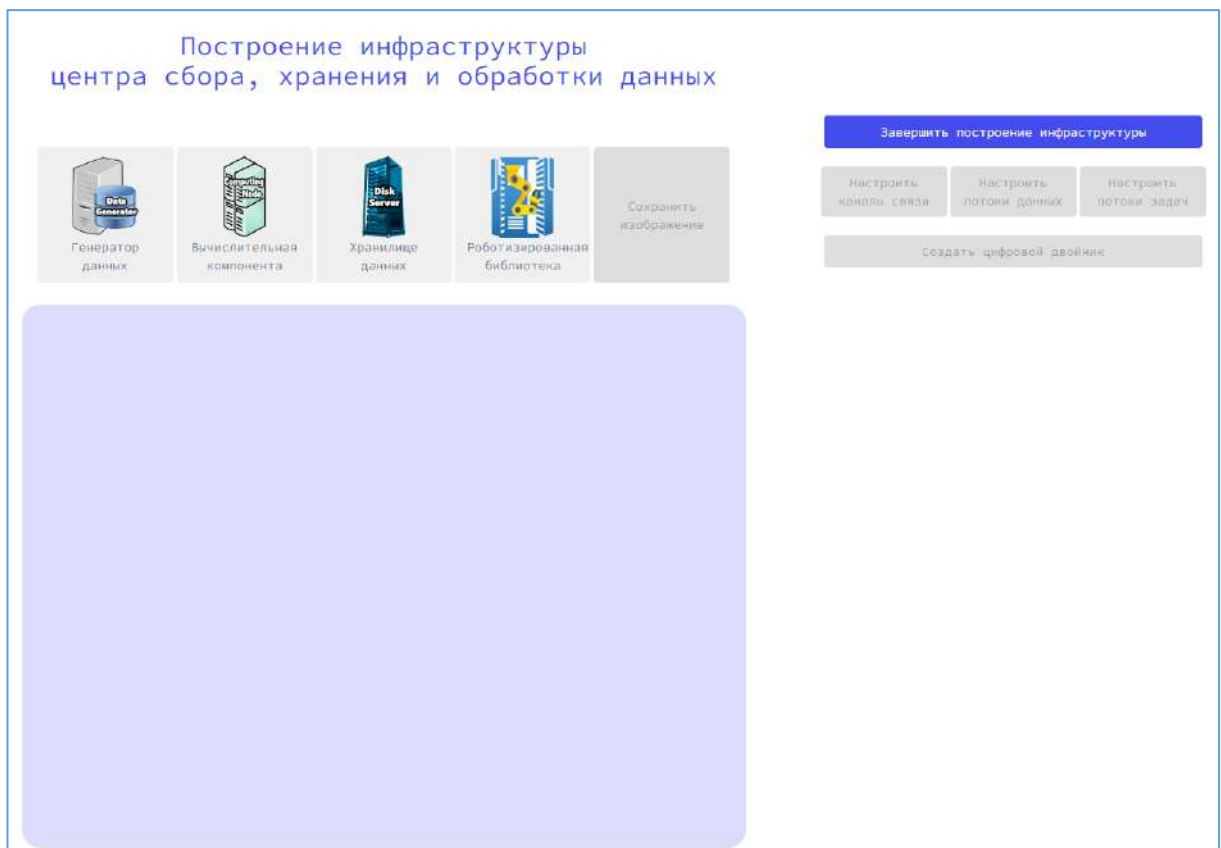


Figure 80. Building the infrastructure of DDCs

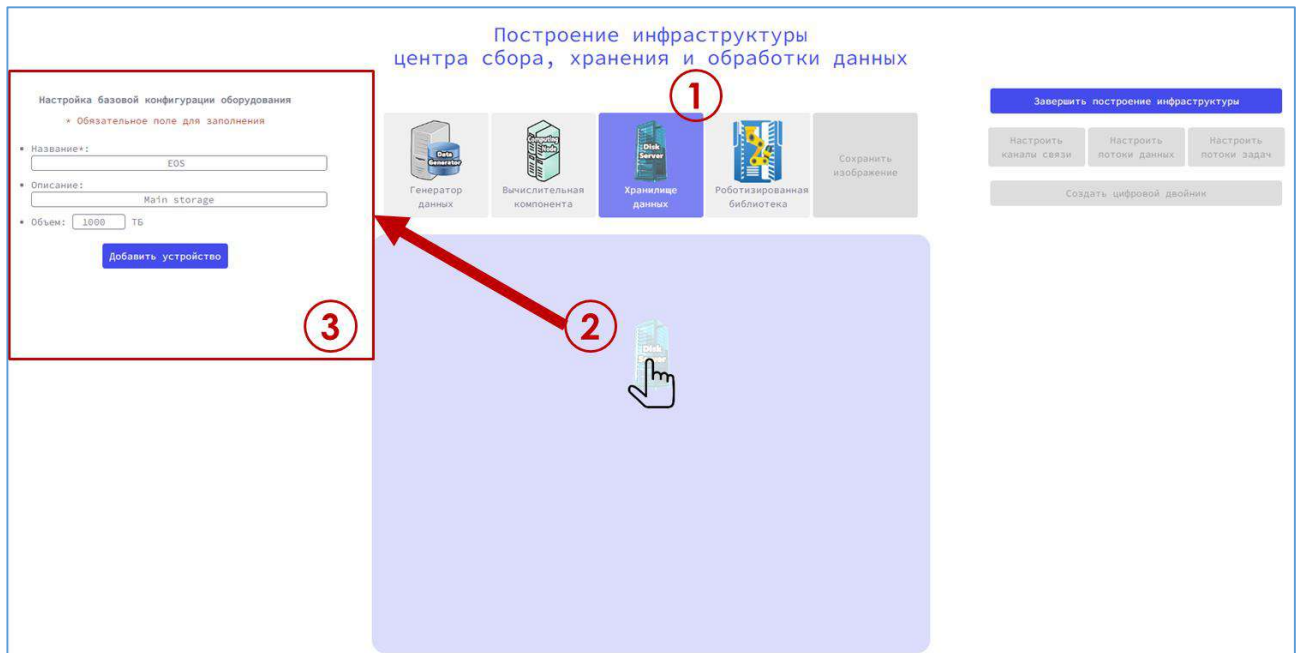


Figure 81. Adding a data storage to the DDC infrastructure and configuring its parameters



Figure 82. Result of building DDC infrastructure

2. The built infrastructure can be edited, for example, to change the location of objects or parameters of the basic configuration. To do this, click on the “Edit Infrastructure” button, then in the drawing area select an object to move or change its parameters (see Fig. 83). All changes should be saved by clicking on the corresponding button.



Figure 83. Moving one of the data stores in the process of editing the DDC infrastructure

3. Once all the necessary objects have been added, the user must configure the communication links between them (see Fig. 84). To configure links, select the objects to be connected between and set the required parameters. User can make changes to existing communication links (see Fig. 87).




