

Министерство науки и высшего образования Российской Федерации
 федеральное государственное автономное
 образовательное учреждение высшего образования
 «Национальный исследовательский Томский политехнический университет» (ТПУ)

School **School of Computer Science & Robotics**
 Academic program **09.04.01 Computer Science and Engineering**
 Division **Division for Information Technology**

MASTER'S GRADUATION THESIS

Topic of research work
Development of tools for building simulation models of cloud computing infrastructures and software-defined distributed networks

UDC 004.415.2:004.94:004.75:004.451

Student

Group	Full name	Signature	Date
8BM01	Mouhamad Ibrahim		

Scientific supervisor

Position	Full name	Academic degree, academic rank	Signature	Date
Associate Professor	Botygin I.A.	PhD		

SECTION ADVISERS:

Section «**Financial Management, Resource Efficiency, and Resource Saving**»

Position	Full name	Academic degree, academic rank	Signature	Date
Full Professor	Zhironkin S. A.	PhD		

Section «**Social Responsibility**»

Position	Full name	Academic degree, academic rank	Signature	Date
Full Professor	Fedorenko O. Y.	PhD		

ADMITTED TO DEFENSE:

Director of program	Full name	Academic degree, academic rank	Signature	Date
Internet application development	Kochegurova E.A.	PhD		

LEARNING OUTCOMES

Expected learning outcomes

Code competencies	Learning outcome (a graduate should be ready)
Universal competencies	
UK(U)-1	Able to critically analyze problematic situations using a systematic approach, to develop a strategy of action
UK(U)-2	Able to manage a project through all stages of its life cycle
UK(U)-3	Able to organize and manage a team, develop a team strategy to reach the set target
UK(U)-4	Able to use modern communication technologies, also in foreign language(s), for academic and professional interactions
UK(U)-5	Able to analyze and take into account the diversity of cultures in the process of intercultural interaction
UK(U)-6	Able to identify and implement priorities of their own activities and ways to improve them on the basis of self-assessment
General professional competencies	
GPC(U)-1	Able to independently acquire, develop and apply mathematical, natural-science, socio-economic and professional knowledge to solve non-standard tasks, including in a new or unfamiliar environment and in an interdisciplinary context
GPC(U)-2	Able to develop original algorithms and software tools, including those using modern intellectual technologies, to solve professional tasks
GPC(U)-3	Capable of analyzing professional information, summarizing, structuring, presenting in analytical reviews with substantiated conclusions and recommendations
GPC(U)-4	Capable of applying new scientific principles and research methods in practice
GPC(U)-5	Capable of developing and upgrading software and hardware for information and automated systems
GPC(U)-6	Capable of developing components of hardware-software complexes for information processing and computer-aided design
GPC(U)-7	Capable of adapting foreign data processing and CAD systems to the needs of domestic enterprises
GPC(U)-8	Capable of managing effectively the development of software tools and designs

Professional competencies	
PC(U)-1	Capable of designing and administering database management systems
PC(U)-2	Capable of designing complex user interfaces
PC(U)-3	Capable of managing processes and projects to create (modify) information resources
PC(U)-4	Capable of managing the development of complex projects at all stages and phases of work
PC(U)-5	Capable of designing and organizing the learning process of educational programs using modern educational technologies

Министерство науки и высшего образования Российской Федерации
 федеральное государственное автономное
 образовательное учреждение высшего образования
 «Национальный исследовательский Томский политехнический университет» (ТПУ)

School **School of Computer Science & Robotics**
 Academic program **09.04.01 Computer Science and Engineering**
 Division **Division for Information Technology**

APPROVED BY:
 Director of program
 _____ Kocheurova E.A.
 « _____ » _____ 2022 г.

**ASSIGNMENT
for the Graduation Thesis completion**

In the form:

Master thesis

For a student:

Group	Full name
8BM01	Mouhamad Ibrahim

Topic of research work:

Development of tools for building simulation models of cloud computing infrastructures and software-defined distributed networks	
Approved by the rector's order (date, ID)	No. 96-19/c of 05.04.2022

Deadline for completion of Master's Graduation Thesis:	01.06.2022
--	------------

TERMS OF REFERENCE:

<p>Initial data for research work</p> <p><i>(name of the object of study or design; productivity or load; mode of operation (continuous, periodic, cyclical, etc.); type of raw material or product material; requirements for the product, product or process; special requirements for the specific functioning (operation) of the object or product in terms of operational safety, environmental impact, energy costs; economic analysis, etc.).</i></p>	<p>Development of a toolkit based on freely available software for building simulation models of modern cloud computing infrastructures in software-defined distributed networks.</p>
---	---

<p>List of the issues to be investigated, designed and developed</p> <p><i>(analytical review of the literature in order to find out the achievements of world engineering science in the field in question; statement of the research, design, construction task; content of the research, design, construction procedure; discussion of the results of the work performed; names of additional sections to be developed; conclusion on the work).</i></p>	<ol style="list-style-type: none"> 1. Modeling and analysis of scalable cloud computing environments. 2. Designing generalized cloud computing simulation system in software-defined networks. 3. Designing a generalized web interface for simulation experiments. 4. Software implementation and results. 5. Financial management, resource efficiency, and resource saving. 5. Social responsibility.
--	--

Advisors to the sections of the Master's Graduation Thesis

Section	Advisor
Financial Management, Resource Efficiency, and Resource Saving	Full Professor Zhironkin S. A.
Social Responsibility	Full Professor Fedorenko O. Y.
English Language	Associate Professor Didenko A.V.

Date of issuance of the assignment for Master's Graduation Thesis completion according to the linear schedule	01.03.2022
--	------------

The assignment was given by the scientific supervisor:

Position	Full name	Academic degree, academic rank	Signature	Date
Associate Professor	Botygin I.A.	PhD		

The assignment was accepted for execution by the student:

Group	Full name	Signature	Date
8BM01	Mouhamad Ibrahim		

Министерство науки и высшего образования Российской Федерации
 федеральное государственное автономное
 образовательное учреждение высшего образования
 «Национальный исследовательский Томский политехнический университет» (ТПУ)

School **School of Computer Science & Robotics**
 Academic program **09.04.01 Computer Science and Engineering**
 Division **Division for Information Technology**

Form of presenting the work:

Master's Thesis

**SCHEDULED ASSESSMENT CALENDAR
for the Master's Graduation Thesis completion**

Student:

Group	Full name
8BM01	Mouhamad Ibrahim

Topic of research work:

Development of tools for building simulation models of cloud computing infrastructure and software-defined distributed networks
--

Deadline for completion of Master's Graduation Thesis:	01.06.2022
--	------------

Assessment date	Title of the section (module) / type of work (research)	Maximum score of a section (module)
01.03.2022	Drawing up and approving the terms of reference	10
10.03.2022	Selection and study of materials on the topic	10
30.03.2022	Research of subject area	15
20.04.2022	Conducting experiments	25
15.05.2022	Analysis and description of results	25
01.06.2022	Preparing for thesis defense	15

**COMPILED BY:
Scientific supervisor**

Position	Full name	Academic degree, academic rank	Signature	Date
Associate Professor	Botygin I.A.	PhD		

**AGREED BY:
Director of program**

Position	Full name	Academic degree, academic rank	Signature	Date
Associate Professor	Kochegurova E.A.	PhD		

Group	Full name	Signature	Date
8BM01	Mouhamad Ibrahim		

**TASK FOR
«FINANCIAL MANAGEMENT, RESOURCE EFFICIENCY, AND RESOURCE SAVING»
SECTION**

To student:

Group	Full name
8BM01	Mouhamad Ibrahim

School	School of Computer Science & Robotics	Department	Division for Information Technology
Educational level	Master Degree Program	Academic program	Computer science and engineering

Initial data to «Financial management, resource efficiency, and resource saving » section:

1. <i>The cost of scientific research resources: material, technical, energy, financial, information, and human</i>	Supervisor's salary – 35111,5 rubles. Engineer's salary – 22695 rubles. Electricity tariff – 6,59 rubles per 1 kWh
2. <i>Norms and standards of resource expenditure</i>	
3. <i>The taxation system used, the rates of taxes, deductions, discounting, and lending</i>	Coefficient of contributions to extra-budgetary funds – 30%.

List of issues to be researched, designed, and developed:

1. <i>Evaluation of commercial and innovative potential</i>	Definition of the purpose of the research project, description of potential consumers of the project, and the results of its implementation. Determination of the target market and its segmentation. Performing a SWOT analysis of the project.
2. <i>Development of the charter of the scientific and technical project</i>	
3. <i>Planning of the management process: structure and schedule, budget, risks, and procurement organization</i>	Organization and planning of work. Cost estimates calculation of the project implementation.
4. <i>Determination of resource, financial, and economic efficiency</i>	Conducting an assessment of the social effectiveness of the project and publishing it as an open-source project

List of graphic material (with the exact indication of the required drawings):

1. *Market segmentation*
2. *Assessment of the competitiveness of technical solutions*
3. *Assessment of the project's readiness for commercialization*
4. *SWOT Matrix*
5. *Schedule and budget of the research*
6. *Assessment of resource, financial and economic efficiency of the research*

Date of task obtaining

The task was given by the adviser:

Position	Full name	Academic degree, academic rank	Signature	Date
Full Professor	Zhironkin S. A.	PhD		28.02.2022

The task was accepted by the student:

Group	Full name	Signature	Date
8BM01	Mouhamad Ibrahim		28.02.2022

TASK FOR «SOCIAL RESPONSIBILITY» SECTION

To student:

Group 8BM01		Full name Mouhamad Ibrahim	
School	School of Computer Science & Robotics	Department	Division for Information Technology
Educational level	Master Degree Program	Academic program	Computer science and engineering

Topic of research work:

Разработка инструментальных средств для построения имитационных моделей инфраструктур облачных вычислений и программно-определяемых распределённых сетей
Development of tools for building simulation models of cloud computing infrastructures and software-defined distributed networks

Initial data to «Social responsibility» section:

Introduction

- Characteristics of the research object (substance, material, device, algorithm, methodology) and its application areas.
- Description of the working area (workplace) during the development of the design solution/during operation

Research object: a platform for simulating the infrastructure of cloud environments and software-defined distributed networks. The platform tools allow describing all the basic components of the projected data center within a virtual system. This makes it possible to simplify the model integration of hardware and software and various applications within a single software-controlled environment and to investigate the performance of all its technological control processes. At the same time, quantitative operational parameters of system components and estimates of the execution time of management processes are automatically generated, depending on the configuration of the infrastructure.

Working area: office space containing a desk, chair, and PC;

Room dimensions: 6*4,92 m;

Heating: water central heating systems;

Ventilation: General exchange ventilation.

List of issues to be researched, designed, and developed:

1. Legal and organizational issues for safety support during the project development:

- special (characteristic for the operation of the research object, the projected work area) legal norms of labor legislation;
- organizational arrangements for the layout of the work area.

Labor Code of the Russian Federation of 30.12.2001 No. 197-FZ (ed. of 09.03.2021);

GOST 12.0.003-2015 «Occupational safety standards system. Dangerous and harmful working factors. Classification»;

GOST 12.2.032-78 «Operator's location in a sitting position. General ergonomic requirements»;

TOI R-45-084-01 «Standard instructions on labor protection when working on a personal computer»;

SanPiN 1.2.3685-21 «Hygienic standards and requirements for ensuring the safety and (or) harmlessness of environmental factors for humans»;

SP 52.13330.2016 «Natural and artificial lighting. Updated version of SNiP 23-05-95»;

GOST 12.1.003-83 «Occupational safety standards system. Noise. General safety requirements»;

	<p>GOST 12.1.002-84 «Electric fields of industrial frequency»;</p> <p>GOST 12.1.038-82 «Occupational safety standards system. Electric safety. Maximum permissible values of pickp voltages and currents»;</p> <p>GOST 12.1.004-91 «Occupational safety standards system. Fire safety. General requirements»;</p> <p>GOST 17.4.3.04-85 « Nature protection. Soils. General requirements for contamination control and protection».</p>
<p>2. Industrial safety during the development of the designed solution:</p> <ul style="list-style-type: none"> – Analysis of identified harmful and hazardous production factors – Calculation of the level of a dangerous or harmful production factor 	<p>Analysis of identified harmful factors:</p> <ol style="list-style-type: none"> 1. deviation of microclimate indicators; 2. insufficient illumination of the working area; 3. exceeding the noise level. <p>Analysis of identified hazards:</p> <ol style="list-style-type: none"> 1. increased magnetic field strength; 2. electric shock. <p>Means of collective protection:</p> <ol style="list-style-type: none"> 1. protective coatings against electromagnetic radiation; 2. protective earthing devices <p>Calculation: calculation of artificial lighting system</p>
<p>3. Environmental safety during the development of the designed solution</p>	<p>Impact on the residential area: absent</p> <p>Impact on the lithosphere occurs during disposal:</p> <ol style="list-style-type: none"> 1. used devices; 2. faulty electronics; 3. fluorescent lamps; 4. waste paper. <p>Impact on the atmosphere and hydrosphere: absent</p>
<p>4. Safety in emergency situations during the development of the designed solution</p>	<p>Possible emergencies: fires, thunderstorms, hurricanes, landslides.</p> <p>The most typical emergency: fire</p>
<p>Date of task obtaining</p>	

The task was given by the adviser:

Position	Full name	Academic degree, academic rank	Signature	Date
Full Professor	Fedorenko O. Y.	PhD		

The task was accepted by the student:

Group	Full name	Signature	Date
8BM01	Mouhamad Ibrahim		

Summary

The Master's Graduation Thesis contains: 135 pages, 56 figures, 26 tables, 49 sources, and 1 appendix.

Keywords: cloud computing, software-defined network, modeling tools, CloudSim Plus, data visualization, web application.

The study object is cloud computing infrastructure in software-defined networks.

The purpose of the work is to provide a generalized and extensible simulation environment that enables seamless simulation and experimentation with new cloud computing infrastructures and application services.

Research objectives include:

- review the research on SDN architecture and its main components.
- defining the tasks for the formation of multi-stage programs for modeling cloud computing infrastructure in software-defined networks and integrating its implementation with the CloudSim Plus platform.
- development of a web interface for simulation experiments in the modified CloudSim Plus platform and a graphical display of simulation results.

In the process of the study, the software-defined networking key components were analyzed, as well as popular cloud computing simulation tools.

As a result of the study, a simulation tool was developed for modeling cloud computing infrastructure in software-defined networks, as well as a web interface for simulation experiments and visualizing the results using graphical charts.

The area of application of the research results is research institutes and companies whose main activity is to conduct research into energy efficiency, and high-speed data transfer within and between data centers with horizontal and vertical scalability.

The simulation tool was developed using Java programming language and based on CloudSim Plus. The web interface was developed using Java programming

language, Spring Framework, and React JavaScript library for building user interfaces. Nivo data visualization library was used to visualize the simulation results.

List of abbreviations

DC	Data Center
VM	Virtual Machine
CP	Cloud Provider
IaaS	Infrastructure as a Service
MPLS	Multiprotocol Label Switching
VPN	Virtual Private Network
WAN	Wide Area Network
LAN	Local Area Network
ONF	Open Networking Foundation
SDN	Software-Defined Network
A-CPI	Application-controller plane interface
D-CPI	Data-controller plane interface
STP	Spanning Tree Protocol
ISP	Internet Service Provider
OSI	Open Systems Interconnection
API	Application Programming Interface
GUI	Graphical User Interface
REST API	Representational State Transfer API

Table of contents

Introduction	17
1. Modeling and analysis of scalable cloud computing environments	19
1.1. Principles of cloud computing and software-defined approach to network management	19
1.1.1. Analysis of cloud computing technology	19
1.1.2. Analysis of SDN solutions for cloud networking	21
1.2. Analysis of simulation problems in complex systems.....	26
1.3. Simulation modeling tools	29
1.4. Overview of specialized cloud infrastructure simulation systems	31
1.5. Cloud simulators classification approaches	35
1.6. Direction and objectives of the study.....	40
2. Designing generalized cloud computing simulation system in software-defined networks	42
2.1. Analysis of the SDN core components	42
2.2. Analysis of the network simulation models in CloudSim Plus	45
2.3. Object-oriented design of generalized cloud computing simulation system in software-defined networks.....	47
2.3.1. Network Elements	47
2.3.2. Control Tables	49
2.3.3. Physical and Virtual Networks	51
2.3.4. Controllers	53
2.3.5. Switch Power Modeling	60
2.3.6. CloudSim Plus Extension.....	61
2.4. Conclusion.....	65
3. Designing a generalized web interface for simulation experiments.....	66
3.1. General architecture	66
3.2. Core component.....	67
3.3. Engine component	67
3.4. Backend component	69
3.5. Frontend component.....	71
5. Finance Management Resource Efficiency and Saving	92
5.1. Pre-project analysis.....	92
5.1.1. Potential consumers of research results	92
5.1.2. Analysis of competitive solutions	93
5.1.3. SWOT analysis	96
5.1.4. Assessment of the project readiness for commercialization	98
5.1.5. Methods for commercialization of scientific and technical research results	99
5.2. Project initiation.....	100

5.2.1.	Objectives and results of the project	100
5.2.2.	Organizational structure of the project.....	101
5.3.	Planning of scientific and technical project management	102
5.3.1.	Hierarchical structure of the project work	102
5.3.2.	Project plan.....	102
5.3.3.	Budget of the scientific research	104
5.4.	Evaluation of the economic efficiency of the project.....	109
5.5.	Conclusion.....	110
6.	Social responsibility.....	111
	Introduction	111
6.1.	Legal and organizational issues for safety support during the project development	111
6.1.1.	Special legal norms of labor legislation	111
6.1.2.	Ergonomic requirements for the correct location and layout of the work area	112
6.2.	Industrial safety.....	114
6.2.1.	Analysis of hazardous and harmful production factors	115
6.2.2.	Justification of measures to reduce the levels of exposure to dangerous and harmful factors on the researcher	118
6.3.	Environmental safety	122
6.4.	Safety in emergency situations	123
6.4.1.	Analysis of probable emergencies that may occur during the development and operation of the software.....	123
6.4.2.	Justification of emergency prevention measures and development of an action procedure in case of an emergency	123
6.5.	Conclusion.....	124
	Conclusion.....	125
	List of student publications	126
	References	127
	Appendix A	132

Introduction

In today's environment, there is a strong trend toward the need to improve the technical and economic performance of complex structured reconfigurable systems, which include complex information and communication systems, cloud computing, and reconfigurable software-defined networks, among others. The main characteristic of such systems is the fundamentally modular reconfigurable infrastructure. In this case, the multi-criteria task of optimizing the functioning of complex structured reconfigurable systems is to determine the optimal values of structural parameters, as well as the parameters of functional modules that ensure the achievement of given values of the relevant quality criteria.

It should be noted that complex structured systems are virtually impossible to describe mathematically using analytical methods due to the extreme complexity of circulating random processes and quantities. In the practice of researching and designing such complex systems in various application areas, the simulation modeling apparatus is widely used.

The computational complexity of simulation models of complex structured systems should be noted. In terms of the machine resources required for their implementation, which limits the possibility of their effective use in real-time decision-making. This requires the development of new approaches to the structural organization of simulation models that provide a significant reduction in the time and memory resources required for their practical use. Given the large number of typical processes implemented within complexly structured systems, it seems appropriate to assign a limited number of reconfigurable simulation modules to each type.

In this regard, the relevance of the topic of this study is dictated by the need for further development of simulation modeling apparatus based on new approaches to the structural organization of simulation models of complex structured systems, for example, implementing the modular principle of construction based on reconfigurable modules of standard processes. The specified circumstance provides an opportunity to use the simulation modeling apparatus in

the operational conditions of management decision-making. The main goal of this project is therefore to provide a generalized and extensible simulation environment that enables seamless simulation and experimentation with new cloud computing infrastructures and application services.

1. Modeling and analysis of scalable cloud computing environments

1.1. Principles of cloud computing and software-defined approach to network management

1.1.1. Analysis of cloud computing technology

Non-stationary processes whose features fluctuate over time are common in systems serving user requests in continuous mode. The fact that users create an uneven load on the computing core at different times (morning, afternoon, evening, night) causes nonstationary user load. Cloud computing is an excellent example of such a system, as it runs continuously. The performance of a cloud application is directly proportional to the number of compute nodes (real or virtual) running its instances, thus applications running in the cloud have the property of horizontal scalability. A functioning clone of an application that handles tasks from a shared queue is referred to as a cloud application instance.

A common approach to cloud computing is to combine all of a data center computing resources into a single loop of computing resources known as a computing cloud. The structure and logical interconnection of physical nodes in a computing cloud, are of interest to the system administrators and typically vary over time. Multiple cloud applications are typically executed at the same time within a single compute cloud that integrates all hardware resources of a data center. In this approach, each physical server is represented in the compute cloud topology as a collection of virtual compute nodes (virtual machines) that can run instances of cloud applications. This level of resource virtualization enables the load to be efficiently redistributed across cloud nodes, for example, by assigning computing power based on the number of user requests being processed.

The virtual machines (VMs) are part of the cloud service and are controlled by a cloud system, which is software that manages the physical and virtual resources of the computing cloud. The cloud system, in particular, may automatically shut down virtual machines that are not in use, decommission idle hardware resources, and create fresh instances of the cloud application, boosting the performance of the computing cloud.

Unless intentional limits are specified, the following rules apply to any structural modifications in the computational cloud:

1. At any given time, a virtual machine is either idle or related to only one cloud application.
2. Any unoccupied virtual machines, including those available as free resources, can be provisioned for any cloud application on any physical host.
3. A physical node can host as many virtual computers as the physical restrictions allow.
4. Within a single virtual machine, only one instance of the cloud application is allowed to run.

Software suites for developing cloud systems have emerged as the capabilities of specialized software have grown. For instance, VMware VSphere [1], OpenNebula [2], HPE Eucalyptus [3], and other software packages. This type of software combines an administrative interface with system tools to operate popular virtual machine hypervisors such as VMware ESXi [4], Xen [5], KVM [6], Microsoft Hyper-V [7], and others. Due to these software packages, the creation of cloud systems has become a typical activity, and operator functions have been partially automated.

With the emergence of such software packages, the level of automation in the creation and management of cloud systems has improved substantially. Cloud providers are companies that produce and integrate these software products into unified computing platforms. Amazon EC2 (part of Amazon Web Services) [8], Microsoft Azure [9], and other IaaS providers are examples of such platforms. This type of technology offers automated provisioning of running virtual machines, hardware resource management, fault tolerance, and availability of all cloud computing system components.

Previously, the data center was usually owned and operated by a single company that also owned and needed access to the machines. Using existing tools, the problem of interconnection between the various branches and data centers was

readily overcome. Users in branch offices or campus networks can connect to applications hosted on servers in the data center via a standard Wide Area Network (WAN). To protect and sustain reliable communications, dedicated Multiprotocol Label Switching (MPLS) lines are typically employed.

Cloud providers build multi-tenant data center environments by allowing other users access to their networks. Obviously, in such situations, separating possibly thousands of tenants whose resources must be distributed arbitrarily over different virtual machines while providing them all with private Internet access to their slices of the cloud is required [12]. Traditional tools like MPLS and VPN, on the other hand, do not work in multi-tenant data center environments [12].

1.1.2. Analysis of SDN solutions for cloud networking

The goal of Software-Defined Networking (SDN) is to provide open interfaces that allow the software to control the connection given by a collection of network resources, as well as the flow of network traffic via them, as well as possible traffic inspection and change in the network. Typical network elements, such as switches, consist of hardware and software layers that are roughly coupled. Also, these devices can have vendor-specific protocols. Depending on those facts, protocols or policy changes require manual complex reconfiguration of the entire network hardware.

Inflexibility is another disadvantage of traditional networking. Modern user applications necessitate dynamic load-balancing capabilities in networks, which are difficult to implement in traditional networks due to their static nature. Because of these issues, traditional networking performance is unsatisfactory. SDN overcomes the major setbacks of traditional approaches to network management by enforcing the following principles:

1. Decoupling of controller and data planes [10]
2. Logically centralized control [10]
3. Exposure of abstract network resources and state to external applications [10]

The SDN architecture is represented in Figure 1.1. The SDN controller is

software that could be placed on a dedicated physical resource or on a virtual machine that is running in a shared physical computing resource. The control plane, which manages the networking functionality is located in the controller, making the network elements simple forwarding devices. The packets are routed by the forwarding layer, which is made up of network elements, according to the instructions given to them by the controller.

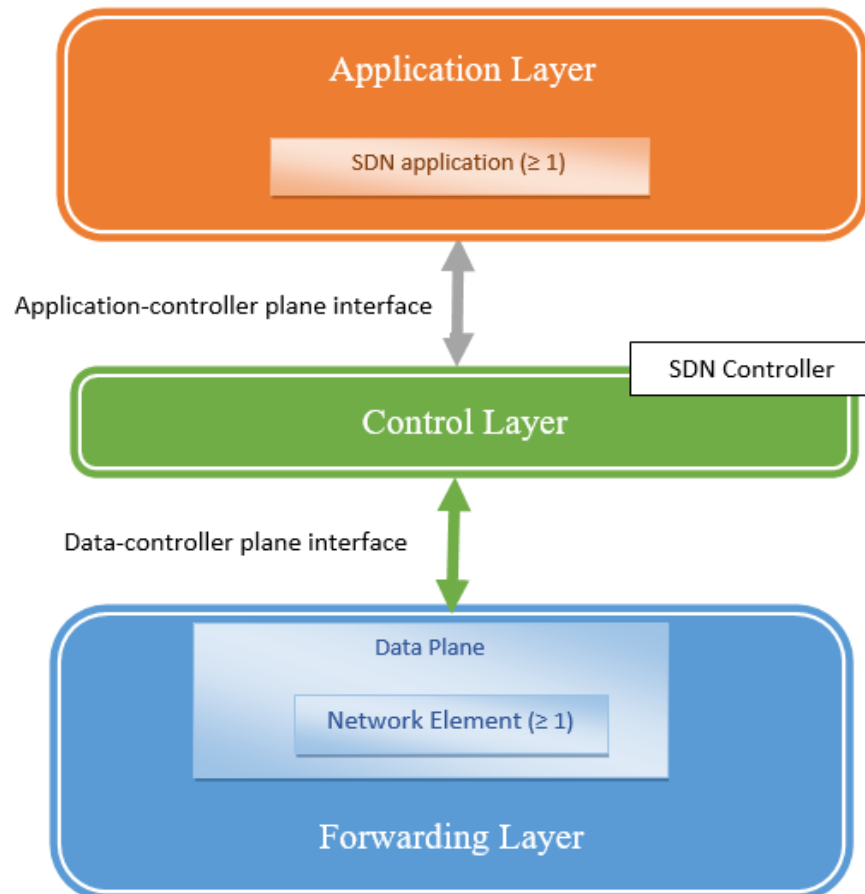


Figure 1.1 - SDN Architecture

SDN is an ideal choice for large, complex networks with high availability requirements. Moving from traditional network management methodologies to SDN will improve system reliability. The controller gives the ability to automatically reconfigure the routing and forwarding tables in the network elements on the fly without the need for manual reconfiguration. Furthermore, SDN gives network administrators real-time visibility into network performance, which aids in network performance and efficiency optimization. Redundancy is used in traditional networks to ensure network availability by installing more

equipment, which comes at extra costs. The uptime of the network is boosted without adding new hardware or increasing costs since SDN provides the ability to instantly reroute or stand-up new functions and routes in real-time. To support the expansion of applications, cloud environments rely on complicated networking infrastructure, making it one of the areas where SDN technologies can be used.

There are two distinguished areas where SDN can be applied in the cloud environment depending on the geographical distribution of its network devices. The SDN controller may reside inside the data center and control the network elements at the LAN level, or at the WAN level between the data centers, which are possibly located in different cities and countries.

A big portion of the traffic in the cloud environment flows between virtual machines where the cloud applications are hosted. Thus, it can be managed at the virtual network layer. Traditional 2-layer technology is employed inside the data center, which faces the following issues.