Figure 84. Setting up communication links


Построение инфраструктуры центра сбора, хранения и обработки данных



Генератор данных



Вычислительная компонента

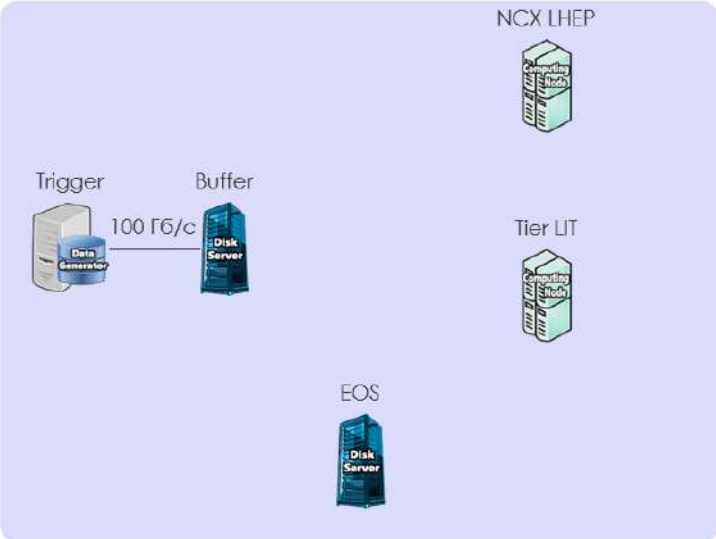


Хранилище данных



Роботизированная библиотека

Сохранить изображение



Редактировать инфраструктуру

Настроить каналы связи
Настроить потоки данных
Настроить потоки задач

Создать цифровой двойник

Настройка каналов связи

* Обязательное поле для заполнения

Выберите существующий канал из списка, чтобы изменить параметры


- Название*:
- Описание:
- Канал от*: до*:
- Пропускная способность: Гб/с

Сохранить изменения


Figure 85. Editing communication links

4. Next, user must to create data flows for processing at the DDC (see Fig. 86) and job flows that will process this data (see Fig. 87). To create flows it is necessary to click on the corresponding button and then fill in the opened form.


Построение инфраструктуры центра сбора, хранения и обработки данных




Генератор данных



Вычислительная компонента

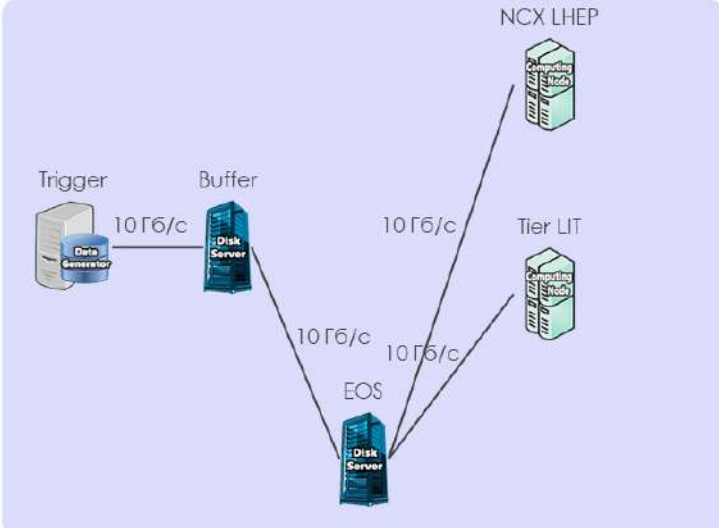


Хранилище данных



Роботизированная библиотека

Сохранить изображение



Редактировать инфраструктуру

Настроить каналы связи
Настроить потоки данных
Настроить потоки задач

Создать цифровой двойник

Настройка потоков данных


* Обязательное поле для заполнения

- Название*:
- Описание:
- Канал передачи данных*:
- Объем данных для передачи: %

Добавить

Figure 86. Setting up data flows


Построение инфраструктуры центра сбора, хранения и обработки данных




Генератор данных



Вычислительная компонента

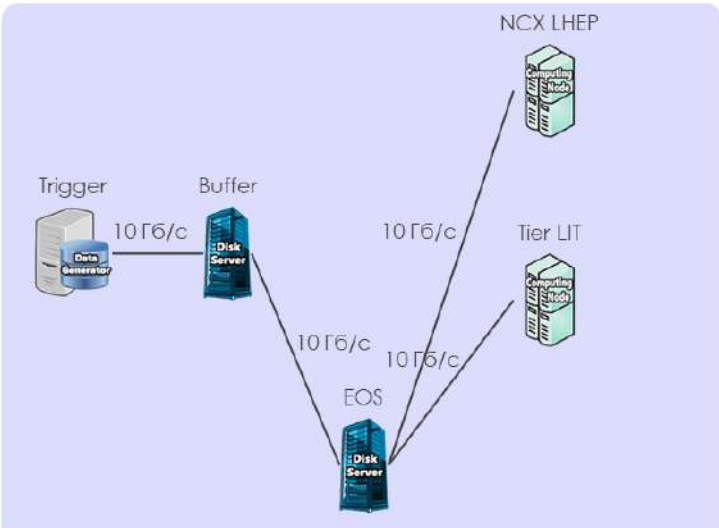


Хранилище данных



Роботизированная библиотека

Сохранить изображение



Редактировать инфраструктуру

Настроить каналы связи
Настроить потоки данных
Настроить потоки задач

Создать цифровой двойник

Настройка потоков задач

* Обязательное поле для заполнения

- Название*:
- Описание:
- Тип обрабатываемых данных*:
- Объем входных данных для задачи*: ГБ
- Объем входных данных (допустимое отклонение): ГБ
- Тип результирующих данных*:
- Объем выходных данных*: ГБ
- Объем выходных данных (допустимое отклонение): ГБ
- Распределение вероятности для генерации данных:
- Среднее время выполнения задачи*: с
- Среднее время выполнения (допустимое отклонение):
- Распределение вероятности для генерации задач:
- Общее количество задач *

Добавить

Figure 87. Setting up job flows

5. The user can edit the DDC infrastructure, basic equipment configuration, change data flow parameters and job flows before moving to the next stage, which is directly creating the DDC.