First, a Spanning Tree Protocol (STP) is utilized to prevent looping, which can result in a variety of problems, including low bandwidth utilization [13]. Secondly, the forwarding information base (fib) table of the standard Ethernet switch is small to accommodate tens of thousands of MAC addresses. Thirdly, in data centers with a large number of network nodes, broadcast-based protocols such as Address Resolution Protocol (ARP) lead to wastage of bandwidth. Finally, the number of virtual networks supported by VLAN, which is used by traditional layer 2 technology, may be limited in data centers with a large number of tenants.

The rise of SDN has inspired interest in adopting it inside data centers to replace traditional 2-layer technology and overcome its flaws. An SDN-enabled cloud data center can solve many traffic engineering issues. Auto topology discovery, auto addressing, auto-routing, load balancing, and quick fault detection and recovery are the advantages of the proposed design by the authors [13]. Figure 1.2 illustrates an SDN-enabled data center with a tree network topology.

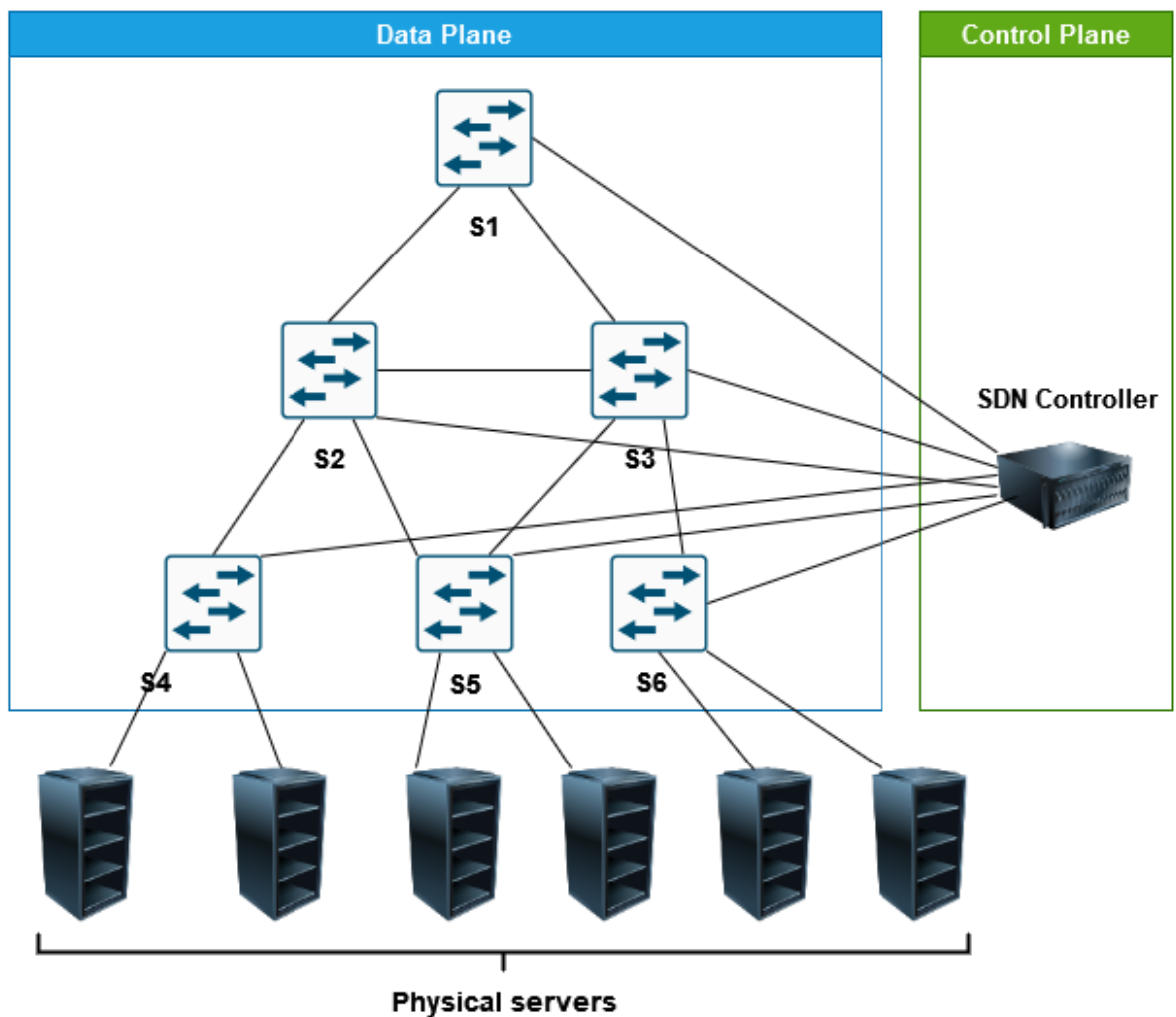


Figure 1.2 - Example of an SDN-enabled data center

An SDN-enabled data center is located at the LAN level of the cloud, behind an edge network device, which is responsible for connecting the data center with the global Internet. At the WAN level, where multiple geographically distributed data centers are connected, the services of ISPs and telephone, cable, or satellite companies are involved to create the communication network for the customer.

WAN technologies operate on the first three layers of the OSI model [14]. At the WAN level, a packet switching mechanism is used over the shared transport network. Virtual channels are used over the packet-switched network to provide private communications. MPLS is one of the technologies that provide connection-oriented packet-switched solutions. The MPLS forwarding table, which contains all of the necessary information for distributing labeled packets, is used in the label

switching mechanism.

The demand for on-demand bandwidth and low latency prompted by cloud applications is always increasing. As a result, the WAN sector is having trouble keeping up with rising application demands, necessitating WAN upgrades as well. A software-defined WAN is an emerging concept used in cloud computing. SD-WAN refers to a global computing network that spans large areas and has a significant number of nodes, some of which may be in different cities or countries. SD-WAN can be thought of as a software abstraction of MPLS that can be used in a broader range of scenarios. It enables a novel networking concept known as application-driven networking, in which the network is intended to adapt to the needs of the applications.

Figure 1.3 shows an example of an SD-WAN network. The edge routers of data centers are managed through the SDN controller, while they are connected through the public Internet.

Both SDN-enabled data centers and SD-WAN techniques can assist in reducing operational expenses and improving resource utilization. Without compromising data security or privacy, network administrators can make better use of available bandwidth while assuring excellent performance for critical applications. Both techniques employ the same idea to separate the control plane from the data plane to make the network more intelligent, but the primary difference is in their use. In the SDN-enabled data center, the customer provides their requirements regarding bandwidth and QoS, while in the SD-WAN the management of the bandwidth and QoS is carried out in the background by the provider, eliminating complexity for the end-user.

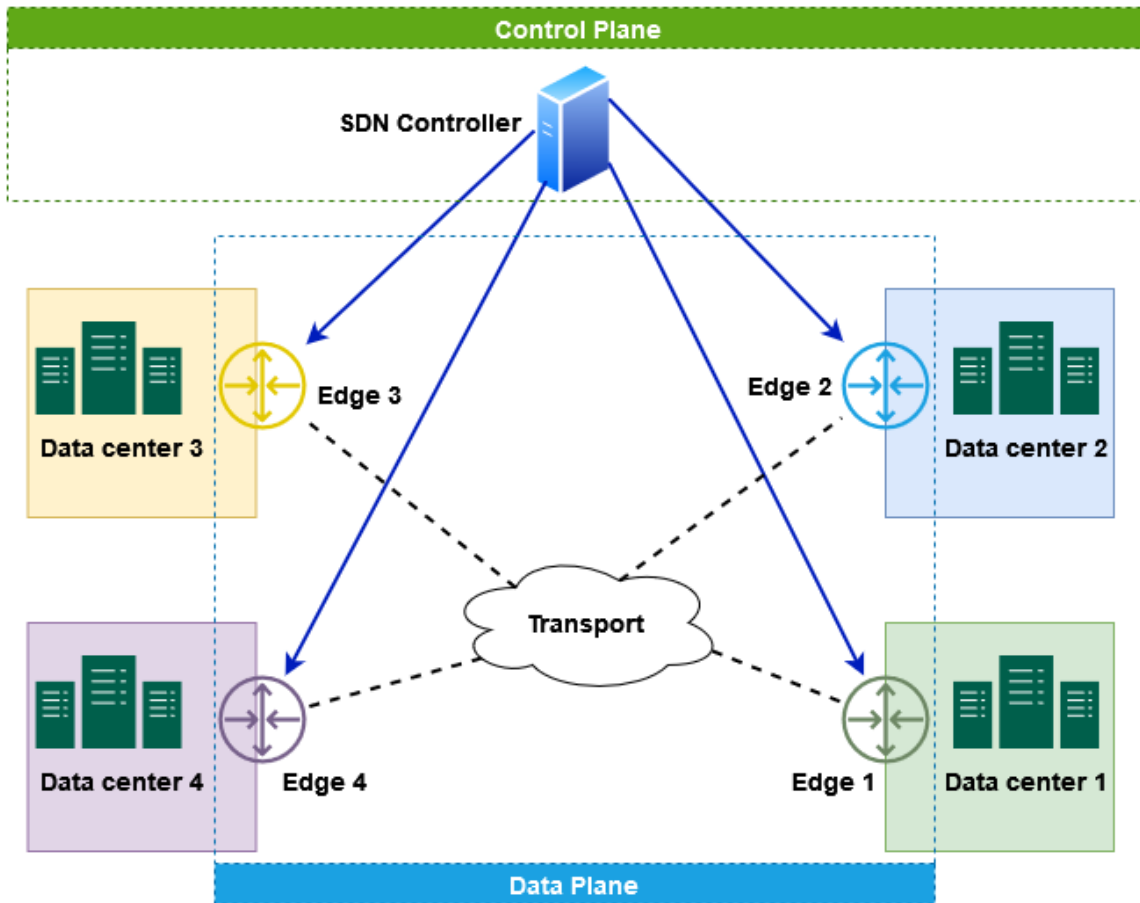


Figure 1.3 - Example of an SD-WAN architecture

1.2. Analysis of simulation problems in complex systems

With the advancement of computing equipment, a new subclass of mathematical models for simulation purposes has emerged. Simulation modeling (also known as situational modeling) refers to the design of a model in the form of a computer program that simulates the sequence of state changes in the system and represents the behavior of the system being modeled. A simulation model implements a timing diagram of the operation of the system being simulated. Simulation modeling or simulation experiment refers to the process of developing and evaluating such models, while simulation model refers to the algorithm itself. A simulation model is a logical-algorithmic description of the behavior of individual system elements and the rules of their interaction, representing the sequence of events that occur in the simulated system.

Typically, simulation modeling is used when:

- experimenting on a real object is either too expensive or impracticable.

- an analytical model cannot be constructed (the system has time, causality, consequence, non-linearity, and stochastic variables).

The following are the common stages in simulation technology:

1. Developing a process model;
Examining the model closure and devising a method for calculating internal features from known exterior ones;
2. Development of a computer program to calculate internal characteristics as well as output characteristics as a function of internal and external characteristics;
3. Identification of the model, i.e., determining the values of the model exterior attributes;
4. Verification of the model, i.e., determining its applicability limits;
5. Organizing the model operation, which includes simulation experiments.

Thus, the goal of simulation modeling is to recreate the behavior of the system under investigation based on the findings of an analysis of the most important relationships between its components, or, in other words, to create a simulation modeling of the subject area under study for various simulation experiments.

When integrating components into a complicated simulation model, several issues arise. One such important and non-trivial problem is model time management. A discrete model with a fixed simulation time step is the simplest method of control. However, because the different components of the model have their unique time scales, a constant simulation step is not always convenient, especially when modeling complicated, multi-component processes. When numerous models, each with its model time management, operate in parallel and exchange messages that can drastically modify the behavior of the model that receives the message, serious problems with model time management occur.

Parallel processes in simulation calculations frequently exchange data in addition to sending and receiving messages. There can be a variety of major issues,

such as multiple parallel processes attempting to edit the same data at the same time, or a process attempting to retrieve data while someone else is modifying it. In this case, a data set may appear which is, for example, not conceivable in principle according to the task logic.

When working with material and information resources, it is also worth addressing the simulation issues. Material resources are divided into two varieties: non-movable and movable. A non-movable resource is allocated at a specific location (both in reality and in the model). A relocatable resource is allocated to a client, after which the client uses it at other locations and returns it only when no further use is needed.

The simulation of a **non-movable resource** as a multi-channel service unit is required. One service channel is associated with each resource unit. A transaction is accepted by a channel for a specific amount of time, which is determined by the node properties, the transaction, and other parameters. The channel is unconditionally released after the service timer expires, and the transaction continues to the next node. Channels can operate to interrupt the service of lower priority transactions with higher priority transactions.

A **movable resource** could be a warehouse of resource units with a known number. There are no restrictions on the number of warehouses. A transaction is queued in a warehouse and requires a particular number of resource units to be allocated. The resource warehouse is represented as a node in the simulation model. A node creates a queue of transactions that can be sorted chronologically or by priority criteria.

Essential information, operational information (e.g., stock market information from Internet sites), temporary entitlements, documentation, and other intangibles are information resources that are required to execute a critical function. These resources are divided into two varieties:

1. A *start information resource* without which a function cannot be started (e.g., the right or permission to perform it, instructions for assembling a brand-new device). It is the start information resource that makes it

possible to send a request for a function, i.e., to place a transaction in the maintenance queue.

2. An *operational information resource* is required in the fulfillment of the function (e.g., operational dispatch information, the absence of which makes it impossible for an aircraft to land at an aerodrome).

In general, simulation models bring different types of uncertainty to the outcomes:

- methodological, related to the methods used to calculate the characteristics of the models;
- statistical, due to the random number generators used;
- transformational, characterized by the assumptions made during the computer representation of the input data;
- arithmetic, associated with the processor bit grid limitation.

The simulation method uses a series of experiments rather than a single experiment to reduce the statistical error. However, the simulation time and the cumulative arithmetic error increase when repeating the experiments. As a result, in the practice of simulation research and complex system design in various application fields, it is required to pay special attention to methodological and technological issues related to the use of simulation modeling system tools.

In today era of digital transformation, traditional design schemes for complex structured systems and their components are being replaced by the new strategy focused on the construction of so-called digital twins of products or systems. The term "digital twin" refers to a single virtual model that accurately describes all of the properties, operations, and relationships of a single object as well as the entire system. The design of such a model begins early in the product development process and is refined throughout the product life cycle.

1.3. Simulation modeling tools

There are numerous simulation application packages, environments, and systems. This diversity of tools is partly due to the many types of simulated systems, purposes, and modeling methodologies. They differ in terms of

application area, modeling technology, and visualization method. They fall into two categories: general-purpose modeling platforms and specialized software tools for modeling specific systems. The most common general-purpose modeling tools are covered in this section.

The first simulation tool platforms appeared in the mid-1970s. In particular, the GPSS system allowed the creation of models of controlled processes and objects mainly for technical or technological purposes. In the 1980s, simulation modeling systems began to develop more actively. At that time more than 20 different systems were in use in different countries. In particular, GASP-IV, SIMULA-67, SLAM-II, GPSS-V, etc. In the 1990s, the set of packages for simulation modeling appeared which became more and more developed and convenient even for the untrained user. Such tools include Powersim, ReThing, Extend+BPR, Pilgrim, etc. So far, a variety of computer simulation systems have been developed. The most common simulation systems are listed below.

- **Powersim Studio**

The methodology used in Powersim Studio [16] is based on classical system dynamics methods. It has extensive visual programming tools and various advanced capabilities, including built-in risk analysis and business process optimization. Models created in Powersim Studio can be dynamic or simply computational, allowing the calculation of complex operational and financial indicators. Powersim Studio can simulate both discrete and continuous processes. For tasks, which do not match Powersim Studio built-in integration capabilities, the Powersim Studio SDK, a software development kit, can be used.

- **ExtendSim**

ExtendSim [17] is an easy-to-use but extremely powerful process-modeling tool. It includes a set of tools for any type of simulation. ExtendSim is a multi-approach environment, so continuous and discrete events, agent-based models, linear, non-linear, and mixed systems can be simulated simultaneously.

- **Enterprise Dynamics**

Enterprise Dynamics [18] is an object-oriented modeling platform

combined with an event-driven approach to provide enterprise supply chain modeling. The Enterprise Dynamics discrete event modeling software platform is delivered on a single core basis. The user can select standard simulation objects (called "atoms"), which capture the behavior of their real-world equivalents, from a library and create a model by clicking and dragging the objects into the model space. For each simulation object, parameters can be changed to change its behavior.

- **AnyLogic**

AnyLogic [19] is a versatile simulation platform that is suitable for tasks of any complexity from a variety of industries. The AnyLogic toolkit provides a multi-approach simulation capability (discrete event, agent-based, and system dynamics). These three methods can be used in any combination on a single application to simulate a business system of any complexity. AnyLogic supports different visual modeling languages - process diagrams, state diagrams, flow and storage diagrams. The visual development environment speeds up the model-building process. AnyLogic models are completely separate from the development environment. This means that AnyLogic allows models to be exported to the AnyLogic Cloud [20]. AnyLogic Cloud is a service based on distributed cloud computing. This secure web-based platform allows running multiple experiments simultaneously, speeding up and simplifying the operational use of simulation models.

1.4. Overview of specialized cloud infrastructure simulation systems

Ecosystems of cloud architectures as well as the growing demand for energy-efficient information technologies require timely, reproducible, and controllable approaches to evaluate algorithms, applications, and policies before the actual development of cloud products. One such option is the use of simulation tools that open up the possibility of evaluating the design before software development in an environment where tests can be replicated. Simulation-based systems have significant advantages in cloud computing, where access to infrastructure is charged in real currency because they allow cloud customers to test

their services for free in a reproducible and controllable environment and tune performance bottlenecks before deploying to real clouds. On the provider side, simulation environments enable the testing of various resource provisioning scenarios under various workloads. Such research can assist providers in lowering the cost of accessing resources while improving profitability.

Today, in addition to general-purpose simulation platforms that enable the simulation of many different types of production systems, there are specific, cloud-oriented software systems that simplify the simulation model creation process. Such software systems generate a cloud model, based on initial data about its infrastructure and the communication facilities used, the intensity of request flows between data center components, data transfer rates, the types of equipment used, the applications running, etc.

Cloud infrastructure simulation models are not created from scratch. Datacenter brokers, clusters, containers, job queues, and other key cloud components have ready-made simulation models. Models of individual elements of cloud infrastructures are created based on various data - test results of real devices, analysis of their operating principles, analytical relationships, etc. Such models provide a library of generic cloud elements that can be configured using pre-defined parameters. Additionally, the simulation system toolkit includes tools for statistically processing the simulation findings. The correctness of the initial data about the cloud computing infrastructure provided to the simulation system determines the quality of the simulation results.

- **CloudSim**

CloudSim [21] is one of the most popular simulation platforms for cloud computing infrastructures and services. The platform provides a generalized and extensible environment with basic classes for describing data center network topologies and messaging applications, brokers, users, computing resources, and policies for managing different parts of cloud computing infrastructure, dynamically inserting cloud infrastructure elements, stopping and restarting simulations.

CloudSim tools enable users to model and test new large-scale cloud computing infrastructures and application services, allowing them to concentrate on the specific data center architecture concerns they want to study. In particular, assessing the efficacy of cloud methods (policies, scheduling algorithms, load balancing policies, etc.) from a variety of viewpoints, including cost/benefit and application runtime acceleration. The absence of a graphical user interface (GUI) is CloudSim main limitation. CloudSim was developed at the Department of Computer Science and Software Engineering Faculty of the University of Melbourne [22].

- **CloudSim Plus**

CloudSim Plus [23, 24] is a fully-featured and fully-documented cloud simulation environment based on CloudSim, with numerous unique features ranging from the fundamental (making simple models) to the advanced (building complex models and simulating more realistic cloud scenarios). Multi-cloud simulation with virtual machine migration between data centers, vertical and horizontal scaling of computing capacity, parallel simulation experiments on multicore computers, dynamic creation of virtual machines and clusters, dynamic task ingestion, random host CPU core failures, and replication of failed virtual machines are among the advanced components available.

- **CloudSimEx**

CloudSimEX [25-26] is a set of CloudSim extensions that simplify the development of simulation models of cloud computing infrastructures and allow the simulation of new types of applications not supported by CloudSim. CloudSimEx extensions allow web session simulation, MapReduce simulation, and automatic identity generation. In addition, improved logging utilities, utilities to generate CSV files for statistical analysis, utilities to run multiple experiments in parallel, utilities to simulate network latency, and a new broker that supports the dynamic preparation and scheduling of virtual machines and cloud services are also available.

- **CloudSimSDN**

CloudSimSDN [27] is an add-on package for CloudSim designed to simulate host and network usage and response times in SDN-enabled cloud data centers. The main operational components of CloudSimSDN include topology generators, monitoring of power consumption and usage by both hosts and switches, flow channel manager, policies for host selection, link selection, virtual machine allocation, host reservation, etc. However, CloudSimSDN lacks supporting multiple workloads on the same link between resources (Hosts) [28].

- **EdgeCloudSim**

EdgeCloudSim [29] is based on CloudSim and provides edge computing infrastructure capability and features. New classes and modules for edge computing systems have been added to CloudSim, including a Load Generator Module, WLAN and WAN transmission delay, Edge Orchestrator to handle incoming client requests, and mobility module to indicate the location of the mobile device linked to the edge computing. EdgeCloudSim does not allow the decision-maker to choose different parameters for how and where the task will be performed [28].

- **CloudReports**

The graphical user interface provided by CloudReports [30] makes it easier for researchers to conduct empirical research. CloudReports allows researchers not only to document the findings of their empirical study but also to create and update policies via an application programming interface (API). Different resource allocation strategies for VMs are not available in CloudReports [28].

- **CloudAnalyst**

CloudAnalyst [31] is a tool that can assist in analyzing how a large Internet application operates in a cloud environment. The CloudAnalyst includes classes and various innovative functions that make a variety of tasks and procedures easier. These include realistic behavior model modeling in the same way that traffic generators do, network latency for data delivered over the Internet, administration of user requests across several data centers, and provision of a comprehensive GUI. However, for large-sized DC, CloudAnalyst needs to include more mechanisms and algorithms for resource management [28].

- **GreenCloud**

GreenCloud [32] major use case as a packet-level simulator is in the field of energy-efficient cloud computing data centers. The majority of GreenCloud goals are connected to cloud communications. It is important to keep in mind that the GreenCloud Simulator is an adaptation of the widely used NS2 network simulator, which is built in the C++ programming language and provided under a GPL license. The GreenCloud simulator is potential for estimating energy consumption levels connected with information technology equipment in a data center is one of its strengths. Computing servers, communication channels, and network switches are among the equipment that GreenCloud most effectively represents in this regard.

1.5. Cloud simulators classification approaches

The use of cloud computing is quickly growing, and there are several applications for various deployment patterns. Simulators are used to more easily analyze algorithms or policies because it is difficult to measure and verify application performance in a real cloud environment. Choosing the right modeling tool for an experiment, on the other hand, might be difficult for researchers. Despite the abundance of modeling tools, researchers obtain little direction or information about their applicability or accuracy for a specific role in each circumstance. Furthermore, because cloud computing simulators cannot replicate every situation in the system and are typically built to duplicate non-functional elements of the cloud system, they are ill-equipped for crucial research points in cloud systems. In this section, we provide an analysis of research related to the classification of cloud simulators which may help researchers in selecting a proper tool for their experiments.

Maarouf et al. [33] provided detailed information about existing simulators, their functions, and their characteristics, which allows researchers to get a general idea of each simulator before using them in their experiments. Ahmed et al. [34] analyzed existing simulator information and divided it into different categories such as underlying platform, availability, programming language, GUI, simulation

time, and energy model, as shown in Table 1.1. They noted that many researchers prefer to use and expand open-source cloud simulators such as CloudSim and GreenCloud.

Table 1.1 - Comparison between cloud Simulators [34]

Simulator	Underlying Platform	Available	Programming Language	Cost Modeling	GUI	Communication Model	Simulation Time	Energy Model	Federation Policy
CloudSim	SimJava	Open Source	Java	yes	No	Limited	second	yes	yes
CloudAnalys t	CloudSim	Open Source	Java	yes	yes	Limited	second	yes	yes
GreenCloud	NS-2	Open Source	C++, otcel	No	Limited	Full	Minute	yes	no
MDCsim	CSIM	Commercial	JAVA/C++	No	No	Limited	second	Rough	no
iCanCloud	SIMCAN	Open Source	C++	yes	yes	Full	second	No	no
NetworkCloudSim	CloudSim	Open Source	Java	yes	No	Full	second	yes	yes
EMUSIM	CloudSim, AEF	Open Source	Java	yes	No	Limited	second	yes	no
GroudSim	-	Open Source	Java	No	Limited	No	second	No	no
MR-CloudSim	CloudSim	Still not available	Java	yes	No	Limited	-	yes	yes
DCSim	-	Open Source	Java	yes	No	No	Minute	No	no
SimIC	SimJava	Still not available	Java	yes	No	Limited	second	Rough	Yes

Bashar [35] presented an evaluation of the simulation frameworks applicable to Cloud Computing systems. The comparison in Table 1.2 is summarized in terms of the API features, programming flexibility, Operating System requirements, supported services, licensing needs, and popularity. Bashar [35] provided recommendations on choosing the most suitable platform for researchers, administrators, and managers of cloud computing systems. In the absence of hardware, CloudSim simulation software is the best choice for simulating various performance evaluation experiments.

Byrne et al. [36] considered the features of modeling platforms and the survey results showed that CloudSim can act as a *de facto* base platform for development and research in the field of modeling. In total, more than 20 CloudSim extensions are used in various fields of research.

Table 1.2 - Comparison Summary of Cloud SIMULATORS [35]

Simulator	Provider	License	Category	API	OS	Services	Popularity	Comments
Eucalyptus	Eucalyptus Systems Inc.	Open source	Software Framework	Java/C	Linux, Windows	IaaS	10.00	AWS compatible Private Cloud
ns2	University of South California	Open source	Simulation Software	C++/OTel	Linux	N/A	2.78	Extended in GreenCloud simulator
CloudSim	University of Melbourne	Open source	Simulation Software	Java	Linux, Windows, Mac	IaaS	2.50	Generalized and extensible framework
Opnet	Opnet Inc.	Paid	Simulation Software	C/C++	Windows, Linux	N/A	2.35	Limited application testing for Cloud
GreenCloud	University of Luxembourg	Open source	Simulation Software	C++/OTel	Linux	IaaS	1.79	Based on ns2, focus on energy efficiency
OpenStack	Group of Companies	Open source	Software Framework	Python	Linux	IaaS	1.36	Compatible with Amazon Web Services
Open Cloud	Group of US Universities	Propriety Membership required	Scientific Testbed	Hardware based	Hardware based	IaaS, PaaS, SaaS	1.35	Appropriate for testing Cloud applications
Open Cirrus	Group of Universities and Companies	Propriety Membership required	Scientific Testbed	Hardware based	Hardware based	IaaS, PaaS, SaaS	0.34	Focus on testing Federated Cloud Datacenters
CloudAnalyst	University of Melbourne	Open source	Simulation Software	Java	Linux, Windows, Mac	IaaS	0.23	Extension of CloudSim with GUI
iCanCloud	University of Madrid	Open source	Simulation Software	C++	Linux	IaaS	0.09	Uses OMNET++ framework

Additional capabilities such as heterogeneous data centers, graphical user interfaces, and more complicated distributed applications are supported with CloudSim extensions. As a result, it may be more effective for researchers to conduct cloud computing experiments with these extensions rather than modeling the relevant services directly in CloudSim in some circumstances. Byrne et al. [36] examined 20 extensions that were chosen based on the fact that they are related to at least one Google Scholar publication.

Alshammari [28] re-examined the CloudSim extensions and identified four sets of functionalities, every extension can be characterized in terms of its belonging to these sets. The extensions are arranged in the following categories, which are outlined in Figure 1.2.