6. Clicking on the “Create Digital Twin” button will open the window shown in Figure 88. The user needs to add a computational experiment to the DT by clicking on the corresponding button, after which the form shown in Figure 89 will open. The user is prompted to enter the name and description of the computational experiment, modeling and logging parameters. It is possible to add several computational experiments to solve the problem of searching for the required configuration of equipment of a certain type (see Fig. 90).

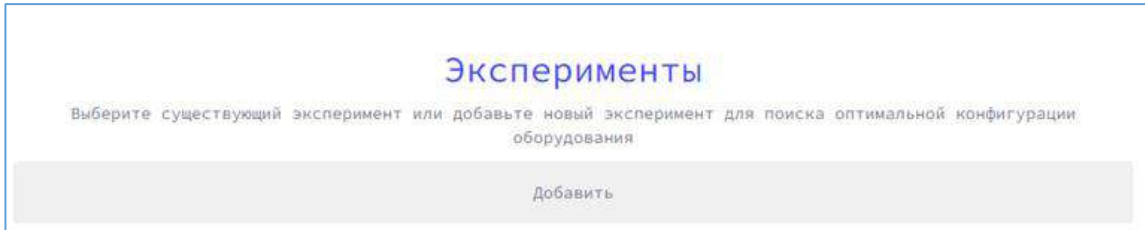


Figure 88. Adding a computational experiment on the DT



Figure 89. Setting up the parameters of the computational experiment on the DT

Эксперименты

Выберите существующий эксперимент или добавьте новый эксперимент для поиска оптимальной конфигурации оборудования

Test 4

Test

Дата создания: 9 марта 2023 г. 15:04

Test 3

Исследование загрузки каналов связи

Дата создания: 7 февраля 2023 г. 10:42

Test 2

Поиск оптимального количества вычислительных ресурсов

Дата создания: 7 февраля 2023 г. 10:38

Test 1

Поиск оптимального количества ресурсов для хранения данных

Дата создания: 7 февраля 2023 г. 10:36

Добавить

Figure 90. List of added computational experiments on the DT

7. Clicking on one of the available computational experiments will open a page with information that reflects all parameters and the basic configuration of the equipment (see Fig. 91). To add new equipment modifications, click on the “Add modification” button. To add and configure events that can occur in the DDC, click on the “Add Event” button.

Информация об эксперименте

Дата создания: 7 февраля 2023 г. 10:36

Название эксперимента
Test 1

Описание эксперимента
Поиск оптимального количества ресурсов для хранения данных.

Параметры моделирования

- Продолжительность работы моделируемой инфраструктуры = 500 ч.
- Ускорение процесса моделирования в 1000 раз.

Параметры логирования

- Объекты моделируемой инфраструктуры
 - Хранилища данных
 - Вычислительные компоненты
 - Каналы связи
- События
 - Генерация данных
 - Потери данных
 - Работа с файлами
 - Генерация, запуск, выполнение задач

Посмотреть результаты

Выбрать другой эксперимент

Базовая конфигурация

Хранилища данных		
название	описание	объем (TB)
trigger	Trigger BMS4	10000,0
buffer	Data reception buffer	5400,0
eoslhep	Main storage LHEP	1000,0
eoslit	Main storage LIT	1000,0
dcach	DP	1000,0

Вычислительные компоненты		
название	описание	Количество ядер
e2lit	LIT t2 farm	500
ncxlhep	LHEP main farm	1200
super	Governor	100

Каналы связи		
название	описание	Пропускная способность (ГБ/с)
row0	trigger - buffer	100,0
row1	buffer - lhep	10,0
row2	buffer - lit	10,0
compute0	lhep - farm lhep	10,0
compute1	lit - governor	10,0
compute2	lit - farm lit	10,0
dataeoslheplit	eoslhep - eoslit	10,0
dataeoslitlhep	eoslit - eoslhep	10,0

Добавить модификацию

Список модификаций

Список событий

Добавить событие

Figure 91. Viewing information about the computational experiment on the DT

8. After setting up the computational experiments, the DT is started. If there are several modifications in a computational experiment (see Fig. 92), the DT for each modification can be started simultaneously.