- **NET:** includes CloudSim extensions for modeling distributed applications and sophisticated network settings.

- **GUI:** contains extensions that are more user-friendly than CloudSim for customizing modeling and presenting results, which are generally GUI style front-end.
- **QoS:** provides extensions that represent cloud deployment options and service quality reliability.
- **PWR:** contains extensions that include models for energy estimation.

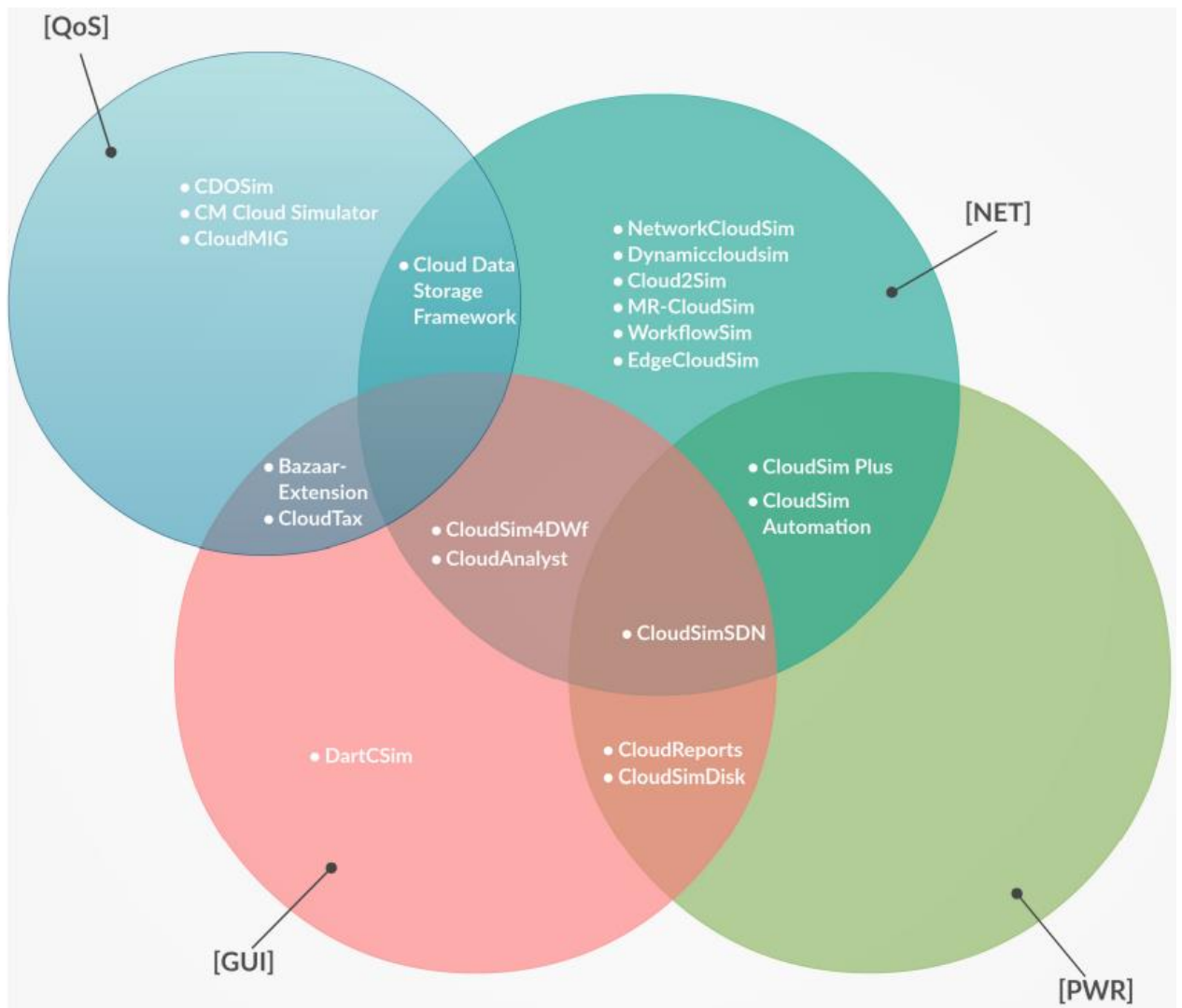


Figure 1.4 - The Main Four Categories for CloudSim Family of Extension [28]

Alshammari [28] also provided comparison tables between those extensions in terms of availability, year of release, number of versions, and features. We provide an updated version of this information, as shown in Table 1.3. We may notice that many extensions are no longer available or are not being updated for a

long time. There are only three extensions (Edge CloudSim, CloudSim Plus, CloudSim Plus Automation) that have been updated recently.

Table 1.3 - Availability and Features of CloudSim Family of Simulators

Simulator	Year	Availability	Number of versions	Last update	Networking Typologies	Graphical User Interface	Energy Consumption
CloudAnalyst	2010	Open Source	1	2010	Yes	Yes	Limited
NetworkCloudSim	2011	Not Available	1	-	Yes	No	Limited
CloudMIG Xpress	2011	Open Source	1	2012	Limited	Yes	No
CDOSim	2012	Not Available		-	Yes	No	Limited
WorkflowSim	2012	Open Source	1	2012	Yes	No	No
MR-CloudSim	2012	Not Available	1	-	Yes	No	No
CloudReports	2012	Open Source	1	2015	Yes	Yes	Yes
DartCsim	2012	Not Available	1	-	Limited	Yes	Limited
DynamicCloudSim	2013	Open Source	1	2013	Yes	No	No
CloudSim (Plus) Automation	2014	Open Source	7	2021	Yes	No	Yes
Cloud2Sim	2014	Open Source	1	2015	Yes	No	No
CloudSimSDN	2015	Open Source	3	2019	Yes	Yes	Limited
CloudSimDisk	2015	Open Source	1	2015	Limited	Yes	Yes
Bazaar-Extension	2016	Not Available	-	-	Limited	Yes	Yes
CloudTax	2016	Not Available	-	-	Limited	Yes	Yes
CM Cloud Simulator	2016	Not Available	-	-	Limited	No	Limited
EdgeCloudSim	2017	Open Source	4	2020	Yes	No	No
CloudSimPlus	2017	Open Source	7	2021	Yes	No	Yes
CloudSim4DWf	2017	Not Available	-	-	Yes	Yes	No

1.6. Direction and objectives of the study

Cloud computing infrastructure is becoming increasingly popular. The usage of cloud computing has reached new heights. Many businesses are shifting their workloads to the cloud to take advantage of the benefits it offers over on-premises alternatives. Multi-tenant data center environments are created by cloud providers, in which different clients have access to the same underlying resources. As a result, new network management concerns at the LAN and WAN levels need to be addressed. SDN technology, on the other hand, is ideal for large, complex networks with high availability demands. New approaches to network management have appeared to adopt SDN architecture on the cloud.

Simulators are commonly used by researchers to simulate essential parts of cloud computing infrastructure, as conducting cloud computing experiments on real cloud infrastructures wastes time and money. Building models using cloud computing modeling tools to simulate SDN architecture can help in understanding the underlying system and measuring its performance.

CloudSimSDN extension has models to simulate SDN-enabled cloud data centers, but only supports one type of workload on the same link between resources [28], lacks documentation, has not been maintained or updated recently, and their GUI implementation is not available publicly. Also, CloudSimSDN lacks significant improvements on the CloudSim code base and the added features which are delivered by CloudSim Plus.

CloudSim Plus is a general-purpose cloud simulation framework that extends CloudSim and has many added features and significant improvements compared with CloudSim. Although CloudSim Plus provides many exclusive features, it has some shortcomings that motivated this work, which are the following: i) absence of a graphical user interface; (ii) absence of power monitoring of network elements such as switches; (iii) absence of SDN simulation models.

This study aims to develop tools for building simulation models of cloud computing infrastructures in software-defined distributed networks. Achieving the goal of the work required solving the following main tasks:

1. to review the research on SDN architecture and its main components.
2. to define the task of monitoring the power of network elements and integrate its implementation with the CloudSim Plus platform.
3. to define the tasks for the formation of multi-stage programs for modeling cloud computing infrastructure in software-defined networks and integrate its implementation with the CloudSim Plus platform.
4. to develop a web interface for simulation experiments in the modified CloudSim Plus platform and a graphical display of simulation results.

2. Designing generalized cloud computing simulation system in software-defined networks

2.1. Analysis of the SDN core components

Before designing a simulation system, a deep understanding and analysis of the target system core components and the relations among them should be acquired. Figure 1.1 illustrates the SDN architecture. The architecture consists of three layers:

- **Application Layer** contains applications or programs that deliver network requirements or behavior to the control plane, the SDN controller, through the A-CPI interface. Also, these applications are given an abstracted perspective of the network which they can utilize to make decisions. Some applications at the SDN application tier include path reservation, network topology discovery, and network resource provisioning.
- **Control Layer** is the essential operating point where the SDN controller is located. It can communicate with the network elements (switches, routers, etc.) through the D-CPI interface. Thus, we can say that the SDN controller is the “brain” in the SDN architecture. It applies the application layer requirements and policies to the devices in the forwarding layer. Also, it provides hardware information back to the application layer including a network overview such as events and statistics.
- **Forwarding Layer** consists of the network elements that route the network traffic according to the controller instructions.

The network elements in the SDN architecture are the routers, switches, and middlebox (firewalls, load balancers, etc.) which forward the packets according to forwarding tables. The SDN controller manages the forwarding tables of the devices by processing the requirements of the application, generating the appropriate rules, and applying them to the network elements. As a result, the network elements are just dumb devices with no logical processing units. Figure 2.1 shows a generic block

diagram and packet processing phases of the network element which is managed by the SDN controller.

The network element contains a forwarding table, which is used in the traffic forwarding process. When a packet is received by the switch, it searches for a matching entry in its forwarding table. The forwarding rule consists of a key and an action. The Forwarding key should be unique and it is generated using the received header data of the packet, such as source and destination addresses, a label or flow-id, etc. If a matching entry is found, then the switch will forward the packet according to the action of that entry to one of the directly connected devices (another switch, host, etc.). When there is no matching in the forwarding table, then a request will be sent to the controller which generates the appropriate rule and updates the forwarding table. Thus, the network element in generic SDN architecture contains two key components: the forwarding table and the forwarding key.

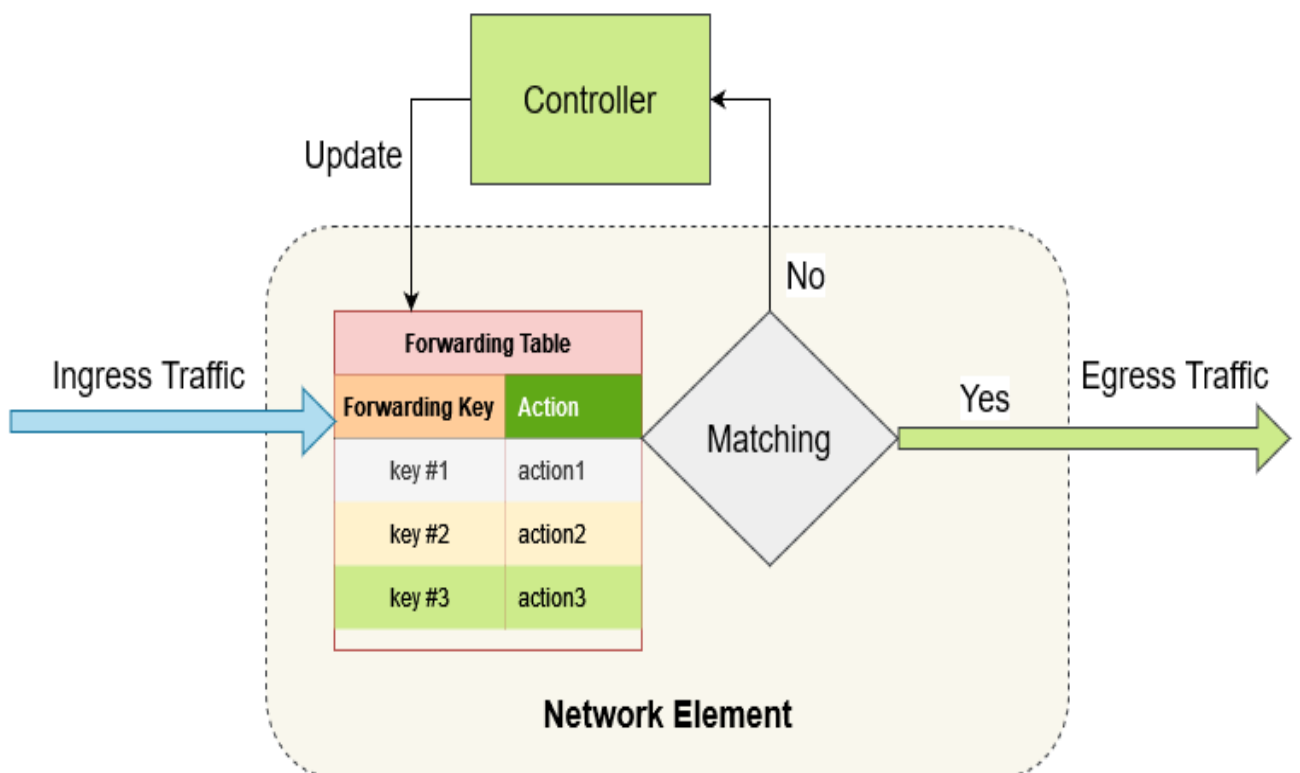


Figure 2.1 - Traffic processing in the network element

The application layer contains different network applications which are used to facilitate network management and enforce policies in the network such as QoS, resource allocation, etc. It gives developers the freedom to create their customized application suite. Thus, the key components of the application layer are the network application and its enforced policies.

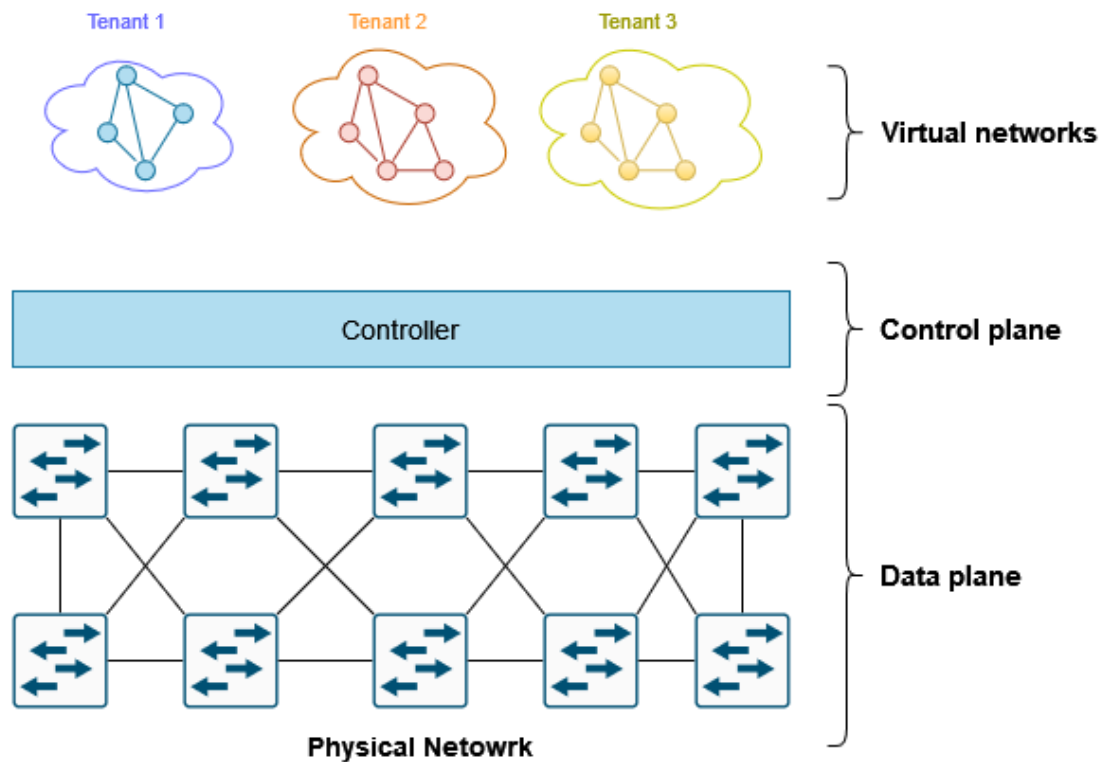


Figure 2.2 - One physical network to multiple virtual networks mapping in a multi-tenant environment

The internal design of an SDN controller is not specified in the architecture. Typically, a controller subnetwork scope usually extends beyond a single physical network element. Thus, the SDN controller considers itself the owner of the network resources that have been assigned to it for management purposes. The knowledge of the physical network topology and the links state between the different NEs compose an information model instance that will be used by the controller to manage the resources which are located under its control. Moreover, the controller must expose the information model instance to the SDN applications through the A-CPI interface. The information model contains sub virtual networks

that give an abstract perspective of the physical network to the applications. The virtual network spans multiple network elements and creates an isolated sub-network of the physical topology. Thus, every application may create one or multiple virtual networks depending on their enforced policies. Typically, one virtual network is owned by one customer that connects their services through the physical network but in an isolated channel such as MPLS VPN.

A typical SDN controller should build the required knowledge about the physical topology and the link state between all network elements under its control. Also, The SDN applications provide the controller with knowledge about the virtual topologies of different tenants, their policies, and requirements.

2.2. Analysis of the network simulation models in CloudSim Plus

CloudSim Plus is the cloud simulation tool that was selected for further development in this study. Thus, an analysis of the network simulation models, which are offered by the tool, is required to get the required knowledge about its structure and to identify the specific models which need improvements or changes to facilitate the design and development of the targeted tool.

CloudSim Plus provides generalized network modeling classes, that simulate the different components of the network in the cloud datacenters. Some modeling classes, which are designed and implemented by NetworkCloudSim [37], are imported and integrated into the tool. Figure 2.3 shows the class diagram of the network modeling in NetworkCloudSim.

The modeling classes in NetworkCloudSim target two main components of the cloud network, which are the following:

1. Network topology modeling classes which consist of:
 - Switch interface and its implementation (EdgeSwitch, AggregateSwitch, and RootSwitch) that can be used as switches or routers.
 - NetworkPacket and HostPacket that represent data flow between VMs. CloudSim Plus made changes to those classes and added VmPacket class. HostPacket is the packet that is transmitted in the data center network between the source and the destination host, while VmPacket is

encapsulated in the HostPacket and represents the actual data flow between the source and destination VMs.

- Network Host and Network classes that extend the native Host and VM classes in CloudSim to make these entities aware of the network.
2. Cloud application modeling classes that model generalized applications, which are AppCloudlet and NetworkCloudlet. CloudSim Plus provided significant improvements to those classes by adding the ability to simulate applications with multiple processing and transmission stages.

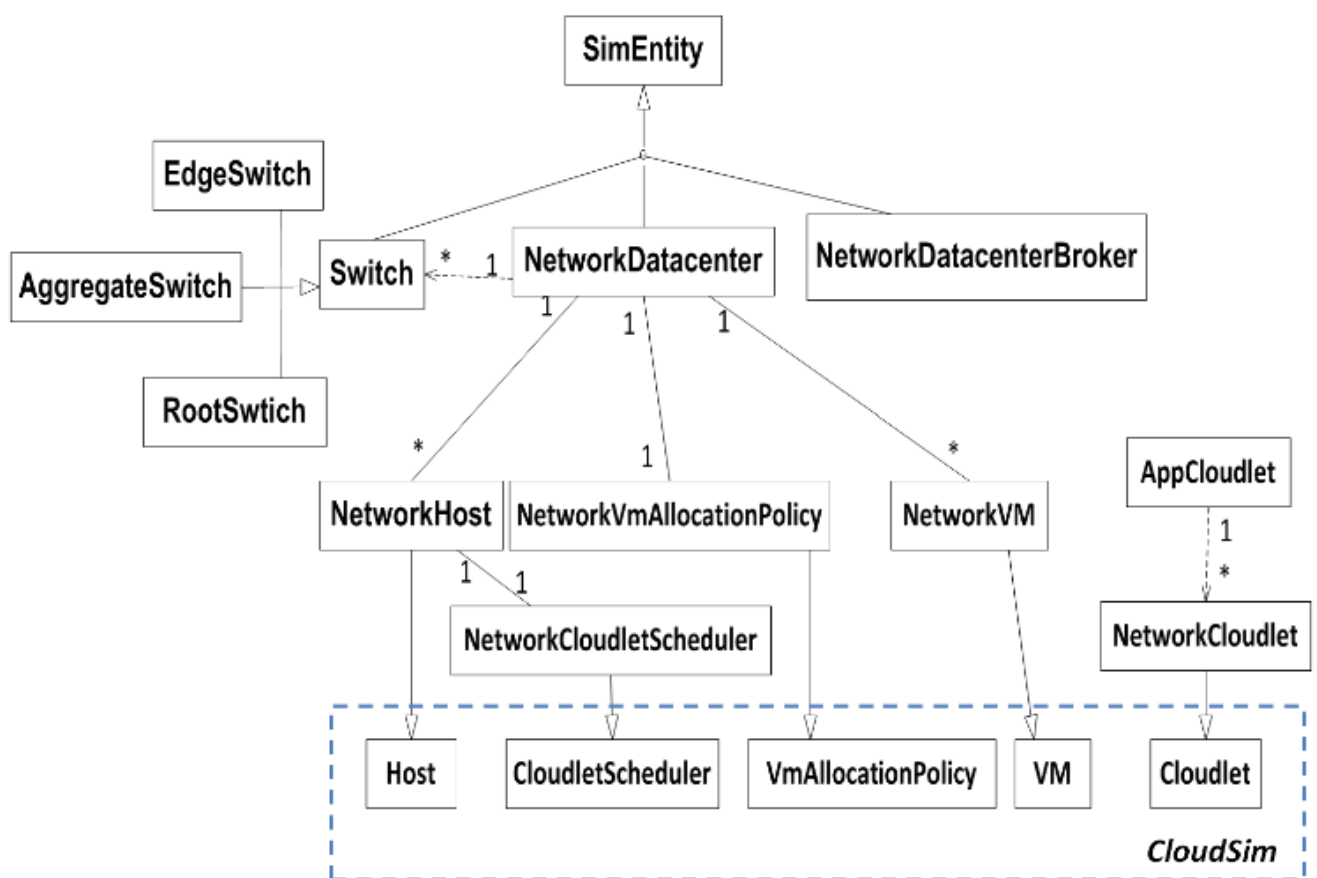


Figure 2.3 - Class Diagram of NetworkCloudSim [37]

In addition to the imported modeling classes, CloudSim Plus provided its network topology modeling, parsing, reading, and delay matrix calculation classes. It depends on the BRITE format [38] to construct the topology graph links, their bandwidth, and delay parameters. Floyd-Warshall algorithm [39] is used to calculate the delay matrix.

The Network simulation capabilities of CloudSim Plus leak SDN modeling

support, also there are no routing or switching algorithms in their implementation. It provides a facility for users to design their routing algorithms, and configure network and switching latencies [37]. It does not simulate the flow of the packets between the different switches or routers, instead, they are directly inserted in the destination host after calculating the transmission delay.

2.3. Object-oriented design of generalized cloud computing simulation system in software-defined networks

The network modeling design of CloudSim Plus and the study of SDN key components provide the essential knowledge about the intended system components and the abilities and leakages of the selected simulation tool. This section provides the design details of a generalized tool to simulate the cloud computing infrastructure in software-defined distributed networks.

The design follows the object-oriented design patterns. Interfaces, abstracted factories, and generics are used to ensure code reusability, extendibility, and maintainability. The design is logically divided into packages, every package represents a specific layer or component in the simulated system. A part of the design extends the CloudSim Plus modeling classes to add the unsupported functionality. First, we will introduce the core packages of the system that model the SDN architecture. Then we will provide details about the modified and extended parts of CloudSim Plus.

2.3.1. Network Elements

Figure 2.5 shows the class diagram of the network elements package. CloudSim Plus provides switches modeling classes, but they lack the support of routing or forwarding mechanisms. NetworkElement interface provides the abstraction of a network element in an SDN network by including the forwarding and routing tables in its definition. This interface is used in all modeling classes to represent a network element. NetworkElementNull class implements the Null Object Design Pattern for Table objects.

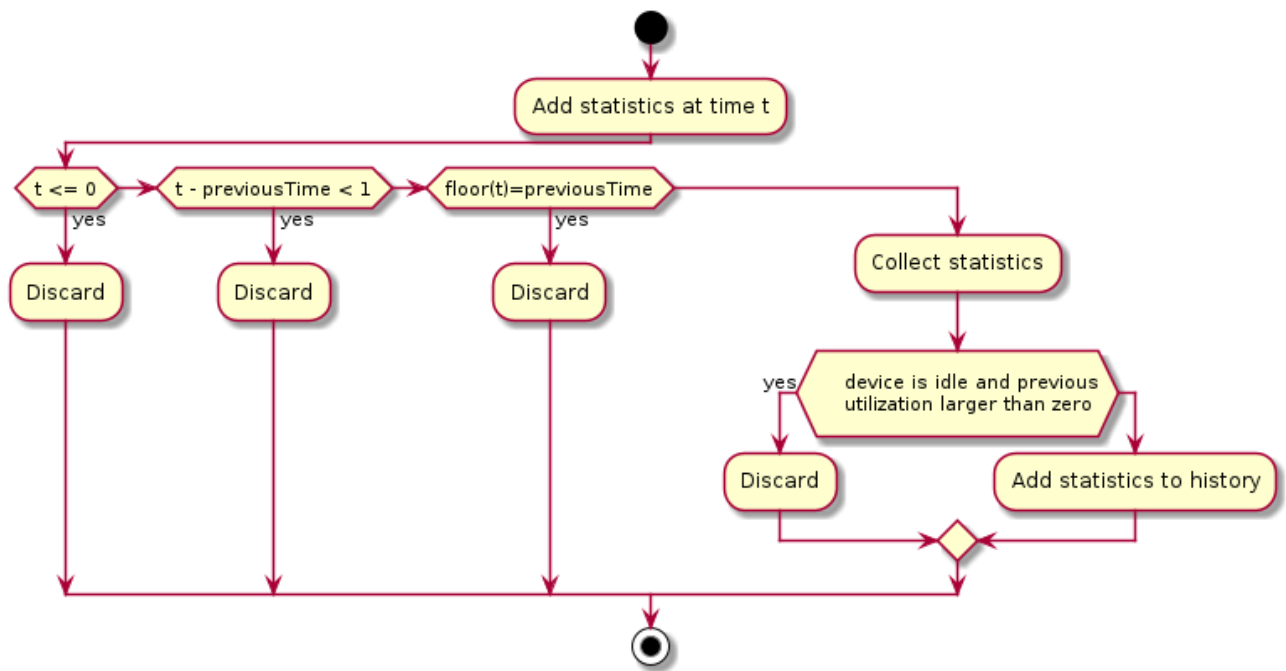


Figure 2.4 - Activity diagram of adding new statistics entry for a network element

The subpackage “stats” contains the interfaces and classes that are required to collect and store the utilization statistics of a network element. The network element should extend the interface `NetworkElementStatsComputer` to enable the computation of statistics for its resource utilization. In addition, it is possible to store a history of the resource state during the simulation. The history entry stores several values at the recorded time, such as the number of bytes transferred through the network element to the upper or lower level in the network topology, the power consumption, etc.

The activity diagram of adding new statistics entries to the utilization history of a network element is shown in Figure 2.4. The utilization entry should not be added if the simulation clock was not changed yet, the time passed is smaller than one second, or the floor time is equal to the previous time.

The subpackage “switches” contain the classes, which are responsible for the statistics collection of the switches. In this way, a new type of network element can be added, such as a firewall-staging block. The packages provided can be used to manage traffic and collect usage statistics.

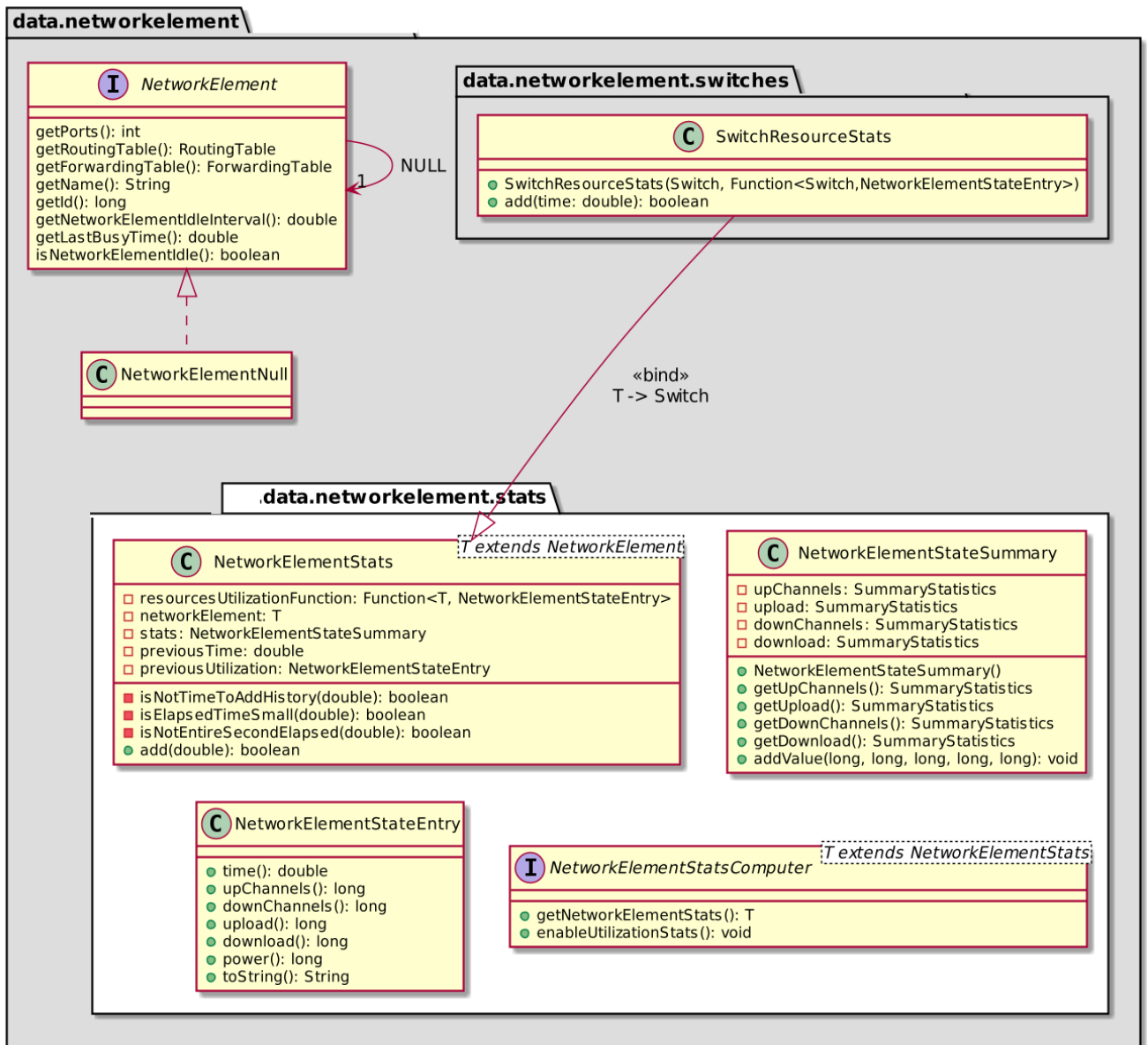


Figure 2.5 - Class diagram of the network elements package

2.3.2. Control Tables

Forwarding and routing tables are key components in the SDN architecture. The controller uses the routing table to store the information about the physical nodes in the network topology and how they are connected. This information helps in decision-making and traffic routing functionality. On the other hand, the forwarding table is managed by the controller and used by the network elements to forward the packets. The class diagram of the control tables package is shown in Figure 2.6. It consists of the following:

- Table interface, which represents a generic table. K and V generic types are the table key and value columns.
- AbstractedTable class provides the implementation for the common methods.
- ForwardingKey class represents a forward key object to be used in the forwarding process. It consists of three parameters, the source, the destination addresses, and a flow Id. The addresses are unique identifiers assigned by CloudSim when creating the simulation entities. flowId parameter is a unique identifier that is generated by the controller to control a specific flow of a specific application or a specific rule in a provided policy from the application layer.
- ForwardingTable class represents a forwarding table in a network element. It extends the AbstractedTable class and uses ForwardingKey and NetworkElement as parameterized types. Also, it implements the functionality of adding a new rule and searching the table for a specific forwarding key.
- RoutingTable class represents a routing table for a specific node in the topology. It will store all nodes that are connected to the node and a list of the physical links to reach them.

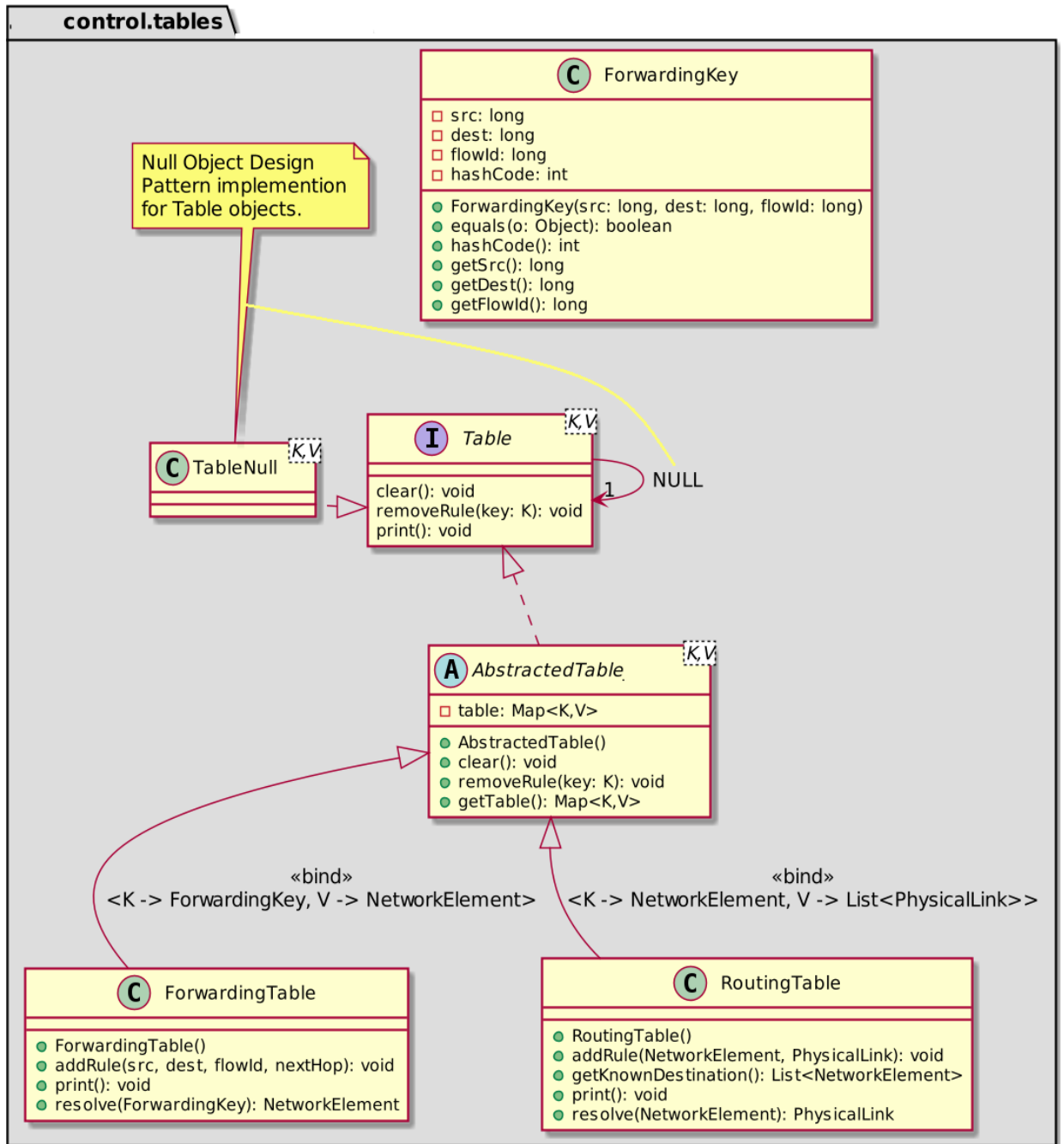


Figure 2.6 - Class diagram of the control tables package

2.3.3. Physical and Virtual Networks

The network package provides the modeling of the physical and virtual topologies. Although CloudSim Plus provides a representation of the network topology, its implementation is coupled with the BRITE format of the topology input file. The `TopologicalLink` class uses the node id from the BRITE file for addressing the source and destination nodes of the link.

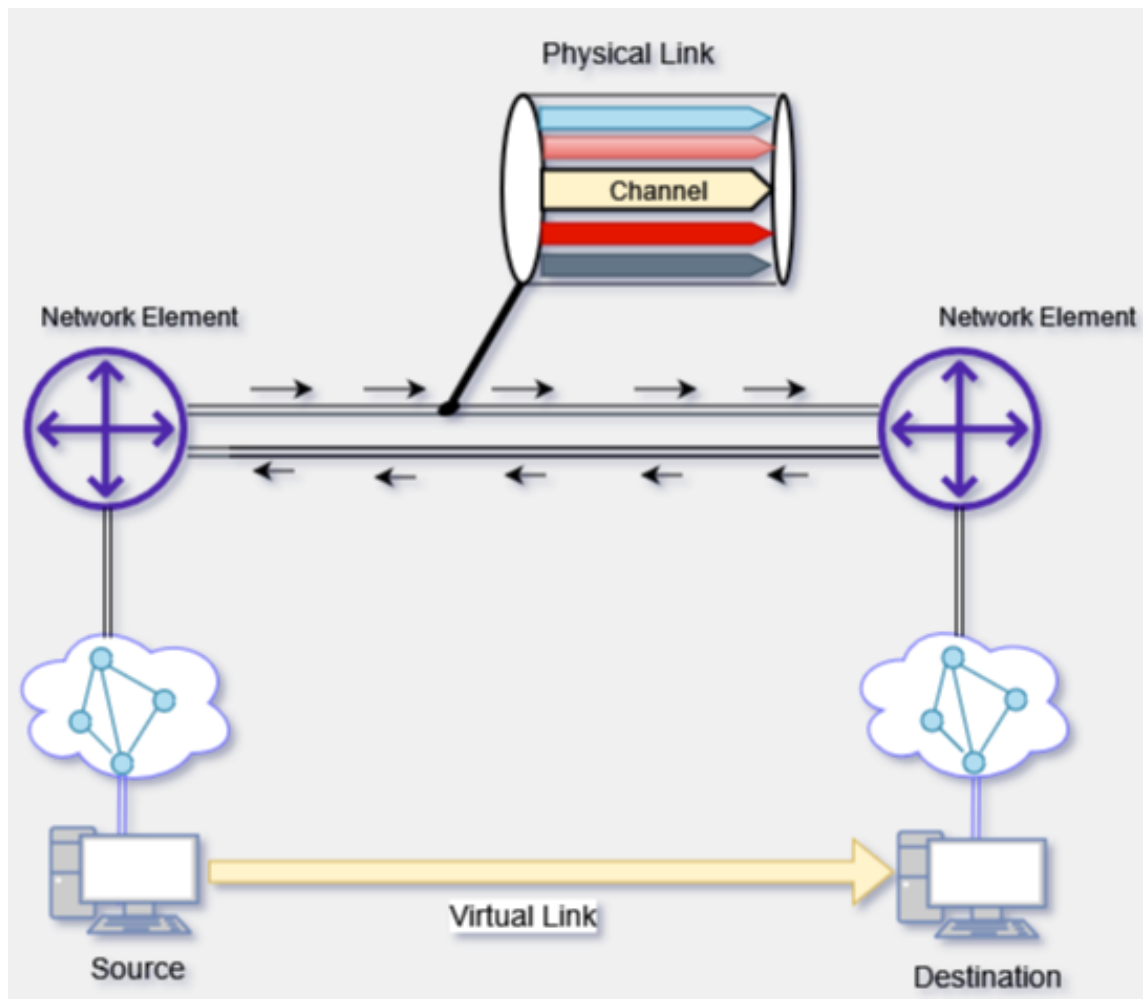


Figure 2.7 - Physical, virtual links and channel representation

We assume that the links are unidirectional, and the traffic flows in one direction. Thus, to represent a bi-directional connection between two nodes we should use two physical links. Every link has its bandwidth capacity and delay value. The virtual topology consists of multiple virtual links. The virtual link is also unidirectional and is defined by the source and destination entities, the required bandwidth, and a unique identifier (e.g., flow Id). Figure 2.7 illustrate how the physical and virtual links are represented in the designed tool.

The physical link may contain multiple channels, every channel is related to one flow (virtual link). The bandwidth of the physical link is shared between the channels according to the policy exposed by the application and could be provided by the customer or by the ISP. The class diagram of the network package is shown in Figure 2.8.

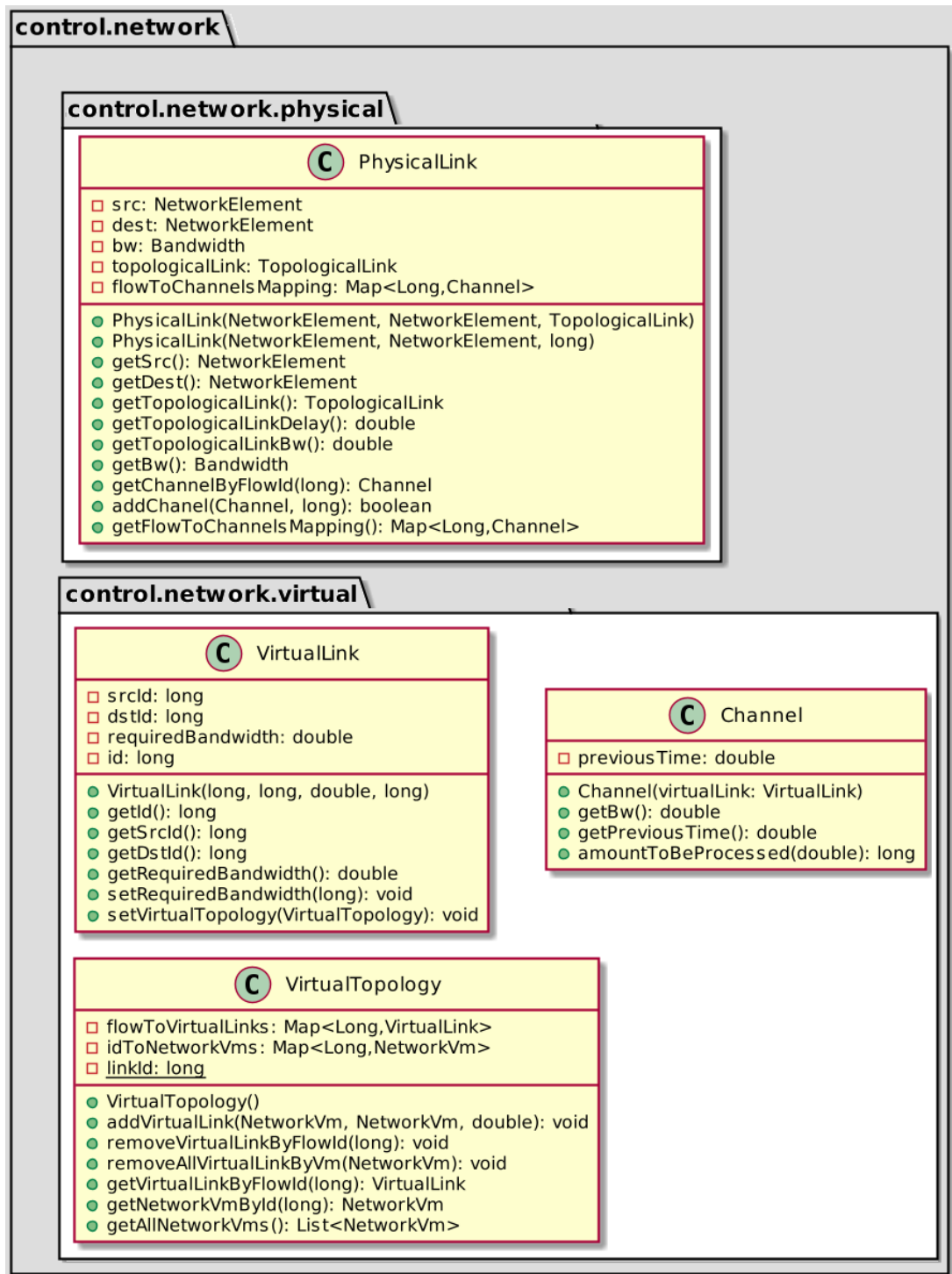


Figure 2.8 - Class diagram of the network package

2.3.4. Controllers

The controllers package contains the interfaces and classes that represent the SDN controllers. The class diagram is shown in Figure 2.9. The Controller interface represents the basic behavior of the controller, while AbstractedController class provides an extensible implementation using Generics and parameterized types to provide a generalized design, that could be used for different network

topology representations.

However, the network topology representation should use the topology package of CloudSim Plus to describe the nodes, links, and their parameters such as bandwidth and delay. Then, the designed models will be used to build the routing tables of the entire network. The ControllerSimple class extends the abstracted controller to provide the customized implementation of the controller functionality regarding the routing tables for the modeled BRITE network topology that is provided by CloudSim Plus.

The SDN controller should have a full view of the undelaying network. For each network element, there is a routing table instance that contains all reachable destinations. The ControllerSimple class builds the routing tables for the tree network topology, where each node is assigned to a level in the tree. Every node should know all directly connected neighbors and all reachable destinations (leaves) in the subtree network in the lower level.

To build the routing tables, the controller starts from the nodes that are located at the lowest level and build the routing tables for its node. Recursively, it visits the higher levels to build the routing tables in their nodes. The activity diagram of adding the directly connected nodes and default routes to the routing table of each node is shown in Figure 2.10. The activity diagram of building the routing tables for the entire network is shown in Figure 2.11, we assume that the tree topology consists of 3 levels, and the nodes in the lowest level are connected to the hosts. The controller will be notified about any changes that happened on the network to rebuild the routing tables.

After building the routing tables, the controller gets the required knowledge about the underlying network of the network elements under its control. Then, it is ready to deploy the application policies and requirements that are provided by the application layer through the A-CPI interface.

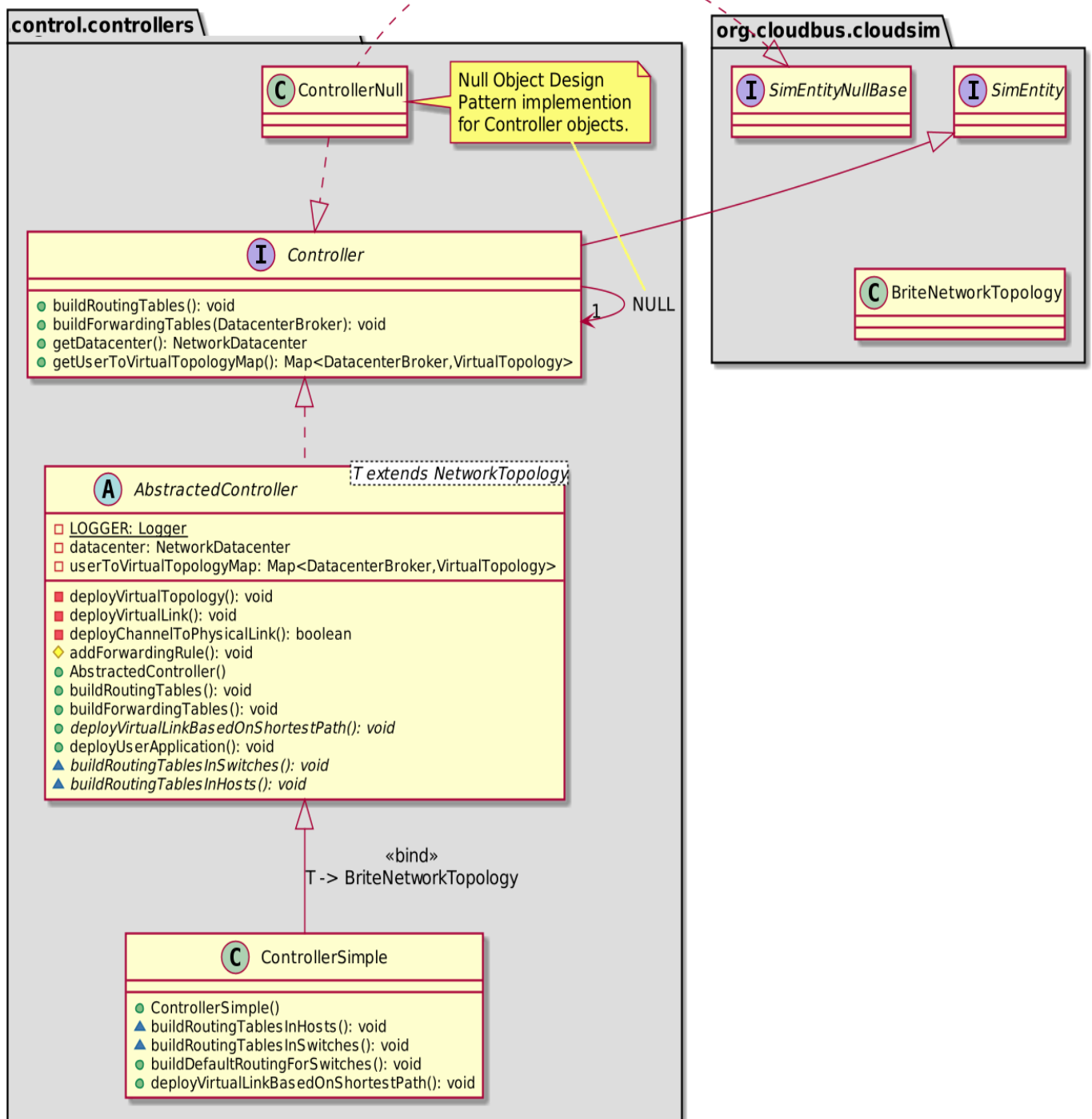


Figure 2.9 - Class diagram of the controller package

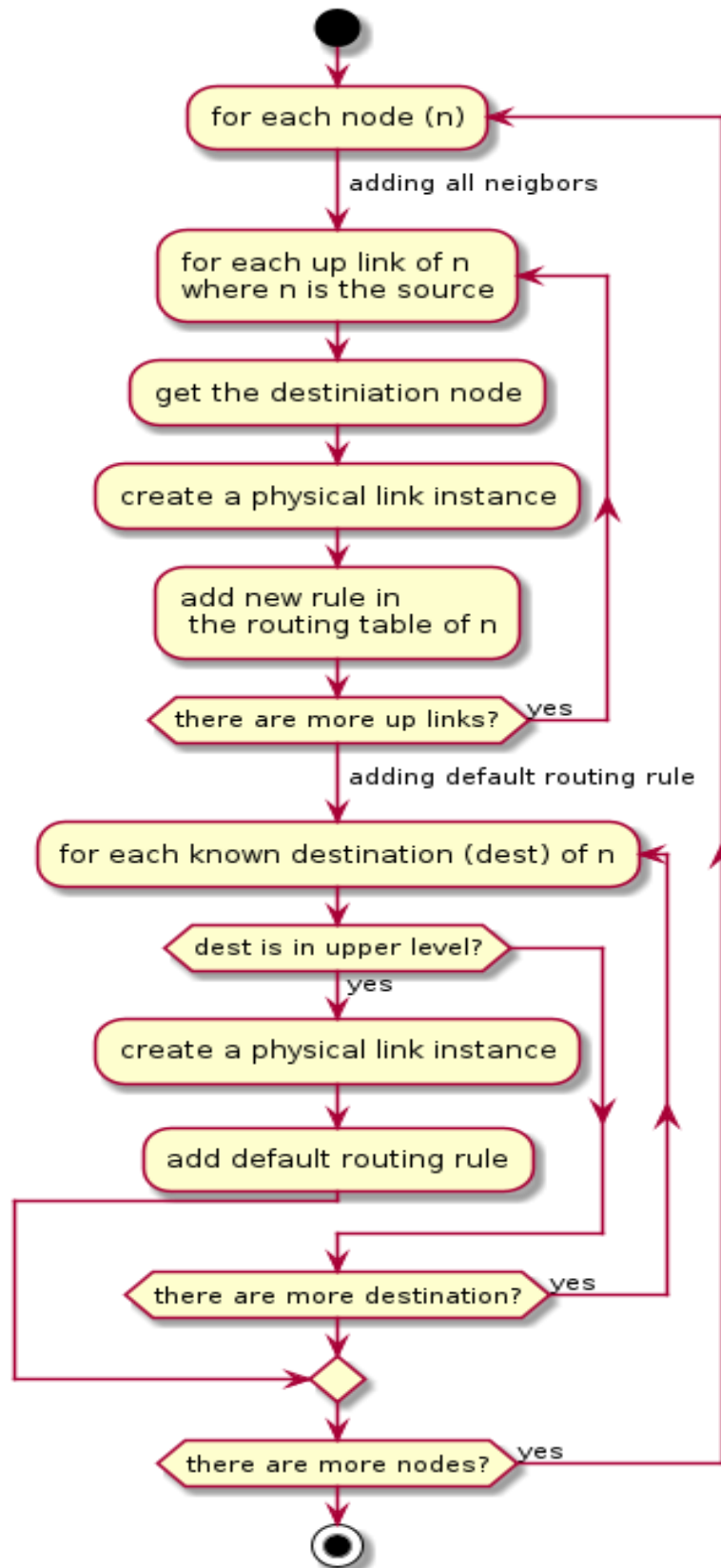


Figure 2.10 - Activity diagram of building the default routing process

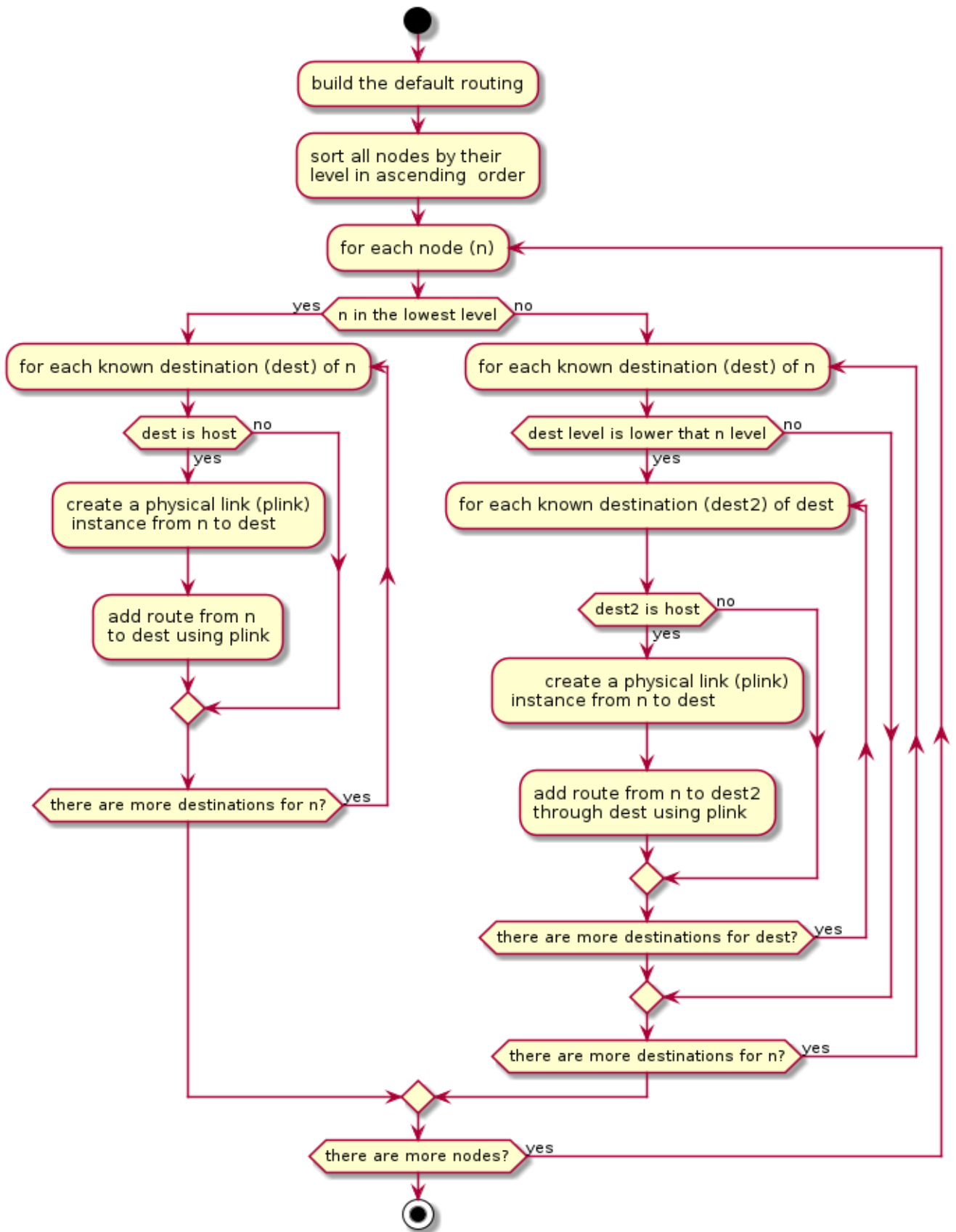


Figure 2.11 - Activity diagram of building the routing tables for the entire network process

Through the deploying process, the controller adds new forwarding rules to the forwarding tables of all nodes that are included by the policy. The definition of the forwarding key plays the main role and it depends on the type and the purpose of the application. In the designed models, the forwarding key consists of source and destination addresses, and a flow identifier. Thus, multiple flows can be owned by the same tenant and belong to one cloud application. Then, the designed tool can simulate multiple users, each user has multiple cloud applications which are running on multiple virtual machines on the same or different servers.