Figure 92. Information about the computational experiment with a list of available modifications and events occurring in DDC

9. During or at the end of the DT operation, the results are available for the user to view. Clicking on the "View Results" button will open a page where the type of equipment of interest should be selected, after which interactive graphs will be plotted. For example, Figure Figure 93 shows a page with graphs reflecting data volumes in storages, and Figure Figure 94 shows the load on communication links during data transfer. Graphs of distribution of different types of files in storage and usage of computational components are also available for viewing. Interactive graphs imply the possibility to zoom and select data for viewing, i.e. hide the results of one or several modifications (see Fig. Figure 94). The user can save the results of DT operation for any modification in the form of images with graphs.



Figure 93. Viewing the DT results – total amount of data in storages



Figure 94. Viewing the DT results – data link loading

Appendix 3. Certificate of registration of a computer program

РОССИЙСКАЯ ФЕДЕРАЦИЯ



СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2023667305

**Программный комплекс для создания цифровых
двойников распределенных центров сбора, хранения и
обработки данных**

Правообладатель: **Объединенный Институт Ядерных
Исследований (RU)**

Авторы: **Кореньков Владимир Васильевич (RU), Пряхина
Дарья Игоревна (RU), Трофимов Владимир Валентинович
(RU)**

Заявка № **2023665363**
Дата поступления **20 июля 2023 г.**
Дата государственной регистрации
в Реестре программы для ЭВМ **14 августа 2023 г.**

Руководитель Федеральной службы
по интеллектуальной собственности

Ю.С. Зубов



РОССИЙСКАЯ ФЕДЕРАЦИЯ



RU2023667305

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ
ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2023667305
Дата регистрации: 14.08.2023
Номер и дата поступления заявки:
2023665363 20.07.2023
Дата публикации и номер бюллетеня:
14.08.2023 Бюл. № 8
Контактные реквизиты:
Нет

Автор(ы):
Кореньков Владимир Васильевич (RU),
Пряхина Дарья Игоревна (RU),
Трофимов Владимир Валентинович (RU)
Правообладатель(и):
Объединенный Институт Ядерных
Исследований (RU)

Название программы для ЭВМ:

Программный комплекс для создания цифровых двойников распределенных центров сбора, хранения и обработки данных

Реферат:

Программный комплекс предназначен для создания цифровых двойников (ЦД) распределенных центров сбора, хранения и обработки данных (РЦОД). Созданные ЦД могут применяться для решения задач построения, совершенствования и развития РЦОД. Основным компонентом программного комплекса является алгоритм, реализующий моделирование распределенных центров с учетом характеристик потоков данных и задач для хранения и обработки, а также вероятностей появления изменений в процессах, происходящих в РЦОД. В программе также реализованы процессы получения информации о РЦОД (архитектура, параметры оборудования, характеристики потоков данных и задач, события, происходящих в РЦОД, сценарии масштабирования системы), структурирования и сохранения результатов работы ЦД. Для пользователя разработана возможность графического построения инфраструктуры РЦОД, запуска ЦД и визуализации результатов его работы. Тип ЭВМ: IBM PC-совмест. ПК на базе процессора Intel/AMD; ОС: Windows 7 и выше; Linux.

Язык программирования:

Python 3.7 с использованием библиотек argparse, json, matplotlib, numpy, plotly, psycopg2, scipy и фреймворка Django; JavaScript, SQL

Объем программы для ЭВМ:

133,14 МБ

Appendix 4. Letters on the application of the results of the PhD thesis research

ПИСЬМО

о применении результатов диссертационной работы Пряжиной Д. И.
«Цифровые двойники для решения задач управления и развития распределенных
центров сбора, хранения и обработки данных»
в эксперименте VM@N проекта NICA

В диссертации на соискание ученой степени кандидата технических наук Пряжиной Д. И. «Цифровые двойники для решения задач управления и развития распределенных центров сбора, хранения и обработки данных» разработан метод построения и использования цифровых двойников распределенных центров сбора, хранения и обработки данных. Актуальность работы подтверждается необходимостью создания цифровых двойников для проектирования аппаратных решений, оценки эффективности функционирования, а также поиска проблемных точек сложных систем сбора, хранения и обработки данных, которыми являются крупные распределенные вычислительные инфраструктуры современных научных проектов.

Предложенный в диссертационной работе подход направлен на моделирование распределенных вычислительных систем, оперирующих большими объемами данных. Программа моделирования, которая является основным элементом цифрового двойника, учитывает характеристики потоков данных и задач, вероятности сбоя и изменения в производительности используемой аппаратной части распределенных систем обработки и хранения данных. Важным результатом диссертационной работы является специальное программное обеспечение для создания цифровых двойников, которое может применяться для широкого класса задач в области проектирования, построения и развития распределенных систем, в том числе с целью выбора наиболее целесообразных вариантов из множества рассматриваемых конфигураций аппаратных платформ.

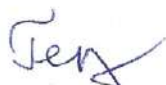
Достоверность подхода и выводов диссертационного исследования подтверждается практикой применения разработанных методов при анализе и прогнозировании работы вычислительной инфраструктуры крупного эксперимента в области физики высоких энергий. Верификация компонентов программного обеспечения проходила в процессе создания цифрового двойника вычислительной инфраструктуры эксперимента VM@N проекта NICA Объединенного института ядерных исследований. Результаты, полученные разработанной в диссертации моделью, показали хорошую согласованность с практическими значениями.

Реализованный метод построения цифрового двойника позволил как описать вычислительную инфраструктуру эксперимента VM@N, учитывая реальные потоки данных и задач, так и применить его для оценки характеристик работы имеющихся

центров обработки при решении задач эксперимента на выбранной инфраструктуре. Результаты работы цифрового двойника, построенного с помощью разработанного специального программного обеспечения, позволили оценить требуемое количество ресурсов для обработки данных с учетом планируемых параметров потоков данных будущих сеансов эксперимента VM@N, проведена оценка времени, необходимого для полной обработки экспериментальных данных по окончании сеанса эксперимента, как с учетом, так и без учета дополнительной обработки имеющихся моделированных данных. В дальнейшем специальное программное обеспечение планируется применять для формирования требований по развитию вычислительной инфраструктуры эксперимента VM@N.

Пряхина Д. И. регулярно представляет для обсуждения и формирования дальнейшей стратегии текущие результаты данной работы на коллаборационных совещаниях эксперимента VM@N.

Выражаю свою поддержку диссертационной работе Д. И. Пряхиной.



к.т.н., Герценбергер Константин Викторович
координатор разработки программного обеспечения
эксперимента VM@N проекта NICA,
начальник группы математического и программного обеспечения
научно-экспериментального отдела физики столкновений
тяжелых ионов на комплексе NICA,
Лаборатория физики высоких энергий им. В.И. Векслера и А.М. Балдина,
Объединенный институт ядерных исследований

« 12 » 02 2024 г.

ПИСЬМО

о применении результатов диссертационной работы Пряхиной Д.И. «Цифровые двойники для решения задач управления и развития распределенных центров сбора, хранения и обработки данных» в эксперименте SPD комплекса NICA

В работах, легших в основу диссертации на соискание ученой степени кандидата технических наук Пряхиной Д.И. «Цифровые двойники для решения задач управления и развития распределенных центров сбора, хранения и обработки данных», был разработан и реализован программный инструментарий для создания цифровых двойников распределенных центров сбора, хранения и обработки данных. Распределенная обработка играет ключевую роль в получении научных результатов крупных экспериментов класса «мегасайенс». Хорошим примером являются эксперименты на ускорительном комплексе NICA. В рамках проектирования физической установки, а так же в процессе проводимых исследований эксперименты производят большой объем данных, для хранения и своевременной обработки которых используются крупные вычислительные системы. К таким комбинированным системам предъявляются достаточно высокие требования по производительности при оптимальных затратах, поэтому работа Пряхиной Д.И. по созданию цифровых двойников, которые помогают обеспечить качественное проектирование, постоянное совершенствование и масштабирование систем хранения и обработки данных, является очень актуальной.

Разработанный в диссертации новый метод построения и использования цифровых двойников сложных систем отличается возможностью моделировать процессы обработки данных, учитывая характеристики потоков данных и вычислительных задач, вероятностей сбоев, отказов и изменений в производительности различных составляющих распределенной системы. Эта особенность является существенным преимуществом работы, так как позволяет применять разработку при принятии решений по улучшению технических характеристик сложных вычислительных систем.