The channel model reflects the deployment of the application policy on the physical links. Each physical link has multiple channels, which are owned by different cloud applications, every channel contains the policy requirements.

In the designed models we assume that the policy requirement is to control the bandwidth which is allocated for a specific flow (cloud application). These requirements may be provided by the user who generates the flow packets, or by the ISP who owns the transport medium.

Through the deployment process, the controller assigns channels to the physical links depending on the defined policy and the available capacity of the physical links. Firstly, it searches for the shortest path between the source and destination nodes. Then, it examines the ability to apply the requirement by searching for the bottleneck through the shortest path. If the required bandwidth is lower than the lowest bandwidth in the discovered path, it creates the required forwarding roles and channels to apply the policy and allocate the required bandwidth, otherwise, it does not apply any changes. The activity diagram of the deployment process is shown in Figure 2.12.

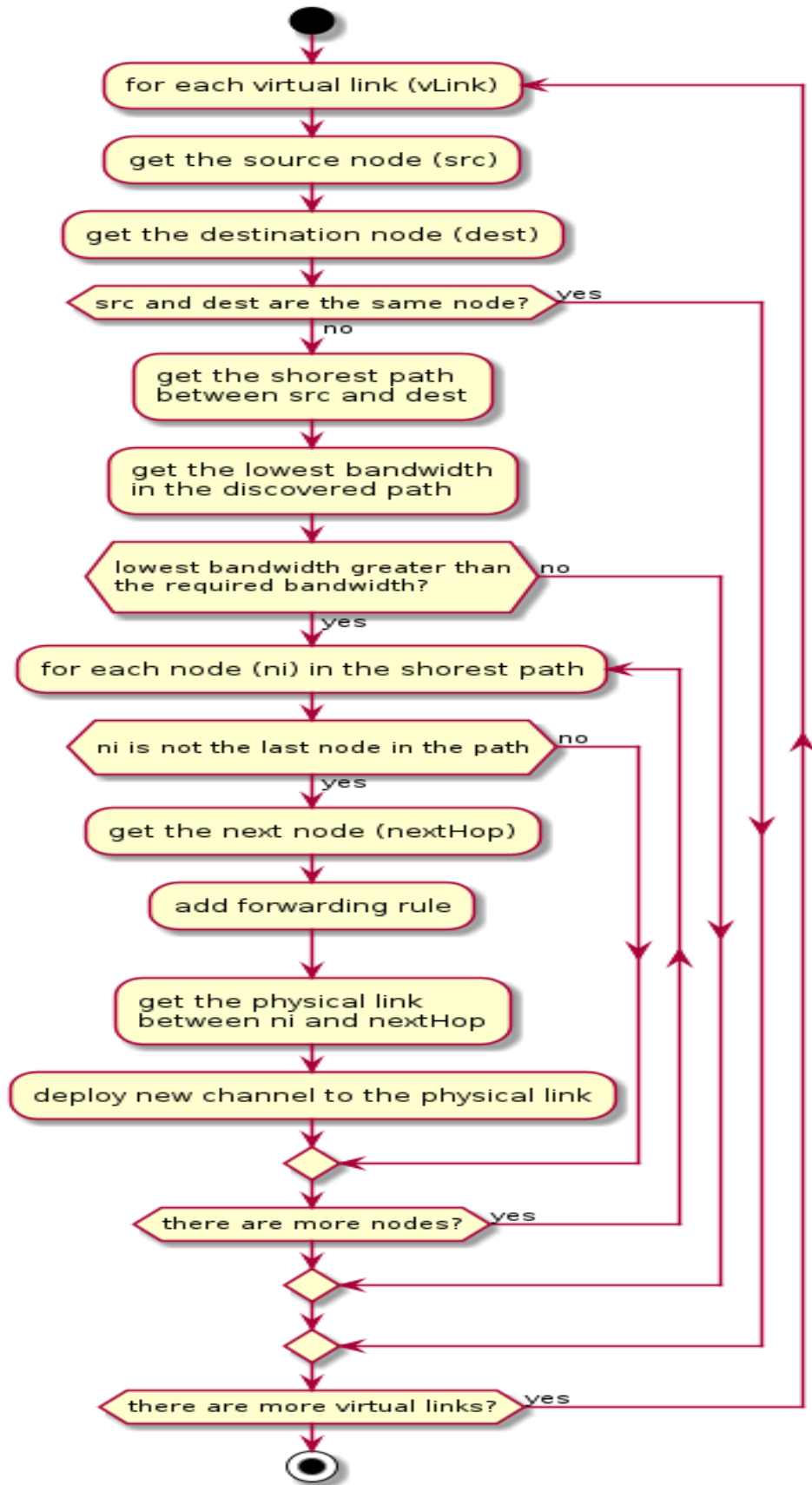


Figure 2.12 - The activity diagram of the deployment process

2.3.5. Switch Power Modeling

Power consumption is one of the parameters which are monitored during the simulation. The switch power modeling package computes the current power usage of the switch in Watts. In the designed tool we provide a simple power model for switches with a linear power profile. At a certain degree of utilization (at a specific time), the current power usage of the switch is computed using the following formula:

$$\text{Power Consumption}_t = S + (P_{active} * N_t) \quad (2.1)$$

where:

S : Static power consumption when the switch is Idle

P_{active} : power consumption per active port

N : Number of active ports in a specific time t .

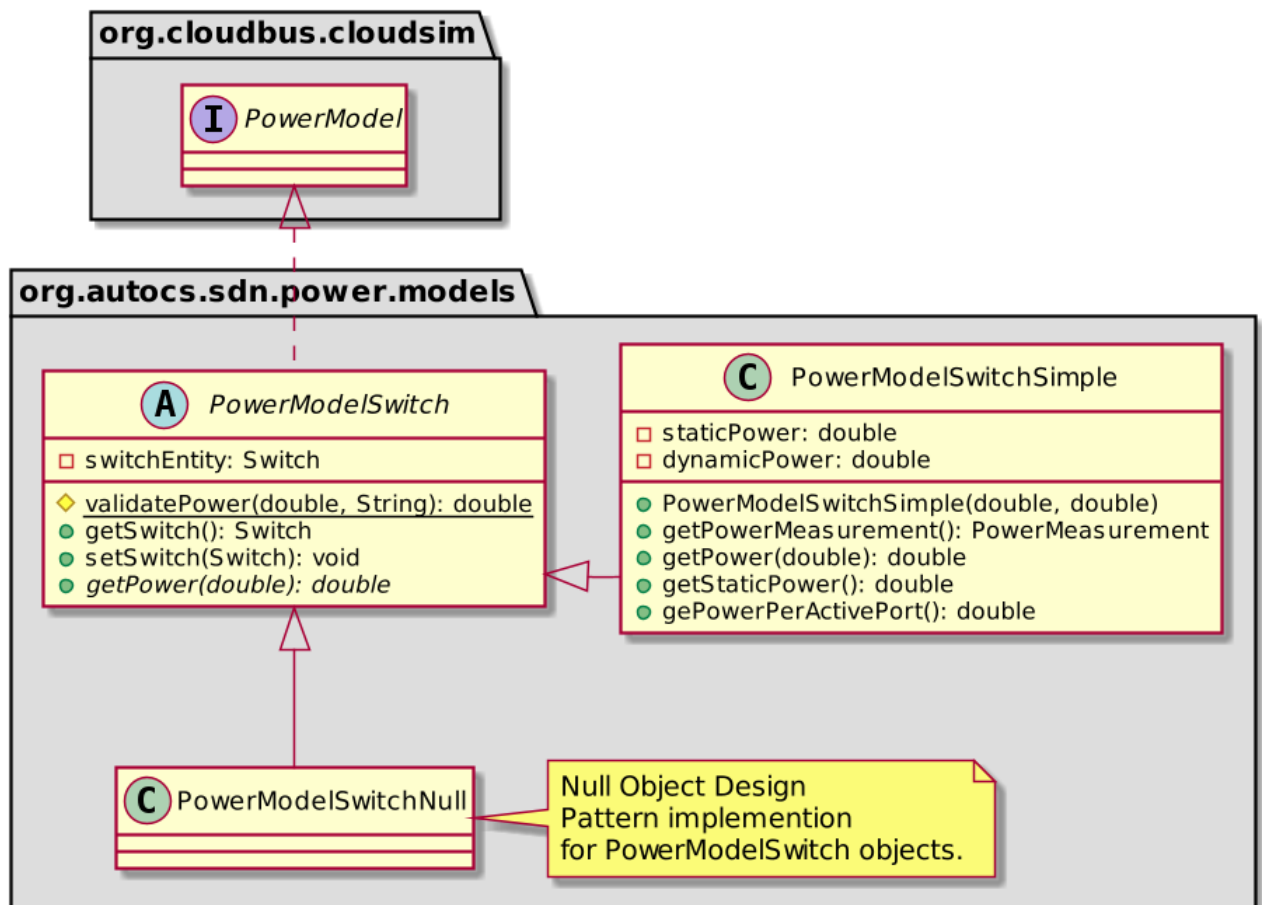


Figure 2.13 - Class diagram of switch power modeling

2.3.6. CloudSim Plus Extension

After the deployment process, the forwarding tables in all nodes are ready and they can process the traffic. The traffic processing requires extending the functionality of some modeling classes of CloudSim Plus. The figures in this section only display the parts of the modeling classes where there are changes in their functionality, and only the added properties and methods are shown.

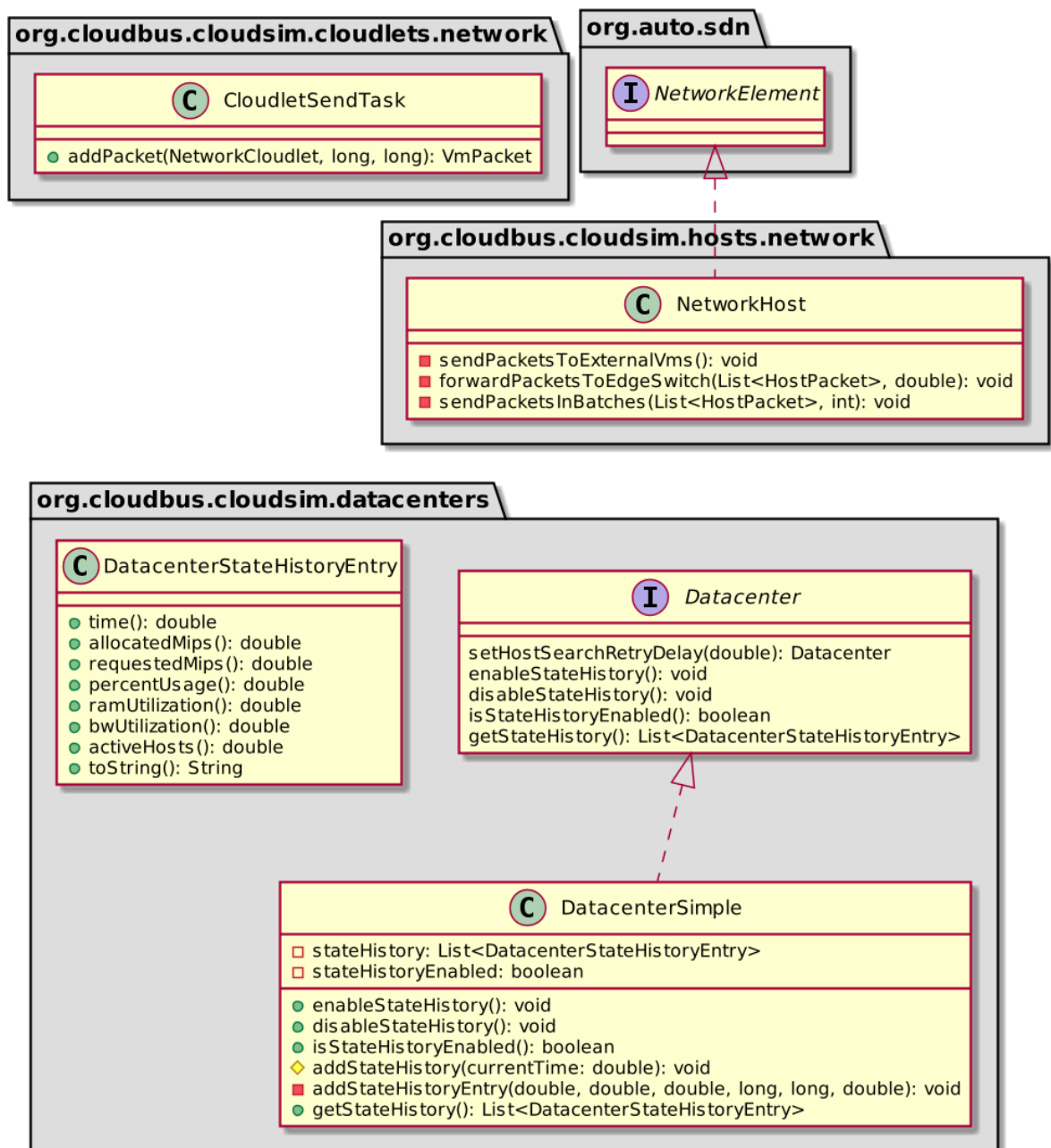


Figure 2.14 - Class diagram of updated classes in the cloudlets, hosts, and datacenters packages

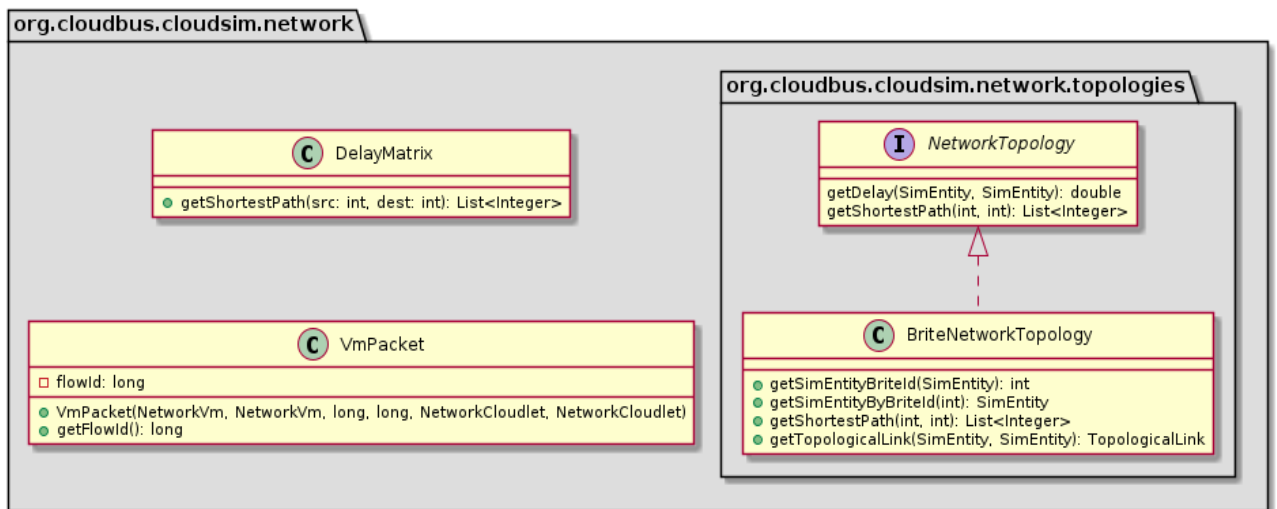


Figure 2.15 - Class diagram of updated classes in the network package

Figure 2.16 shows the updated modeling classes in the switches package which contains multiple classes that represent switches or routers. The significant changes are made in the `AbstractedSwitch` class which contains the logic for traffic processing. The packets are routed using the forwarding table. For each received packet, the forwarding key is constructed using the packet source, destination, and `flowId` fields. Then, the forwarding table is asked for resolving the constructed key. The forwarding table should return the next node where the packet should be forwarded.

Before forwarding the packets, the transmission delay should be calculated. The packet is forwarded to the next node by sending a delayed event to that node. The delay value is calculated based on the capacity of the transmission channel, which is controlled by the applied policy. The packets are divided and forwarded in batches, to make sure that the transferred data through the channel does not exceed the channel capacity (bandwidth). Every batch contains several packets that have the same delay value. In this way, the forwarding process is modeled to simulate the transmission of a real device. The activity diagram of the forwarding process is shown in Figure 2.17.

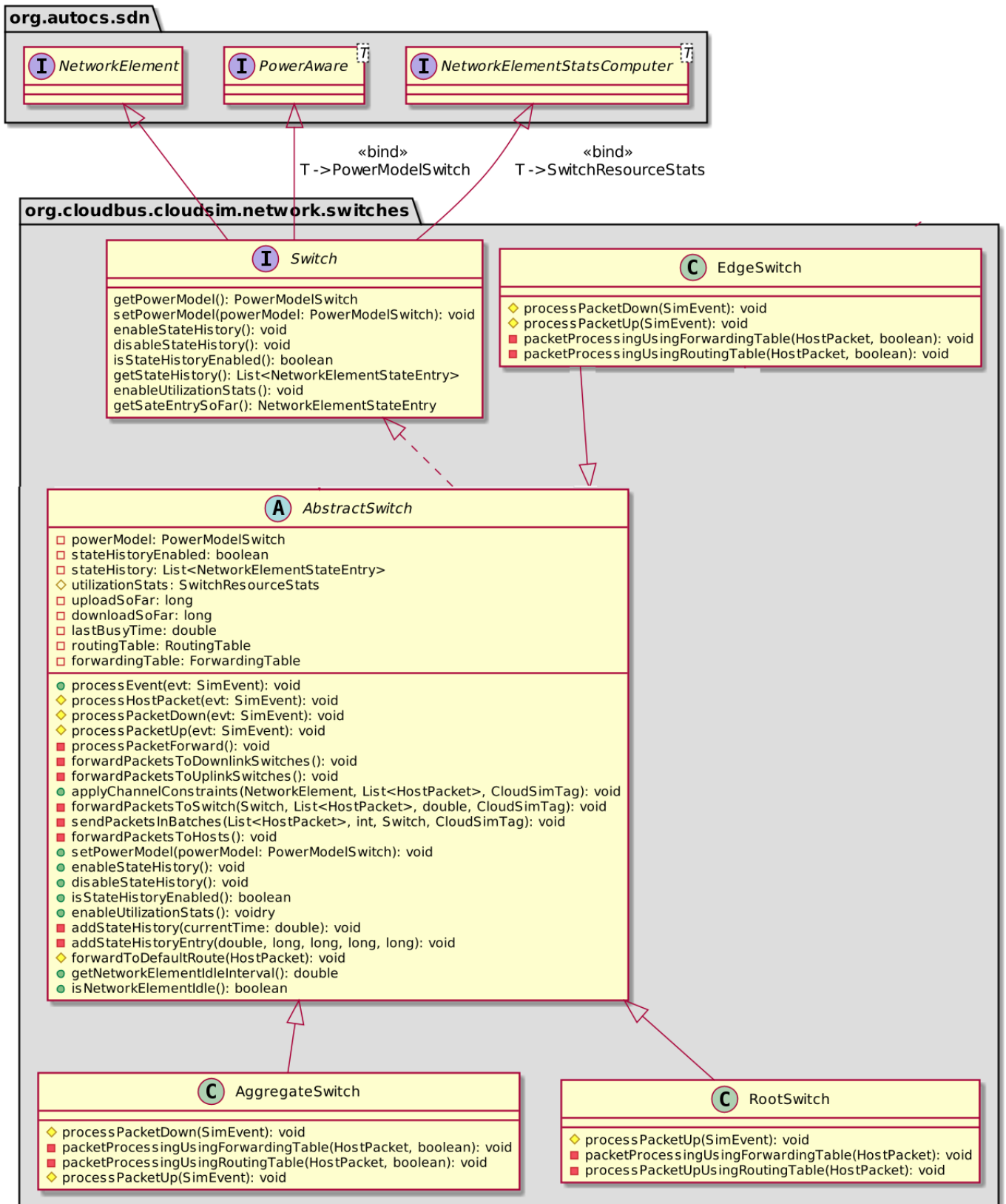


Figure 2.16 - Class diagram of updated classes in the switches package

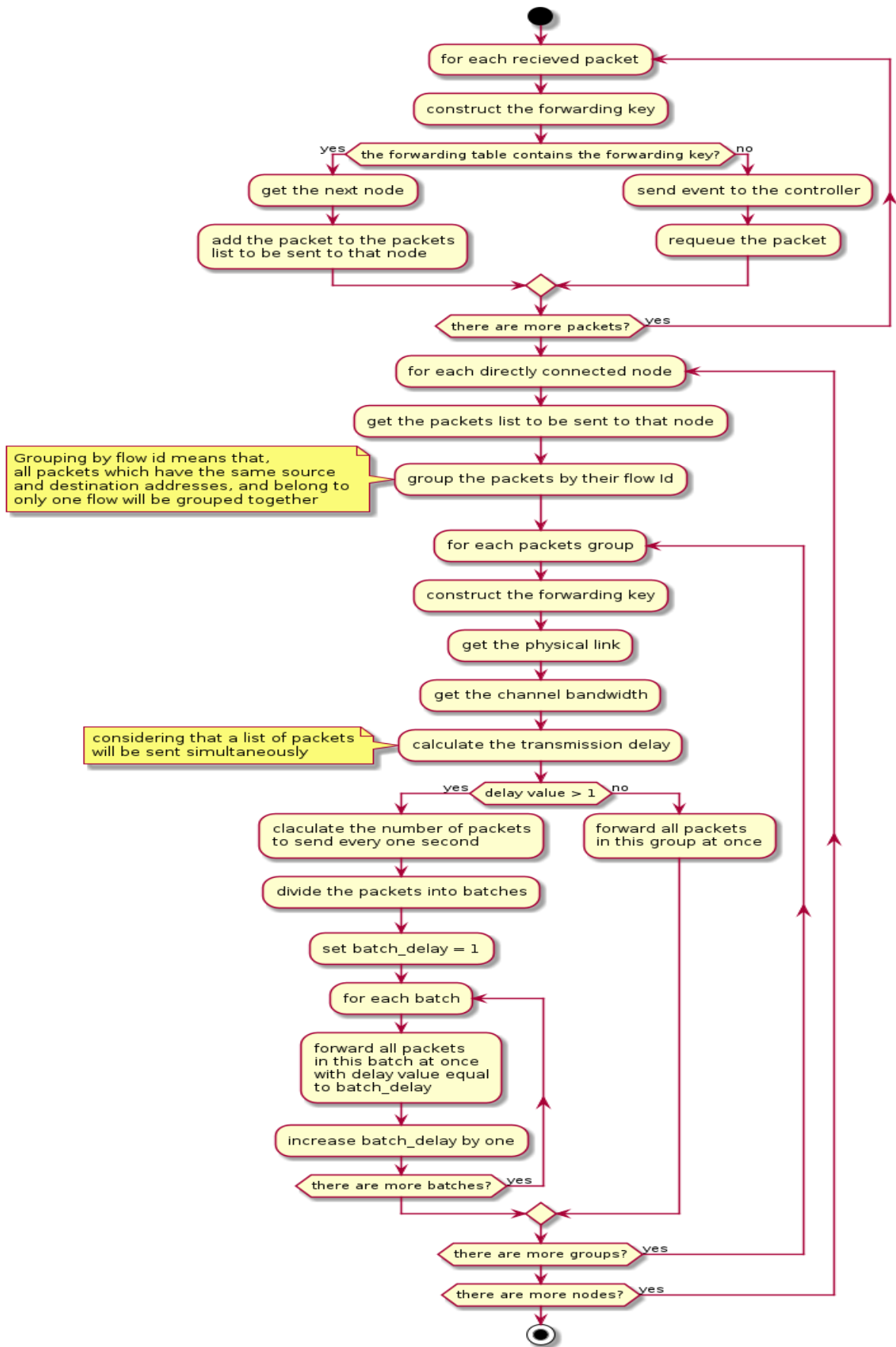


Figure 2.17 - The activity diagram of the forwarding process

2.4. Conclusion

This section describes how to design a tool for modeling cloud computing infrastructure in software-defined networks. Before designing the modeling tool, the key components and processes of the components of the target system were identified. Then, an analysis of the architecture and the network modeling classes, which is provided by CloudSim Plus, is conducted. The analysis aims to identify the critical parts of CloudSim Plus that play the main role in simulating the network of a cloud data center.

The design of the simulation tool follows the object-oriented design patterns to provide an extensible tool. It consists of two main packages. The control package contains the interfaces and modeling classes of the control plane. The data package contains the required entities to extend the functionality of CloudSim Plus.

The routing and forwarding tables play a key role in routing and forwarding the traffic. The controller is responsible for building the routing table by parsing the network topology, the processes of building the routing and forwarding tables are also described. The activity diagrams of the main processes in the designed tool are also provided.

3. Designing a generalized web interface for simulation experiments

Designing and integrating a web interface with the designed simulation tool can help in simplifying and accelerating the process of building, running, and visualizing the results of simulation experiments. In this section, the design of a web API and its integration with the designed simulation tool are explained.

3.1. General architecture

The general architecture of the system is shown in Figure 3.1. The architecture contains a set of loosely coupled services. Every service can be maintained, tested, and deployed independently. The communication between the Frontend and the Backend is carried out using synchronous HTTP/REST API, while the Backend and Engine communicate using asynchronous message queuing protocol.

The main steps to create and run simulation experiments are the following:

1. The user creates and manages the different modeling entities in the Frontend web client through Restful API with the Backend. Also, they can send run requests to execute a simulation scenario.
2. The Backend component manages the modeling entities in the Datastore in response to the user requests.
3. When receiving a run request from the user, the Backend adds a message into the runs queue in the Message Broker
4. The Engine listens and gets the run requests from the runs queue in the Message Broker.
5. The Engine runs the experiment using the Simulation Tool.
6. The Engine adds the result of the experiment into the results queue in the Message Broker
7. The Backend listens and gets the results from the results queue and saves them in the Datastore.
8. After saving the results in the Datastore, the user can explore them in the Frontend.

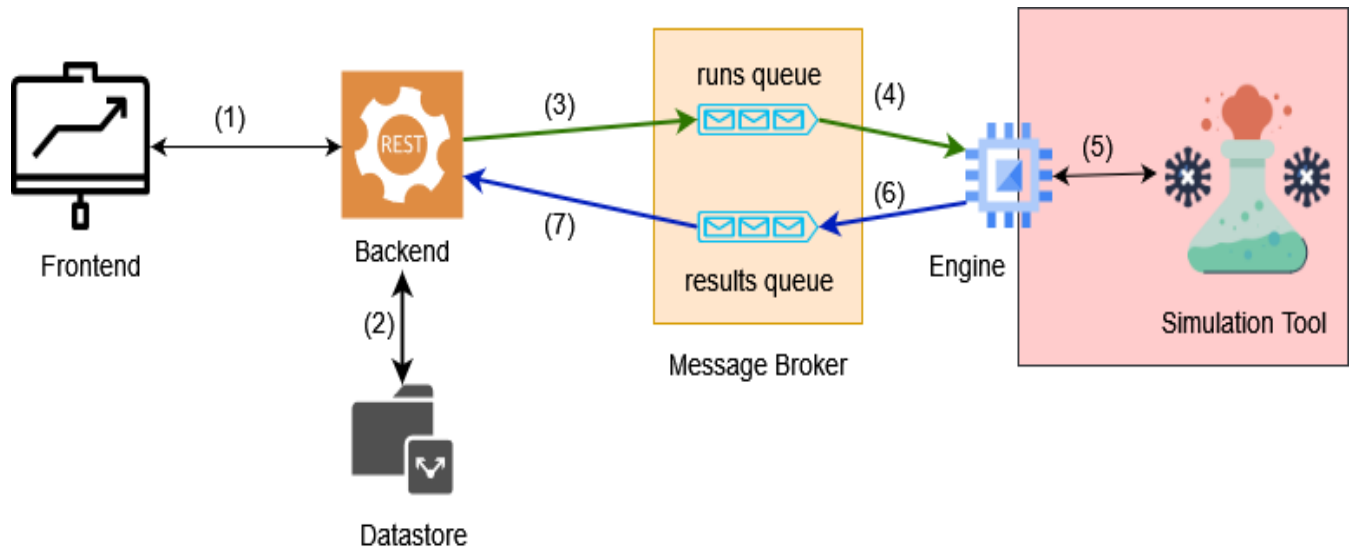


Figure 3.1 - General architecture of the simulation system

3.2. Core component

The designed simulation tool was integrated with the system through the core package. It contains the required serialization functionality for the simulation entities through the web API. Also, it provides the required classes to construct and run the experiment based on the user configuration. The class diagram of the packages of the core component are is shown in Figure 3.2.

3.3. Engine component

The engine component is a lightweight package that is responsible of listening on the runs queue of the Message Broker, get the run requests, execute the experiments using the core component and add the results into the results queue. The class diagram of this package is shown in Figure 3.3.

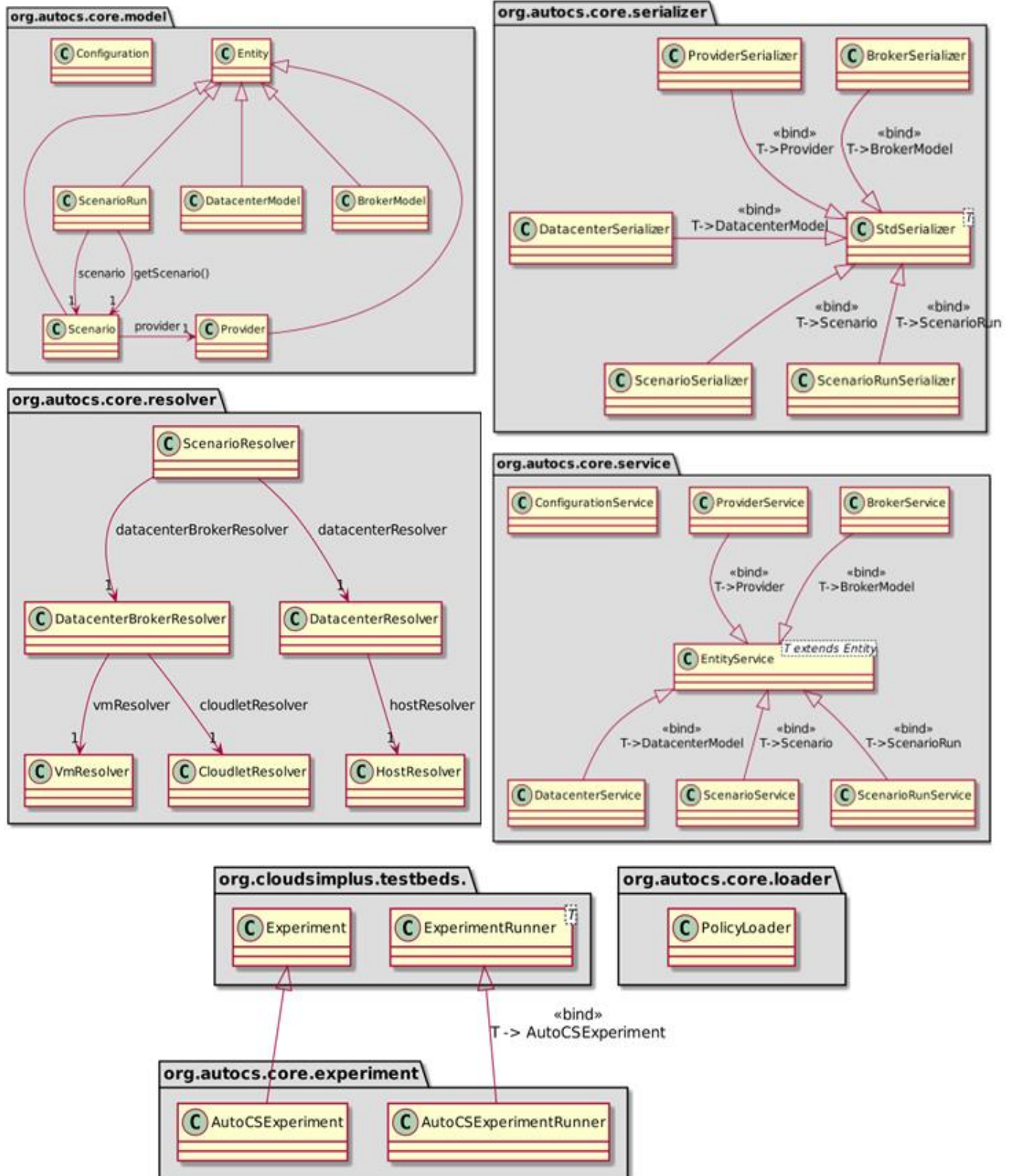


Figure 3.2 - The class diagram of the packages of the core component

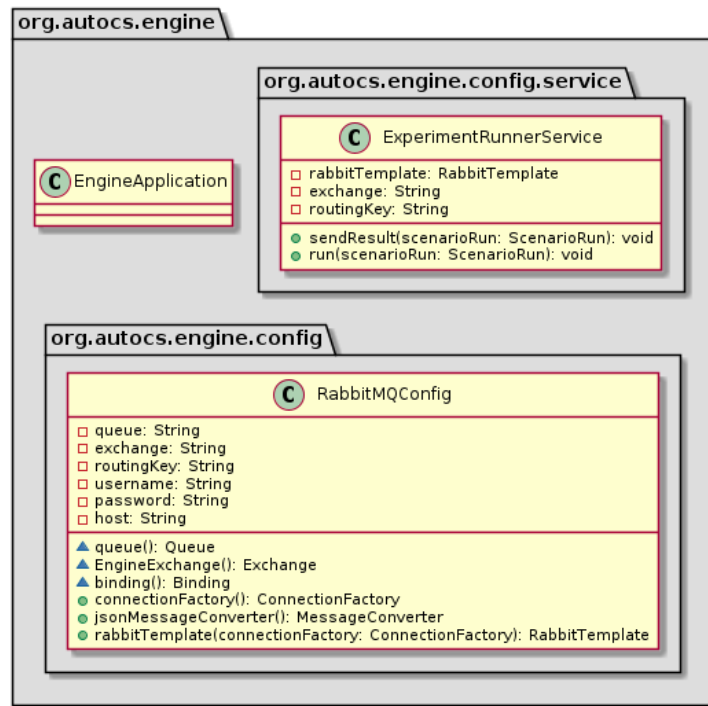


Figure 3.3 - The class diagram of the engine component

3.4. Backend component

The backend component provides a Restful API for the frontend client, through which it can manage the different entities of the simulation experiment. Every entity in the simulation experiment can have a different configuration, such as RAM for hosts or bandwidth capacity for switches. Also, some configuration parameters could be changed, or new parameters could be added to support new added features or algorithms.

To make the system more extensible, the backend component keeps several configuration files, that are used by the frontend component to build configuration forms for the experiment entities. In this way, the separation between the frontend and the simulation tool is achieved. When there is a need to add new parameters to an experiment entity, the configuration file of that entity should be changed. Then the frontend component will reflect these changes on the interface to the user. Figure 3.4 shows an example of a configuration file for the host entity. The configuration section contains a list of all parameters of an entity, and for each parameter, it defines the required information to render a proper field on the frontend.

```

1  {
2    "id": "host",
3    "name": "Host",
4    "description": "A host configuration object",
5    "configuration": {
6      "pes": {
7        "description": "The host's number of processing elements",
8        "type": "int",
9        "defaultValue": 1,
10       "minValue": 1,
11       "required": true
12     },
13     "peProvisioner": {
14       "description": "The provisioning policy used by a host to provide virtu
15       "type": "select",
16       "defaultValue": "Simple",
17       "required": true,
18       "multiple": false,
19       "options": [
20         "Simple"
21     ]

```

Figure 3.4 - An example of a configuration file

The Backend component receives requests to run experiments from the frontend through the Restful API, then it adds a message to the runs queue in the Message Broker. This message contains an instance of ScenarioRun class, which will be serialized through the Message Broker and delivered to the engine component. The class diagram of the backend component is shown in Figure 3.5.

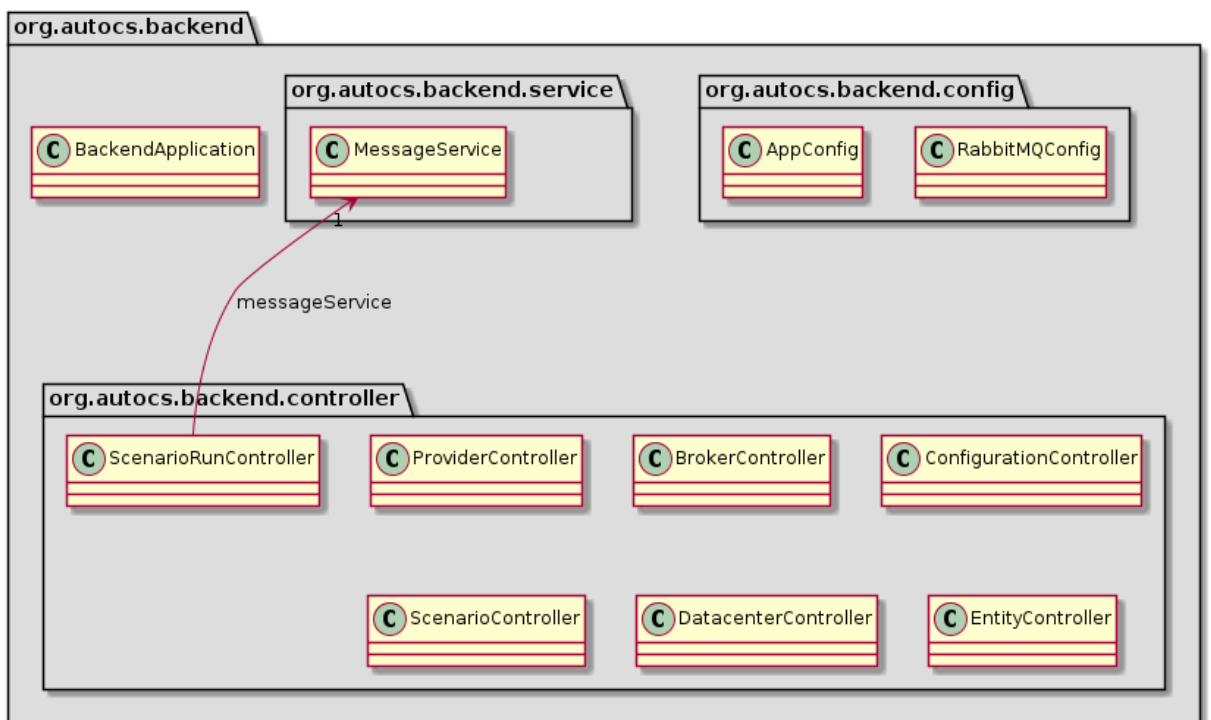
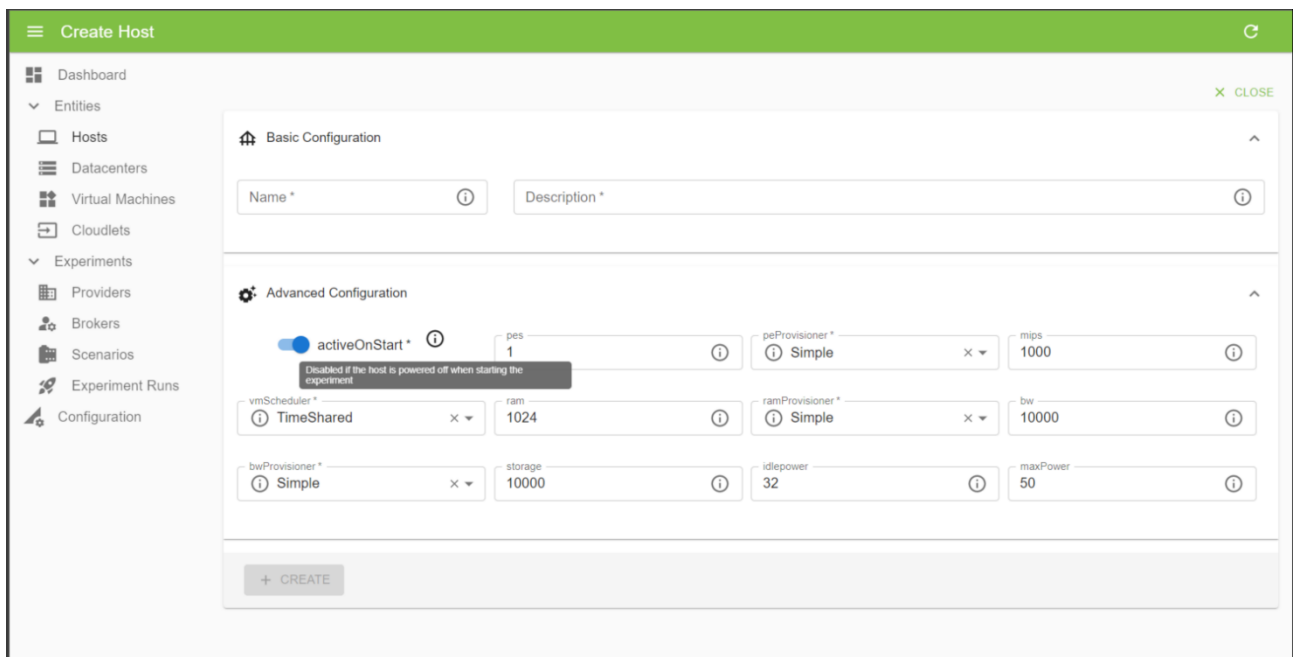


Figure 3.5 - The class diagram of the Backend component

3.5. Frontend component

The frontend component consumes the Restful API, which is provided by the backend component, and provides the required web interfaces to manage the simulation entities and to display the results. The web interfaces are divided into subpages, every page provides one functionality to the user. The user can navigate between the subpages using the left panel.

The web interfaces contain create, edit and list pages for each simulation entity. The create and edit pages render a configuration form for the entity depending on the configuration file. The form displays the entity parameters fields with the required validation and help messages. Figure 3.6 shows the host adding page, a help message is displayed near each field. The form is prefilled with default values that can be changed by the user. If the value is not valid, an error message is displayed.



The screenshot displays the 'Create Host' configuration form. It is divided into two main sections: 'Basic Configuration' and 'Advanced Configuration'. The 'Basic Configuration' section includes fields for 'Name *' and 'Description *'. The 'Advanced Configuration' section includes a toggle for 'activeOnStart *' (currently checked), a tooltip that reads 'Disabled if the host is powered off when starting the experiment.', and several input fields: 'pes' (value 1), 'peProvisioner *' (value Simple), 'mips' (value 1000), 'vmScheduler *' (value TimeShared), 'ram' (value 1024), 'ramProvisioner *' (value Simple), 'bw' (value 10000), 'bwProvisioner *' (value Simple), 'storage' (value 10000), 'idlepower' (value 32), and 'maxPower' (value 50). A '+ CREATE' button is located at the bottom of the form.

Figure 3.6 - Host creation interface

All records of a specific entity are displayed on the list page. It contains a table with multiple columns and for each record there are multiple actions that can be fired by the user, such as edit or view actions. The list page of the hosts is displayed in Figure 3.7. The view action displays the parameter values of a specific

host, as shown in Figure 3.8.

<input type="checkbox"/>	Name	Description	Last modified	
<input type="checkbox"/>	Host type1	Host type1 desc	5/28/2022	● EDIT
<input type="checkbox"/>	host type 2	Host type2 desc	5/28/2022	● EDIT
<input type="checkbox"/>	Host 1	Host 1	5/28/2022	● EDIT
<input type="checkbox"/>	host type 3	Host type3 desc	5/28/2022	● EDIT
<input type="checkbox"/>	Host type 4	Host type4 desc	5/28/2022	● EDIT
<input type="checkbox"/>	Host type 5	Host type 5 desc	5/28/2022	● EDIT

Figure 3.7 - Hosts list page

HOST TYPE1		
PROPERTIES		
Pes	Pe provisioner	Mips
32	Simple	1000
Vm scheduler	Ram	Ram provisioner
TimeShared	2048	Simple
Bw	Bw provisioner	Storage
10000	Simple	1000000
Idlepower	Max power	
32	50	

Figure 3.8 - Host view interface

The basic entities, which are required to build experiment scenarios, are grouped under the (Entities) menu item in the left panel. The user can create and manage different entities with different values of their parameters. In this way, they can reuse the predefined entities in different experiments with different configurations. Figures 3.9 and 3.10 show the creation interfaces for virtual machines and cloudlets (workload).

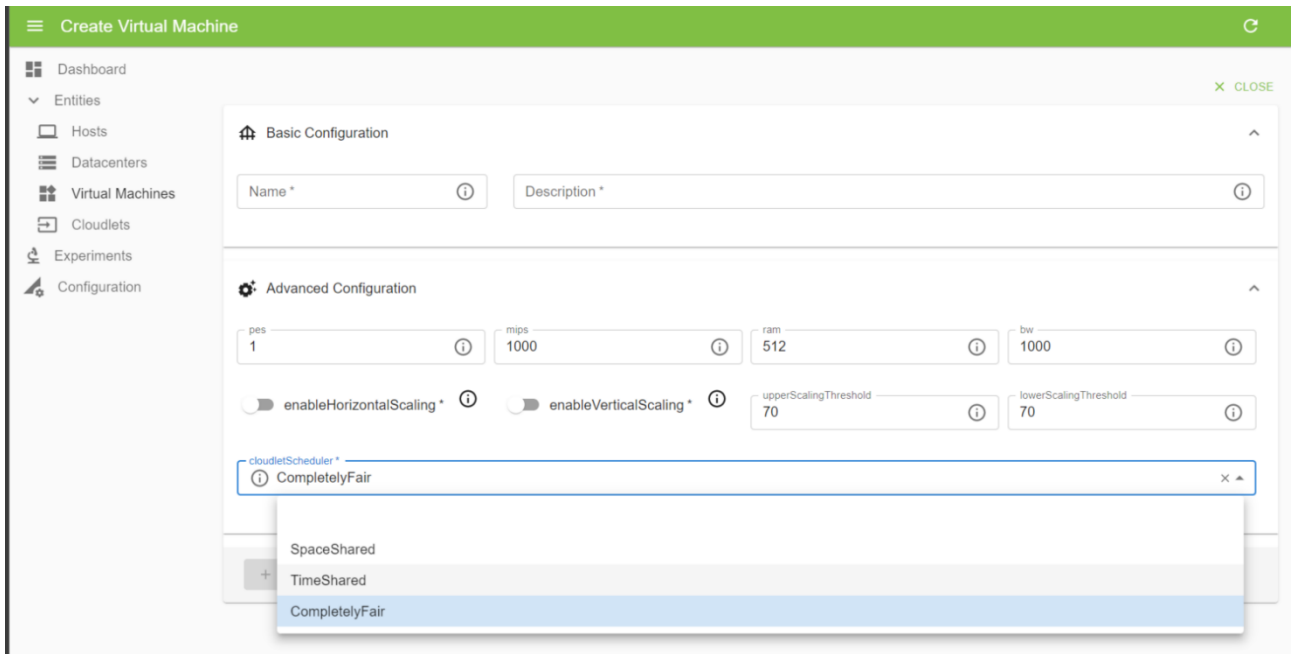


Figure 3.9 - Virtual machine creation interface

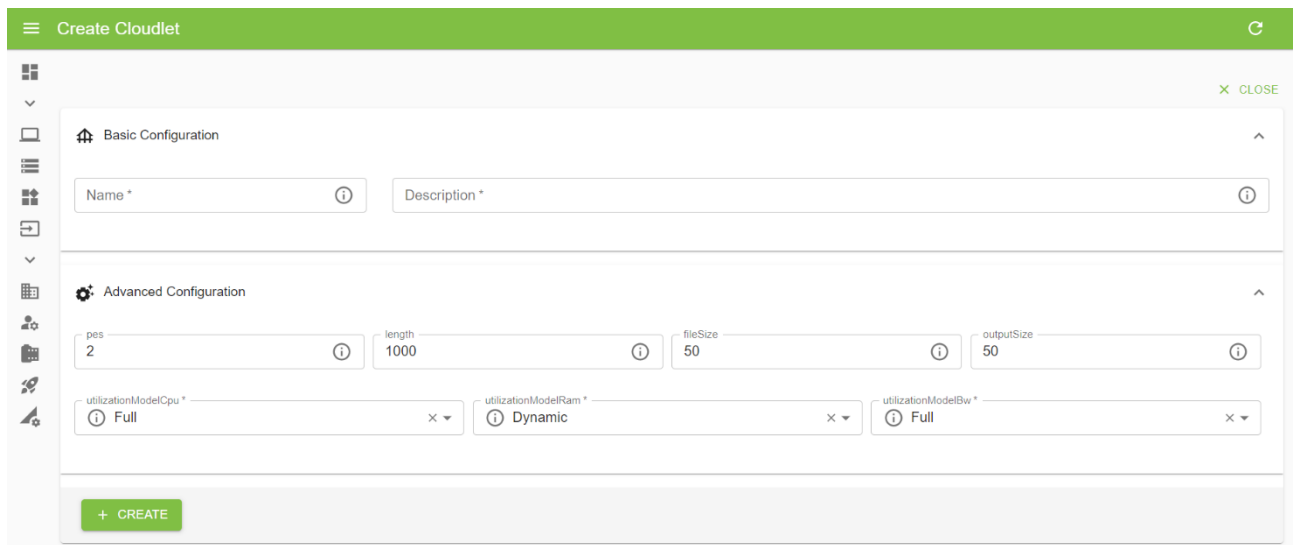


Figure 3.10 - Cloudlet creation interface

Some simulation entities require the creation of nested entities such as the data center, which has a list of hosts. On the data center creation page, the user can select multiple entities of host, and set the number of instances of each host that should be included in the data center. Figure 3.11 shows the data center creation page. After creating the data center, it can be accessed through the list page. The view interface shows some statistics about the created data center, such as the number of hosts, the total processing power, etc. as shown in Figure 3.12.

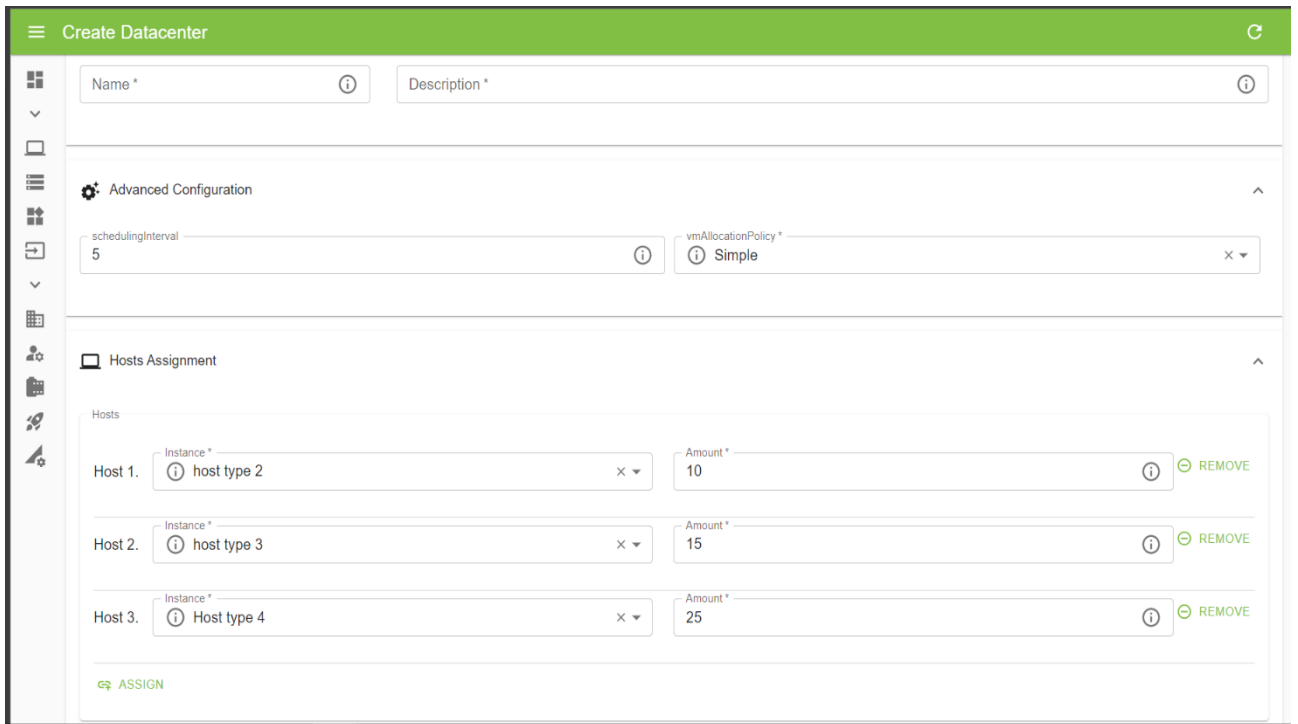


Figure 3.11 - Data center creation interface

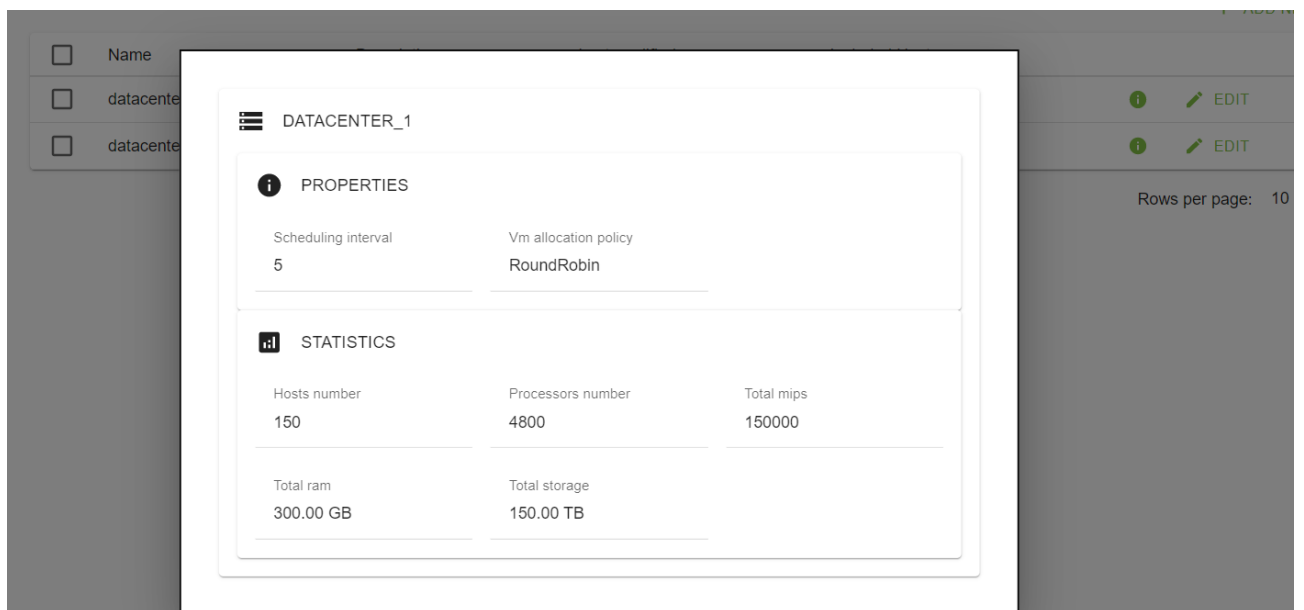


Figure 3.12 - Data center statistics interface

The user can manage the service providers, which own multiple datacenters, by navigating to the providers listing page through the menu item under the (Experiments) item of the left panel. On the creation page, the user can assign multiple data centers to one provider from the predefined data centers list and set their amount, as shown in Figure 3.13.