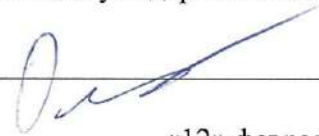
Экспериментальная установка SPD комплекса NICA находится в стадии создания, одновременно реализуется и система первичной обработки данных эксперимент: SPD online filter. Проектирование специализированной вычислительной системы для высокопропускной обработки огромного объема, включает в себя анализ архитектуры и прогнозирование затрат на оборудование, поэтому разработанный Пряхиной Д.И. программный комплекс используется для создания цифрового двойника SPD Online filter. Главная задача цифрового двойника — моделирование подсистем с разными вариантами параметров оборудования, потоков данных и типов обработки. Результаты работы цифрового двойника на данный момент позволили первоначально оценить требуемые параметры оборудования для специализированной системы предварительной обработки с учетом планируемых характеристик потоков данных эксперимента SPD комплекса NICA. Применение результатов диссертационного исследования Пряхиной Д.И. в обозримом будущем позволит обеспечить качественное развитие и поддержку вычислительной инфраструктуры эксперимента SPD.

Полученные результаты первого варианта цифрового двойника и планы развития данного направления получили одобрение на совещании коллаборации эксперимента SPD в

2023 году (VI SPD Collaboration Meeting and Workshop on Information Technology in Natural Sciences).

Выражаю поддержку диссертационной работе Пряхиной Д.И.

к.т.н., Олейник Данила Анатольевич
заместитель координатора разработки программного обеспечения
эксперимента SPD комплекса NICA,
старший научный сотрудник научно-технического отдела внешних коммуникаций и
распределенных информационных систем,
Лаборатория информационных технологий им. М.Г. Мещерякова,
Объединенный институт ядерных исследований



«12» февраля 2024 г.

Appendix 5. Act of implementing the results of the PhD thesis research in the educational process

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«**Университет «Дубна»**»

(государственный университет «Дубна»)



УТВЕРЖДАЮ

И.о. проректора по учебной работе

Государственного университета «Дубна»

О.А. Крейдер

«12» 02 2024 г.



АКТ

о внедрении результатов диссертационной работы Пряхиной Д.И. «Цифровые двойники для решения задач управления и развития распределенных центров сбора, хранения и обработки данных» в Институте системного анализа и управления (ИСАУ) Федерального государственного бюджетного образовательного учреждения высшего образования «Университет «Дубна»»

Комиссия в составе:

д.т.н., проф. Е.Н. Черемисина, научный руководитель ИСАУ, зав. каф. системного анализа и управления,

к.т.н., доц. Е.Ю. Кирпичева, директор ИСАУ,

к.ф.-м.н., доц. Н.А. Токарева, зав. каф. информационных технологий,

к.т.н., О.Ю. Тятюшкина, доцент каф. системного анализа и управления,

П.П. Сычев, доцент каф. распределенных информационно-вычислительных систем

составила настоящий акт о внедрении результатов диссертационного исследования Пряхиной Д.И. в учебный процесс Государственного университета «Дубна». Развитый в диссертации метод построения и использования цифровых двойников распределенных центров сбора, хранения и обработки данных используется при проведении лекций и семинарских занятий для студентов магистратуры по направлению 27.04.03 Системный анализ и управление по профилю «Цифровые платформы и аналитика больших данных» в курсе «Распределенные вычисления и облачные технологии».

Использование в учебном процессе методических рекомендаций и выводов диссертации Пряхиной Д.И. в части подхода к разработке модели распределенных вычислительных систем с учетом характеристик потоков задач и данных для хранения и обработки, а также вероятностей появления изменений в процессах, происходящих в таких системах, позволяет повысить уровень профессиональных компетенций студентов в решении задач производственной и технологической деятельности с учетом современных достижений науки и техники.

Черемисина Е.Н.

Научный руководитель ИСАУ,
зав. каф. системного анализа и управления,
проф., д.т.н.



Кирпичева Е.Ю.

Директор ИСАУ,
доц., к.т.н.



Токарева Н.А.

зав. каф. информационных технологий,
доц., к.ф.-м.н.



Тятюшкина О.Ю.

доцент каф. системного анализа и управления,
к.т.н.



Сычев П.П.

доцент каф. распределенных информационно-вычислительных систем