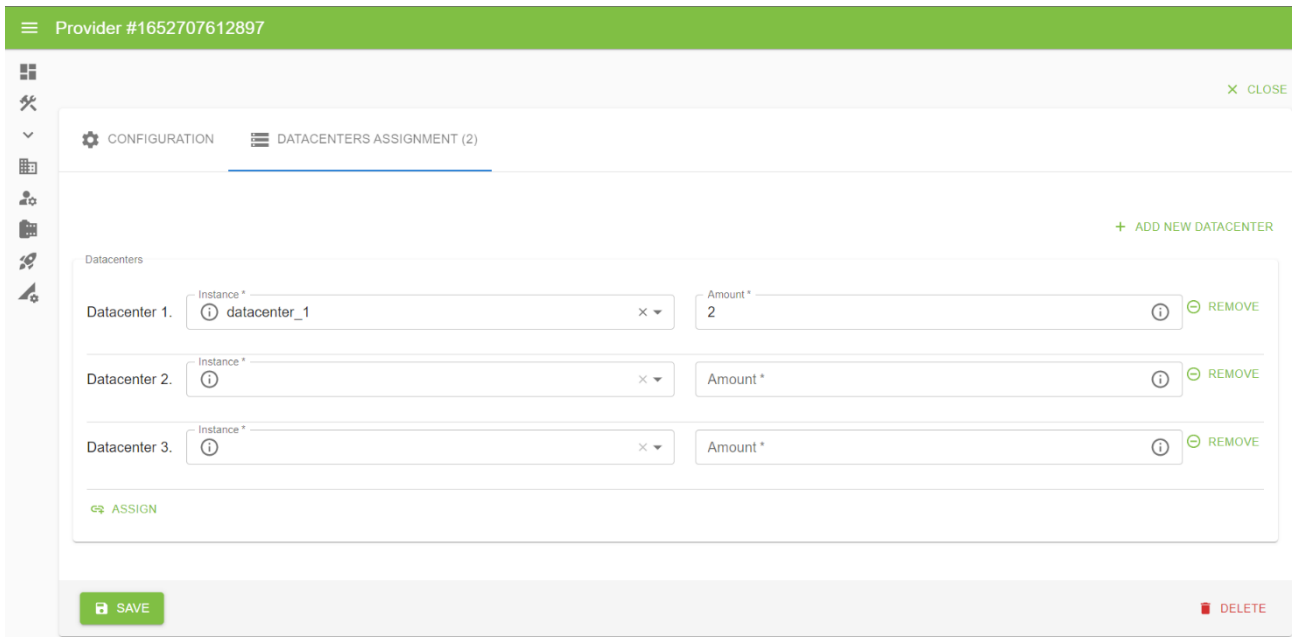


Figure 3.13 - Service provider creation interface

The other interfaces for managing the data center brokers (customers), and building the experiment scenarios are included in the frontend components. After running the experiment, the results are shown using graphical charts.

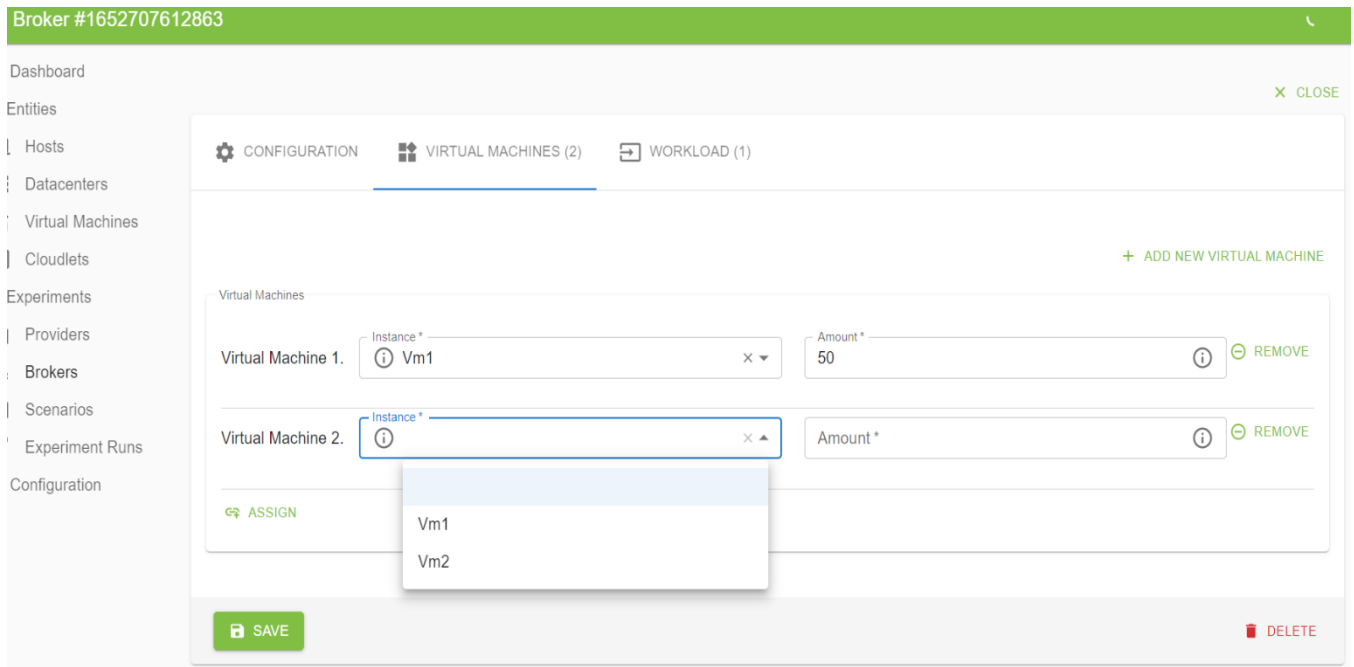


Figure 3.14 - Broker creation page

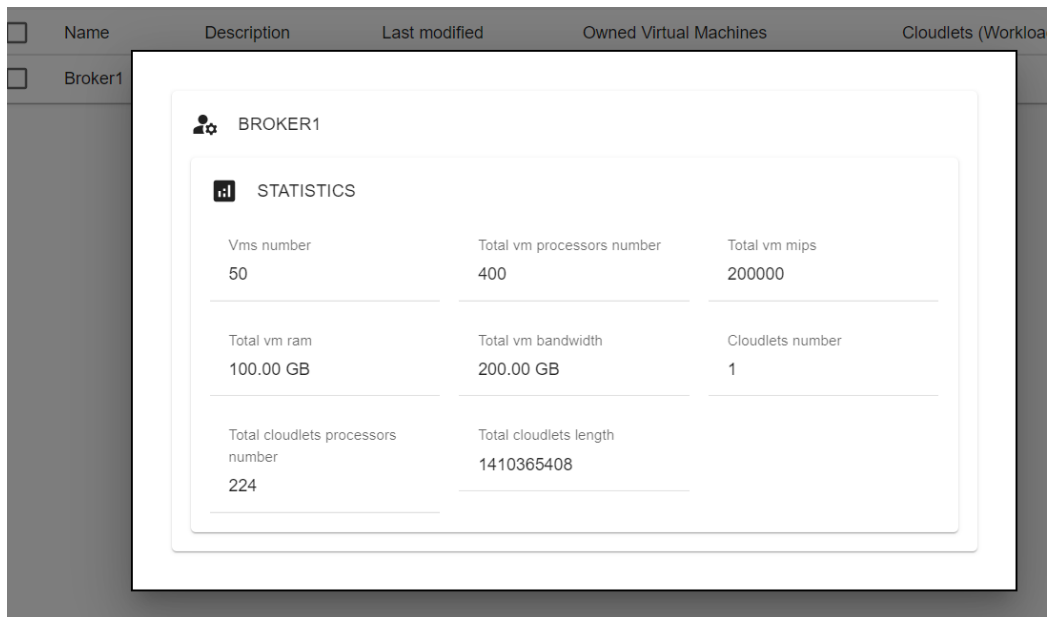


Figure 3.15 - Broker statistics interface

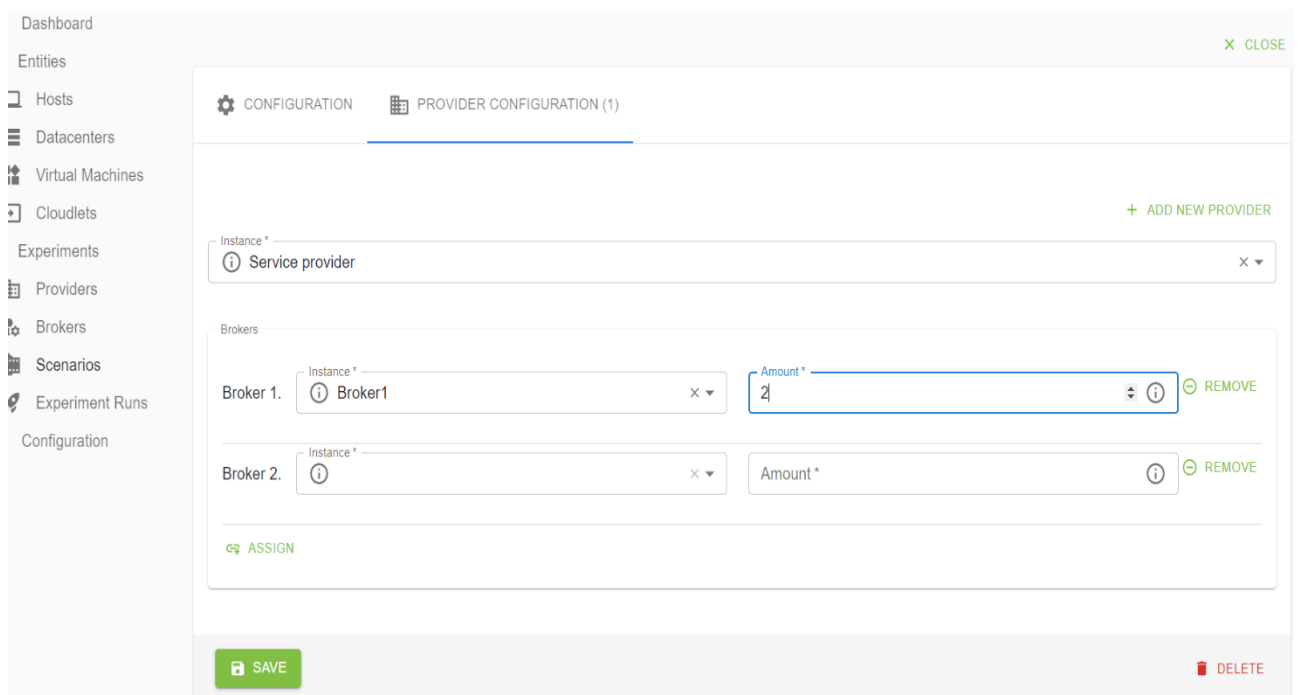


Figure 3.16 - Scenario creation interface

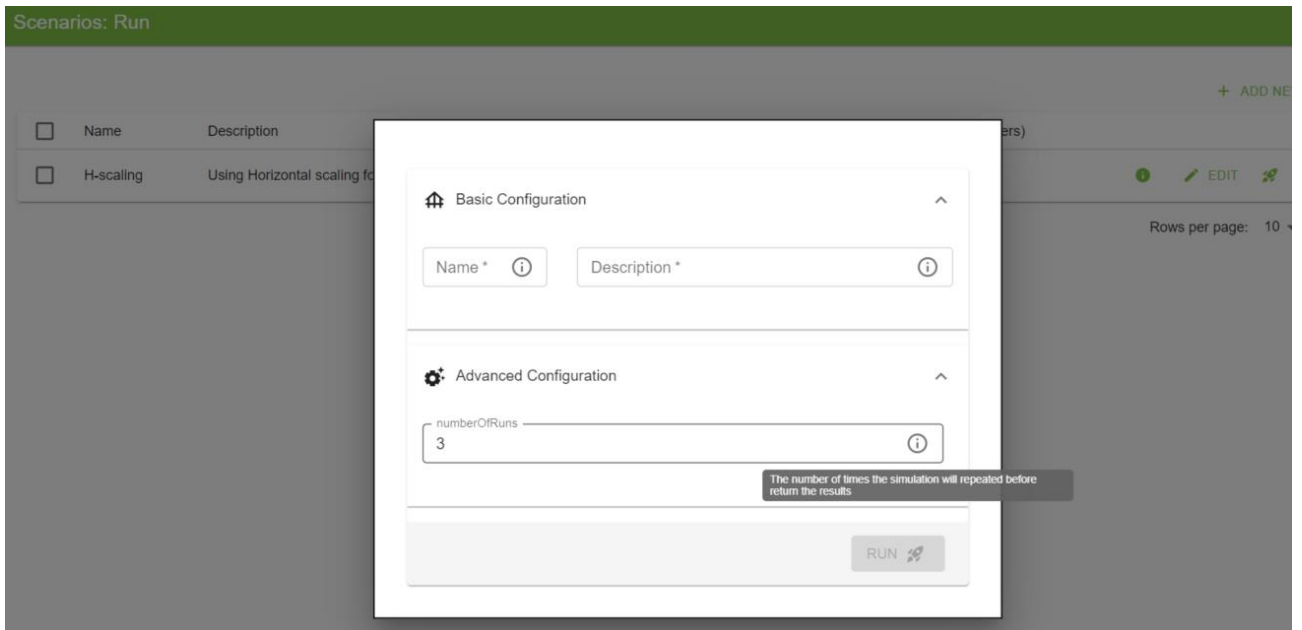


Figure 3.17 - Simulation run interface

5. Finance Management Resource Efficiency and Saving

5.1. Pre-project analysis

5.1.1. Potential consumers of research results

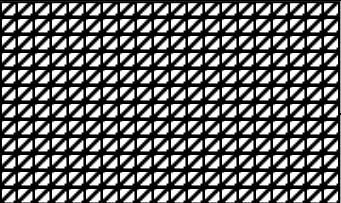




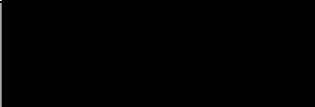

The development to which this work is devoted is a platform for simulating the infrastructure of cloud environments and software-defined distributed networks. Modeling and simulation of equipment, networks, and data center operation is a way to obtain the basic characteristics of a particular algorithm or project. This model is then modeled based on a set of variables to predict its performance in the real world.

According to a recent market research conducted by Frost & Sullivan, \$432.14 billion will be invested annually in the data center market by 2025 [40]. This is more than \$244.74 billion in 2019, which gives an increase of 10 percent. The increased investment will be carried out by data center operators, builders and designers seeking to benefit from the deployment and use of the Internet of Things, Big Data and artificial intelligence.

Strong growth is expected in emerging market economies, with the largest growth observed in the Asia-Pacific region. Due to the wider adoption of cloud computing as a new technology for building infrastructure, the demand for modeling and simulation tools to support cloud data center modeling is expected to increase dramatically in the next five to six years.

The target market of the current development is companies whose main activity is conducting research in the field of energy efficiency, high-speed data transfer within and between data centers and scalability. It is possible to segment the service market according to the degree of need for using these simulation tools. The result of segmentation is shown in Table 5.1.

Table 5.1 - Segmentation map of the Internet resource development services market

		Development application sphere type			
		Energy efficiency	High-speed data transmission	Scalability	QoS
Company Size	Large				
	Medium				
	Small				



Firm A



Firm B



Firm C

5.1.2. Analysis of competitive solutions

Cloud computing modeling tools have been developed to study and evaluate cloud computing technology. These tools include various algorithms and models, so organizations and service providers can modify them according to the solution of the problem before investing in hardware and software. Finding the right tool for this scenario or figuring out what functions each tool has is a difficult task. It is necessary to conduct a comprehensive survey of these tools and their critical assessment.

Surveys [33, 34, 35, 36] have shown that researchers prefer to use and expand open-source cloud simulators such as CloudSim and GreenCloud. In total, more than 20 CloudSim extensions are used in various fields of research. Alshammari [28] reviewed CloudSim extensions and identified four sets of functionalities. Figure 1.4 shows the four sets, and it shows that some extensions are included in more than one set.

There are only three extensions (CloudSimSDN, Edge CloudSim, CloudSim Plus) that have been recently updated. The CloudSimSDN source code is available, except for the GUI part. We choose CloudSimSDN and CloudSim Plus from CloudSim extensions as competitive solutions because they support modeling

of power consumption and network interaction.

In the end, the following products were chosen as competing solutions:

1. GreenCloud (1)
2. CloudSim Plus (2)
3. CloudSimSDN (3)

An expert assessment of the main technical characteristics of these solutions is presented in Table 5.2. As a set of technical evaluation criteria, we consider the following requirements:

1. **Federation policy** is to allow cloud applications to work in heterogeneous clouds in different domains.
2. **Migration policy** is when, what and where to move the VM.
3. **Cost modeling** determines the cost of using a service based on a model or system policies.
4. **Communication modeling** is related to the costs associated with data center communication.
5. **Energy modeling** is the modeling of energy in order to reduce the heat generated in the data center.
6. **Power saving modes** are modes for saving energy consumption in data centers.
7. **Application models** are models supported by the platform for various application components.
8. **Availability** indicates whether the simulator is commercial or available as open source.
9. **Simulation time** determines how long it takes the simulator to perform the simulation.
10. **Parallel experiments** are the conduct of simulation experiments on multiple machines or processors.
11. **Improving user productivity** by providing an easy-to-use graphical interface.

Table 5.2 - Evaluation card for comparing competitive technical solutions

No.	Evaluation criteria	Criterion weight	Scores				Competitiveness			
			S_f	S_{k1}	S_{k2}	S_{k3}	C_f	C_{k1}	C_{k2}	C_{k3}
	1	2	3	4	5	6	7	8	9	10
Indicators for assessing the quality of development										
1	Federation policy	0,08	4	1	4	3	0,32	0,08	0,32	0,24
2	Migration policy	0,1	4	1	4	2	0,4	0,1	0,4	0,2
3	Cost modeling	0,03	4	1	4	3	0,12	0,03	0,12	0,09
4	Communication modeling	0,15	5	4	4	5	0,75	0,6	0,6	0,75
5	Energy modeling	0,1	5	3	4	3	0,5	0,3	0,4	0,3
6	Power saving modes	0,06	4	5	4	4	0,24	0,3	0,24	0,24
7	Application Models	0,1	3	4	3	3	0,3	0,4	0,3	0,3
8	Availability	0,06	5	5	5	4	0,3	0,3	0,3	0,24
9	Simulation time	0,05	5	2	5	5	0,25	0,1	0,25	0,25
10	Parallel experiments	0,01	5	1	5	1	0,05	0,01	0,05	0,01
11	Graphical Interface	0,15	5	5	1	1	0,75	0,75	0,15	0,15
Indicators for assessing the commercial potential of development										
1	Product competitiveness	0,01	5	5	4	3	0,05	0,05	0,04	0,03
2	Market entry level	0,01	1	4	4	1	0,01	0,04	0,04	0,01
3	Price	0,01	5	5	5	5	0,05	0,05	0,05	0,05
4	After-sales service	0,01	1	1	1	1	0,01	0,01	0,01	0,01
5	Financing of scientific development	0,05	5	5	4	3	0,25	0,25	0,2	0,15
6	Time to market	0,02	4	5	5	5	0,08	0,1	0,1	0,1
	Total	1	70	57	66	52	4,43	3,47	3,57	3,12

Based on the analysis, it can be concluded that the vulnerability of competitive technological solutions CloudSim Plus and CloudSimSDN is the lack of a graphical user interface. GreenCloud has drawbacks related to virtual machine migration policy, cost modeling, and parallel execution of experiments.

Based on the analysis, we can conclude that CloudSim Plus is the most powerful competitive solution. This conclusion confirms our decision to use CloudSim Plus as the basis for building our solution on top of it.

5.1.3. SWOT analysis

SWOT analysis is used to study the external and internal environment of the project. The matrix shows the strengths and weaknesses of the project, as well as the opportunities and threats of development based on market analysis and competitive technical solutions.

The first step is to identify the strengths and weaknesses of the project, as well as the opportunities and dangers for its implementation that have manifested or may manifest themselves in the external environment. The SWOT matrix is presented in Table 5.3.

Table 5.3 - SWOT matrix

	Strengths of the research project:	Weaknesses of the research project:
	<p>S1. User-friendly web interface for characterization of various components of simulated data centers.</p> <p>S2. The ability to configure simulation scenarios, run experiments and get results and diagrams from the browser without having to write code.</p> <p>S3. Complete documentation on all configuration parameters.</p> <p>S4. A cloud application that can be used from anywhere.</p> <p>S5. Microservices</p>	<p>W1. The complexity of configuring various data center objects due to the large number of parameters.</p> <p>W2. Incorrect understanding of some parameters can lead to unpredictable simulation results.</p> <p>W3. Lack of a comparison function that can be used to compare experimental results.</p> <p>W4. Inability to cope with all the problems associated with the instability of work on individual cloud solutions.</p>

	architecture that increases scalability, maintainability and provides isolation from failures. S6. An open-source project can attract professional developers to improve it.	
Opportunities: O1. The ability to apply new features to support new areas of research, such as mobile cloud computing. O2. The ability to integrate future new CloudSim Plus features. O3. Receiving financial support from donations.	O1O2S1S2. Expanding the types of experiments that can be conducted. O3S1S2S6. Attracting more attention and expanding the number of participants in improving the code base.	O1O2W1. Increase the difficulty of setting up experiments. O1O2W2. Increase the ambiguity of the system for new users.
Threats: T1. Problems with using cloud solutions.	T1S4S5. Distributed microservices and the use of disaster recovery options can eliminate the impact of problems associated with cloud solutions.	T1W4. The difficulty in considering all the problems that arise when the platform works in the clouds is due to their frequent non-determinism.

In addition, the compatibility of the strengths and weaknesses of the project with the external environment is shown. This is necessary to determine the need for any adjustments to the project. Table 5.4 shows the relationship between the strengths of the project and its implementation options.

Table 5.4 - Interactive project matrix

Strengths of the project							
		S1	S2	S3	S4	S5	S6
Project opportunities	O1	+	+	-	-	0	0
	O2	+	+	-	-	0	0
	O3	+	+	-	-	-	+

Weaknesses of the project					
		W1	W2	W3	W4
Project opportunities	O1	+	+	-	-
	O2	+	+	-	-
	O3	-	-	-	-

Strengths of the project							
Project threats		S1	S2	S3	S4	S5	S6
	T1		-	-	-	+	+

Weaknesses of the project						
Project threats		W1	W2	W3	W4	W5
	T1		-	-	-	+

5.1.4. Assessment of the project readiness for commercialization

It is extremely important for a research project to assess its readiness for commercialization, as well as the level of knowledge necessary for its implementation. Based on the results of such an analysis, it can be concluded that the research project is ready for commercialization, as well as the need to increase the number of members of the project team. The questions list is given in Table 5.5.

Table 5.5 - The form of assessing the readiness degree of the scientific project for commercialization

No.	Name	The degree of elaboration of the scientific project	The level of knowledge available to the developer
1.	The existing scientific and technical reserve has been determined	4	4
2.	Promising directions of commercialization of scientific and technical reserve have been identified	3	5
3.	Industries and technologies (goods, services) for market supply are identified	4	4
4.	The commodity form of the scientific and technical reserve for submission to the market has been determined	3	4
5.	The authors have been identified and their rights have been protected	3	5
6.	An assessment of the value of intellectual property has been carried out	4	2
7.	Marketing research of sales markets has been carried out	3	2
8.	A business plan for the commercialization of scientific development has been developed	2	2

9.	The ways of promoting scientific development to the market are determined	3	3
10.	A strategy (form) for the implementation of scientific development has been developed	4	3
11.	The issues of international cooperation and entry into the foreign market have been worked out	3	2
12.	The issues of using the support infrastructure services, have been worked out	2	3
13.	The issues of financing the commercialization of scientific development have been worked out	3	2
14.	There is a team for the commercialization of scientific development	2	2
15.	The mechanism of implementation of the scientific project has been worked out	5	5
	TOTAL POINTS	48	48

The final values of the elaboration of the scientific project and the knowledge of the developer are 48, which indicate that the degree of readiness of the scientific development and its developer for marketing is above average. They also point out that some aspects of the project were practically not worked out, namely, the analysis of sales markets and the financing of the scientific development commercialization.

5.1.5. Methods for commercialization of scientific and technical research results

Since the prospects for scientific research are above average, not all elements have been studied and investigated. As a result, this is not enough to organize a business (paragraphs 6-8). It can be concluded that for further commercialization of the research project, third-party specialists in this field are required, as well as a more thorough study of the commercialization plan.

The following methods (paragraphs 1-3): patent license trading; transfer of know-how and engineering can help in the commercialization of the project, since the basic scientific and technical basis has been created. The level of elaboration of the scientific project and the level of understanding of the developer are sufficient

to perform the selected components.

5.2. Project initiation

The processes aimed at defining a new research project constitute a group of initiation processes. The objectives of the project, as well as the stakeholders of the project are determined throughout all initiation procedures.

5.2.1. Objectives and results of the project

Before defining the goals, it is necessary to list the stakeholders of the project. Information on stakeholders is presented in Table 5.6. The goals and results of the project are presented in table 5.7.

Table 5.6 - Project stakeholders

Project stakeholders	Expectation of stakeholders
User company	Facilitating the process of evaluating the performance of various components of the cloud computing infrastructure and networks
Developer	Getting a salary
Scientific supervisor, student	Completed final qualifying work
Free Software Community	Open source framework

Table 5.7 - Project goals and results

Objectives:	<ul style="list-style-type: none"> • Explore the various components of cloud infrastructure and networking; • Explore various cloud modeling tools; • Prepare a technical comparison between modeling tools; • Choose the best tool for further development; • Enhance the selected tool by adding new features related to cloud infrastructure and software-defined networks; • Implement a modeling and web interface tool; • Publish the tool as open-source software.
--------------------	--

Expected results:	<ul style="list-style-type: none"> • Web-based modeling tool for modeling cloud infrastructure and software-defined network; • The final qualifying work has been completed.
Criteria for acceptance of the project result:	Successful testing of the functionality in accordance with the functional requirement.
Requirements for the project result	Requirement: <ul style="list-style-type: none"> • Fulfilled all the points of the functional requirement and the requirements for the user interface. • The developed functionality fully corresponds to the design solutions.

5.2.2. Organizational structure of the project

Table 5.8 shows the project team members, their roles, functions, and their labor costs in the project.

Table 5.8 - Project Working Group

No.	Full name, main place of work, position	Role in the project	duty	Labor costs, hour.
1	Botygin Igor Alexandrovich, Associate Professor of the Department of Information Technology	Scientific supervisor	<ul style="list-style-type: none"> • Setting goals and objectives; • Development and monitoring of the calendar plan; • Evaluation of the effectiveness of the results obtained. 	88
2	Mouhamad Ibrahim, Master's student	Engineer	<ul style="list-style-type: none"> • Analysis of the study area; • Software design, development and testing; • Making an explanatory note; 	800
TOTAL:				888

5.3. Planning of scientific and technical project management

5.3.1. Hierarchical structure of the project work

When organizing the process, a complete list of necessary works was determined. The hierarchical structure of the work presented in Figure 5.1 was used as a structure showing the necessary data.

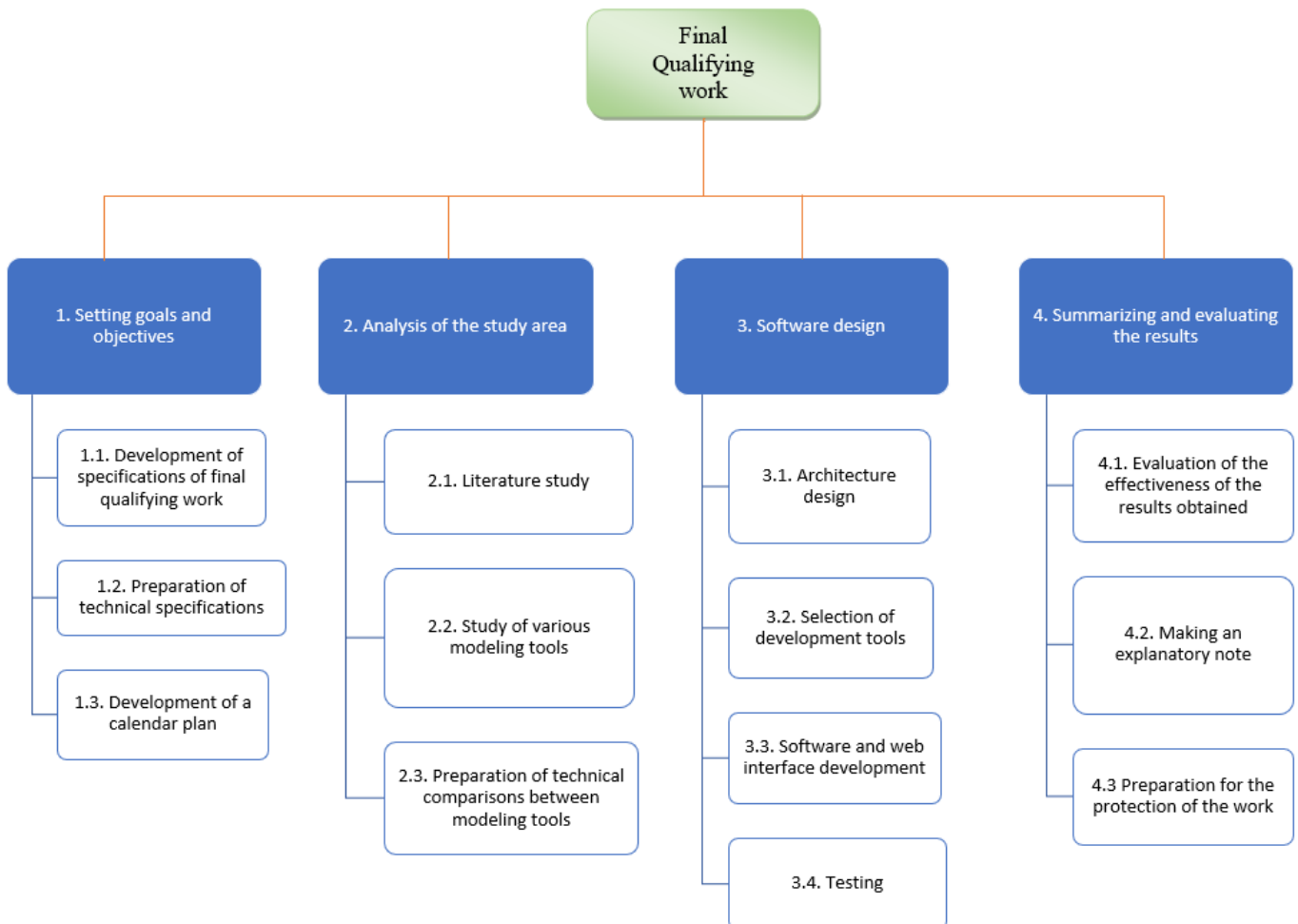


Figure 5.1 - Project hierarchical structure

5.3.2. Project plan

Gantt chart is a type of bar charts (histograms), which is used to illustrate the calendar plan of the project, in which the work on the topic is represented by time-extended segments characterized by the start and end dates of these works. The line graph is represented by the Gantt chart in Table 5.9.

Table 5.9 - Project calendar plan

No.	Task	P	No. day	Work duration																		
				January			February			March			April			May			June			
				1	2	3	1	2	3	1	2	3	1	2	3	1	2	3	1			
1	Setting goals and objectives	S	2	■																		
1.1	Development of specifications of final qualifying work	S, E	5/1	■	■																	
1.2.	Preparation of technical specifications	E	3		■																	
1.3.	Development of a calendar plan	S, E	1/1		■	■																
2.	Analysis of the study area																					
2.1.	Literature study	E	6			■																
2.2.	Study of various modeling tools	E	6				■															
2.3.	Preparation of technical comparisons between modeling tools	E	3					■														
3	Software design																					
3.1.	Architecture design	S, E	1/6					■	■													
3.2.	Selection of development tools	E	4						■													
3.3.	Software and web interface development	E	44							■	■	■	■	■	■	■	■	■	■	■		
3.4.	Testing	E	5																■			
4	Summarizing and evaluating the results																					
4.1.	Evaluation of the effectiveness of the results obtained	S, E	1/2																■	■		
4.2.	Making an explanatory note	E	15																■	■		
4.3.	Preparation for the protection of the work	S, E	1/4																	■		

■ – Scientific supervisor (S)

■ – Engineer (E)

5.3.3. Budget of the scientific research

All expenses necessary for the implementation of the complex of works that make up the content of this development are included in the cost of software development. The following cost components are used to calculate the estimated cost of implementing this development:

- material costs;
- energy costs;
- costs of special equipment for scientific (experimental) purposes;
- depreciation charges;
- basic salary;
- additional salary of scientific and production personnel
- social contributions;
- overhead costs.

Calculation of the material costs of the study

The cost of all materials used during the development is included in this article. The system was created using a personal computer. The development software and other components required for development are freely available and do not require any additional costs. However, minor expenses for various related stationery and printing were also included in this expense item. The price of material resources is determined by the corresponding price tags and Table 5.10 is given. Thus, we get that $C_M = 1478,40$ rub.

Table 5.10 - Calculation of material costs

Item name	Brand, size	Qty	Price per unit, rub.	Total, rub.
Printing paper	A4	1 pkg.	279	279
Printer Cartridge	HP 305	1 pc.	1129	1129
Total for materials				1408
Transportation and procurement costs (3-5%)				70,4
Total for the section C_M				1478,40

Calculation of electricity costs

This category of expenses includes the cost of electricity used to operate the equipment during the project, as defined by the formula:

$$C_{E.eq} = P_{eq} \cdot t_{eq} \cdot T_E, \quad (5.1)$$

where:

P_{eq} : power consumed by the equipment, kW;

t_{eq} : equipment operating time, hour;

T_E : tariff for 1 kW/hour.

For TPU $T_E = 6,58$ rub. for 1 kW/hour (with VAT).

Table 5.11 - Calculation of electricity costs

Name	Equipment operating time t_{eq} , hour	Power Consumption P_{eq} , kW	Costs. $C_{E.eq}$, rub.
Personal computer	888	0,3	1753
Total for the section $C_{E.eq}$			1753

Thus, the cost of electricity is $C_{E.eq} = 1753$ rub.

Basic salary

This section covers executive salary, engineer salary, incentives and surcharges. The calculation is based on the complexity of each stage, as well as on the monthly salary of the performer.

The main calculation of the basic salary was carried out according to the formula:

$$S_{basic} = S_{avg} \cdot T_{day}, \quad (5.2)$$

where:

S_{avg} : average daily salary of an employee, rub.;

T_{day} : duration of work performed by scientific and technical workers, working days.

The following formula is used to calculate the average daily salary of an employee:

$$S_{avg} = \frac{S_M \cdot M}{F}, \quad (5.3)$$

where:

F : valid annual fund of working hours of scientific and technical personnel, working days;

M : the number of months of work without vacation during the year: with a vacation of 48 hours. days $M=10.4$ months, 6-day week;

S_M : monthly official salary of the employee, rub.

Monthly official salary of the employee:

$$S_M = S_b \cdot K_r, \quad (5.4)$$

where:

S_b : base salary, rub.;

K_r : district coefficient (1.3 for Tomsk).

The working time balance is presented in table 5.12.

Table 5.12 - Working time balance

Working time indicators	Supervisor, days	Engineer, days
Calendar number of days	365	365
Number of non-working days		
- days off	52	82
- public holidays	14	14
Loss of working time		
- vacation	45	52
- absenteeism due to illness	–	–
Valid annual working time fund	254	217

The basic salary (Table 5.13) was calculated using the above information, relying on the fact that the supervisor is an associate professor and has a Candidate of technical sciences degree and has a salary of 35111.5 rubles. The salary of the master will be equal to the salary of an assistant without a scientific degree, which is equal to 22695 rubles.

Table 5.13 - The results of the calculation of the basic salary

Performers	S_b , rub.	K_r	S_M , rub.	S_{avg} , rub.	T_{day} , days.	S_{basic} , rub.
Engineer	22695	1.3	29503,5	1414	100	141400
Scientific supervisor	35111,5		45645	1869	11	20559
Total for the section S_{basic}						161959

Thus, the cost of basic wages is 161959 rub.

Additional salary of scientific and production personnel

This section covers the costs of additional wages, taking into account the number of additional payments for guarantees and compensations (when performing state and public duties, when combining work with training, when providing annual paid leave, etc.).

The calculation of additional wages was carried out according to the formula:

$$S_{add} = S_{basic} \cdot K_{add}, \quad (5.5)$$

where:

K_{add} : the coefficient of additional wages (on average – 12%).

Table 5.14 shows the calculation of additional wages for project performers.

Table 5.14 - The amount of additional salary

Performers	S_{add} , rub.
Engineer	16968
Scientific supervisor	2467,08
Total for the section S_{add}	
	19435,08

Thus. the cost of additional wages is 19435,08 rubles.

Deductions for social needs

This expense item reflects mandatory deductions according to the norms established by the legislation of the Russian Federation to the state social insurance bodies (ФСС), pension fund (ПФ) and health insurance (ФФОМС) from the cost of paying employees.

This section of expenses is calculated according to the following formula:

$$C_{social} = k_{social} \cdot (S_{basic} + S_{add}), \quad (5.6)$$

where:

k_{social} : the coefficient of deductions for payment to extra-budgetary funds (pension fund, compulsory medical insurance fund, etc.).

For 2019, Federal Law No212-ФЗ of 24.07.2009 sets the amount of insurance premiums equal to 30%. The calculation of deductions is presented in table 5.15.

Table 5.15 - Contributions to extra-budgetary funds

Performers	S_{basic}, rub.	S_{add}, rub.	C_{social}, rub.
Engineer	141400	16968	47510,4
Scientific supervisor	20559	2467,08	6907,82
Total for the section C_{social}			54418,22

Thus, the cost of insurance premiums is 54418,22 rubles.

Overhead costs

Overhead accounts for all costs not included in the previous items of expenditure: printing and photocopying, payment for the use of services, etc.

The calculation of overhead costs is determined by the formula:

$$C_{over} = k_{over} \cdot (S_{basic} + S_{add}), \quad (5.7)$$

where:

k_{over} : overhead factor.

$$C_{\text{over}} = 0.16 \cdot (161959 + 19435,08) = 29023$$

Thus, overhead costs amounted to 29023 rubles.

Formation of the cost budget of the research project

In this section, the total costs of performing the project are calculated for all the items mentioned above. The calculation is given in Table 5.16. Thus, the expenses for the implementation of the final qualification work are 268066,7 rubles, and the most expensive item turned out to be the costs of basic salaries.

Table 5.16 - Project cost budget

No.	Costs by item						
	Raw materials, materials (less returnable waste), purchased products and semi-finished products	electricity costs	Basic salary	Additional salary	Overhead costs	Deductions for social needs	Total planned cost
1	1478,40	1753	161959	19435,08	29023	54418,22	268066,7

5.4. Evaluation of the economic efficiency of the project

The result of the project is an open-source platform for modeling cloud computing data centers and a software-defined network. This means that anyone can freely use, study, modify and distribute the project for any purpose. These permissions are applied using an open-source license. Thus, economic efficiency is ignored.

The social effectiveness of a scientific project takes into account the socio-economic consequences of the implementation of a scientific project for society as a whole or certain categories of populations or groups of persons, including both the direct results of the project and the "external" results in related sectors of the economy: social, environmental and other non-economic effects.

Table 5.17 - Criteria of social efficiency

BEFOR	AFTER
Lots of modeling tools with predefined types of experiments.	A single general-purpose tool that can be used to identify and conduct various types of experiments.
Writing the expiration code manually with the need for knowledge of the programming language.	A web-based tool with user-friendly interfaces for configuring various components of the experiment.
-	An open-source project that can attract professionals to improve it.

5.5. Conclusion

The study aimed to create a tool that could be used to model cloud infrastructure and software-defined networks. The study was conducted on a computer. According to the SWOT analysis, the project had sufficient strength to capitalize on opportunities and mitigate threats. To commercialize the research topic, we need to enlist the help of other experts and start our own product development company.

A SWOT analysis was built, which describes the strengths and weaknesses of the project, as well as opportunities and threats to the implementation of the project. An interactive project matrix was created to identify compliance and inconsistencies. An assessment of the project readiness for commercialization was carried out, and the development prospects were considered average.

The internal and external stakeholders of the project are identified throughout the initiation phase along with their expectations of the project, its goals, and results. The project schedule is displayed on the Gantt chart, which shows which performer (student or supervisor) completed which tasks and in how many days.

The cost of materials is included in the budget of the engineering project. The basic and additional salaries of the project performers were calculated. The project budget amounted to 268066,7 rubles after deductions for social needs and overhead costs.

6. Social responsibility

Introduction

The purpose of this work is a platform for simulation of a cloud environment and software-defined distributed networks, which will help researchers in evaluating and measuring the performance of approaches to providing network and host bandwidth simultaneously in a data center while avoiding the costs and time to perform these tasks in a real physical environment.

The implementation of the task was to develop software using a personal computer and a local network with Internet access. Software development was carried out in the computer classroom (auditorium No. 417) of the TPU Cybernetic Center.

Thus, this chapter focuses on the rules and regulations of working with computers and determining the main danger of their operation. That is, whether the created conditions are favorable for the health of the developer. Computers affect our vision, posture, and overall health. Thus, if we can properly develop and comply with all measures to ensure healthy and safe working conditions, many harmful circumstances during work can be avoided.

Next, we will discuss in more detail the safety measures related to environmental safety, fire safety, as well as safety in emergency situations, and, consequently, protection from dangerous and harmful factors.

6.1. Legal and organizational issues for safety support during the project development

6.1.1. Special legal norms of labor legislation

The Labor Code of the Russian Federation is used to regulate the rules of labor protection. According to Articles 91 and 111 of the Labor Code of the Russian Federation, a normal working week should not exceed 40 hours. All employees are provided with days off (weekly continuous rest).

The rational organization of work during the entire shift is provided in accordance with state standards and legislative safety standards, which state:

- the duration of the working shift is no more than eight hours;

- according to the internal standards of the organization, employees have the right to a lunch break or rest during the working day.

Employees are paid by the employer organization. Withholding of wages is allowed only in cases defined by Article 137 of the Labor Code of the Russian Federation. If the employee is the delayed payment of wages for more than 15 days, the employee has the right to stop work by notifying the employer in writing.

The employer and his representatives must comply with the norms set out in Chapter 14 of the Labor Code of the Russian Federation, in order to protect the rights and freedoms of man and citizen in the processing, storage, use, and transfer of personal data of the employee. Employees have the right to appeal to the court against any unlawful actions or omissions of the employer in the processing and protection of his personal data. Also, each employee must be trained in safety, electrical safety, and labor protection before hiring.

6.1.2. Ergonomic requirements for the correct location and layout of the work area

Several regulatory documents regulate the organization of the programmer's workspace, as well as the workspace of any other employee who performs long-term work on a personal computer. A desk, an armchair, a display, a keyboard, and a mouse are the main components of a programmer's or operator's workspace. The sitting position is the main working position.

In accordance with the requirements of GOST 12.2.032-78, the fulfillment of the criteria for a workplace in a computer classroom is shown in Table 6.1. The work chair must be rotatable and adjustable in height of the seat and backrest, in the angles of inclination, and in the distance between the backrest and the front edge of the seat. The desktop must be stable and have a solid metal finish that does not allow static electricity to accumulate.

Table 6.1 - Requirements for the researcher's workplace

Requirement	Indoor parameter values
Workplaces should be located in relation to the light openings in such a way that natural light falls from the side, mainly on the left.	Respond
The distance between desktops with video monitors should be at least 2.0 m (in the direction of the back surface of one video monitor and the screen of another video monitor).	Respond
The distance between the side surfaces of the video monitors is at least 1.2 m.	Respond
The screen of the video monitor should be located at a distance of 600-700 mm from the user's eyes, but not closer than 500 mm.	Respond
The reflection coefficient of the desktop surface should be from 0.5 to 0.7.	Respond
The chair should be designed in such a way as to ensure that the employee maintains a reasonable working posture while working at the computer, allowing him to change his posture to reduce static tension in the neck, shoulders, and back, preventing fatigue.	Respond
The optimal seat height is 400 mm.	Respond
The work chair must be lifting and turning, adjustable in height and angles of inclination of the seat and backrest, as well as in the distance between the backrest and the front edge of the seat.	Not Respond
The seat, backrest, and other components of the chair should have a semi-soft surface with a non-slip, slightly electrified, and breathable coating that is easy to clean.	Respond

The workplace in the computer classroom (auditorium No. 417) of the TPU Cybernetic Center does not meet the requirements for a work chair. The chair in the classroom is not lifting and turning. The remaining requirements for the employee's workplace are met.

6.2. Industrial safety

The main sources of harmful and dangerous factors are electronic computing equipment and elements of the office electrical network since all work on the development and operation of the solution is carried out in a room with constantly working electronic devices. Table 6.2 lists potentially dangerous and harmful production parameters observed during this study.

Table 6.2 - Possible dangerous and harmful production factors

Factors (GOST 12.0.003-2015)	Work stages		Regulatory documents
	Development	Operation	
Deviation of microclimate indicators	+	+	SanPiN 1.2.3685-21 «Hygienic standards and requirements for ensuring the safety and (or) harmlessness of environmental factors for humans»
Insufficient illumination of the working area	+	+	SP 52.13330.2016 «Natural and artificial lighting. Updated version of SNiP 23-05-95»
Exceeding the noise level	+	+	GOST 12.1.003-83 «Occupational safety standards system. Noise. General safety requirements»
Increased magnetic field strength	+	+	GOST 12.1.002-84 «Electric fields of industrial frequency»

6.2.1. Analysis of hazardous and harmful production factors

6.2.1.1. Deviation of microclimate indicators

Microclimate refers to the climate of a relatively limited area, especially when it differs from the surrounding climate. The climatic conditions of the workplace must be met to avoid damage to any physical equipment as a result of overload, as well as damage to people in this area, such as suffocation.

The three factors that determine the microclimate of a particular area are:

- air temperature;
- relative humidity of the air;
- the speed of air movement.

The combination of these three criteria creates a pleasant working atmosphere, which is necessary for the safety of all participants. Each workplace must have certain ideal microclimate parameters, and they must comply with hygienic norms and rules.

Permissible microclimatic conditions are determined by the requirements for the permissible thermal and functional state of a person during an eight-hour work shift. They are established when the optimal value cannot be achieved due to technological limitations, technical, and economic justifications. Table 6.3 shows the permissible values of indicators, relative humidity, and air velocity for work performed while sitting and not requiring systematic physical exertion (category Ia).

Table 6.3 - Permissible values of microclimate indicators at workplaces of industrial premises (category Ia) [41]

Year period	Air temperature, °C		Relative humidity of the air %	Air velocity, m/s
	range below optimal values	range higher than optimal values		
Cold	20,0-21,9	24,1-25,0	15-75	0,1
Warm	21,0-22,9	25,1-28,0	15-75	0,1-0,2

When it comes to ensuring that the microclimate in the workplace is within acceptable limits:

- the difference in air temperature in height from the floor level (0.1; 1.0; 1.5) m should be no more than 3 °C [41].
- the horizontal air temperature drop, as well as its changes during the shift, should not exceed 4 °C for work categories Ia and Ib [41].

Thus, to maintain a certain microclimate in these specific conditions, a heating or air conditioning system corresponding to the climate in this particular area is used.

6.2.1.2. Insufficient illumination of the working area

It is extremely important for us to have adequate working conditions during work, and therefore lighting the work area is an essential aspect of ensuring that a person feels comfortable at work. Lighting is one of the most important factors that an employer should take into account since poor lighting can lead to employees feeling tired, psychological, and physical stress will increase. All this is caused by the semi-darkness of the environment, although the light coming from the computer is quite bright. As a result, a person may lose his eyesight.

Natural light is obtained from the sun, and also refers to any light that is scattered naturally. It usually varies depending on the time of day, month of the year, location, and the region in which it is emitted. Artificial lighting is obtained from electrical sources. When natural light is not enough, artificial lighting is used. The use of both natural and artificial light results in a combined light source.

The natural illumination coefficient (CEO) should be 3.5% in the case of overhead or combined lighting and not less than 1.2% in areas with stable snow cover, and not less than 1.5% in the rest of the territory.

The illumination on the surface of the table in the area where the working document is placed should be 300-500 lux [49], and this light should not create glare on the screen surface. In this case, the light emitted by the screen should be more than 300 lux. When using a reflected lighting system, the brightness of the glare on

the PC screen should not exceed 40 cd/m², and the brightness of the ceiling should not exceed 200 cd/m².

In artificial lighting, first of all, fluorescent lamps of the LB type should be used. Luminescent ceiling lamps with mirror arrays and high-frequency triggers are required. It is not allowed to use lamps without diffusers or filter grids.

6.2.1.3. Exceeding the noise level

Noise in the workplace is one of the most common risks, and it causes a lot of problems. First of all, this is due to working equipment, as well as external noise. Noise can take various forms and can affect the human in various ways.

Equipment noise, sometimes caused by multifunction devices and printers of some types, always exceeds the specified value and must be performed in a place that does not affect the working environment or people in a particular work organization. Therefore, in order to reduce noise and vibration caused by certain equipment and devices in the workplace, they must be installed in a place that does not affect the working environment or people in a particular work organization, must be installed on certain special grounds, as well as with shock-absorbing pads, which are described in the relevant regulations.

The noise level allowed for users of personal computers at any workplace should not exceed the value prescribed by the norms and rules and should not exceed 50 dBA [42].

6.2.1.4. Increased magnetic field strength

In the workplace, the source of increased electromagnetic field intensity is a personal computer (PC). This is due to the fact that PCs are supplied with mains filters, uninterruptible power supplies, and other components which in combination create a complex electromagnetic environment in the user's office. Many studies have linked health problems and even pathological diseases, such as headaches, decreased ability to concentrate, decreased blood pressure, functional deterioration of vision, the development of cataracts and skin lesions, with electromagnetic radiation emitted by the PC system unit.

The developer spends a lot of time in front of the computer, which means that the developer is exposed to an electromagnetic field for a long time. The immunological, neurological, and endocrine systems are most vulnerable to electromagnetic radiation. Depending on the frequency of these radiations, hygienic regulation of electromagnetic radiation is based on various principles.

The electric field strength is a criterion for the industrial frequency (50 Hz). The duration of a person's stay is regulated depending on the intensity of the electric field.

In accordance with GOST 12.1.002-84 [43]:

- maximum permissible level of the electric field strength is set to 25 kV/m;
- staying in an electric field with a voltage of more than 25 kV/m without the use of protective equipment is not allowed;
- staying in an electric field up to 5 kV/m is allowed during the entire working day;
- staying in an electric field from 5 to 20 kV/m is allowed during:

$$T = \frac{50}{E} - 2, \text{ hour} \quad (6.1)$$

where:

E: the electric field strength in the working place.

Then the electric field strength in the working area on a normal working day (8 hours) should not exceed 5 kV/m.

6.2.2. Justification of measures to reduce the levels of exposure to dangerous and harmful factors on the researcher

6.2.2.1. Deviation of microclimate indicators

To ensure a proper microclimate at the user's workplace, the following set of actions must be taken:

- equipping general exchange ventilation to ensure an appropriate microclimate in the premises;

- periodically monitoring the humidity of the air;
- using the air conditioning system in summer;
- providing a heating system in the cold season;
- carrying out timely repair and maintenance of ventilation and air conditioning systems;

In the computer classroom (auditorium No. 417), a temperature of 19-20° C is maintained at a relative humidity of 55-58 percent, which meets the criteria of SanPiN 1.2.3685-21.

6.2.2.2. Insufficient illumination of the working area

Calculations of artificial lighting of the worker's work area will be given further in accordance with the methodological guidelines "Calculation of artificial lighting" [44].

Formula (6.2) determines the calculated height of the placement of lamps above the work surface (h):

$$h = H - h_p - h_c, \quad (6.2)$$

where:

H : room height;

h_p : the distance between the work surface of the table and the floor;

h_c : the distance between the light source and the ceiling.

Formula (6.3) determines the index of the room (i):

$$i = \frac{S}{h(A + B)}, \quad (6.3)$$

where:

S : room area;

h : height of lamps placement;

A : room length;

B : room width.

Formula (6.4) determines the illumination of the room (E_ϕ):

$$E_{\phi} = \frac{n \cdot \eta \cdot \phi}{S \cdot k \cdot z}, \quad (6.4)$$

where:

n: number of lamps;

η: light flux utilization factor;

φ: luminous flux of the lamp;

S: room area;

k: stock factor;

z: the coefficient of uneven lighting.

At the workplace, the height of the room **H** is 2,5 meters, and the height of the work surface h_p is 0,8 meters. The lamps are installed at the level with the ceiling, then we assume that the distance between the light source and the ceiling h_c is zero. Thus, according to the formula (6.2), the height of the suspension of the lamps above the work surface for the workplace is equal to:

$$h = 2,5 - 0,8 - 0 = 1,7 \text{ m}$$

By calculating the height of the suspension of the lamps above the working surface h , we can calculate the index of the room **i**. The room has the following parameters:

$$S = 29.5 \text{ m}^2, A = 6 \text{ m}, B = 4,92 \text{ m}.$$

According to the formula (6.3), the index of the room for the computer class (auditorium No. 417) **i** is equal to:

$$i = \frac{29.5}{1,7 (6 + 4,92)} = 1,58$$

The workplace has two windows without curtains, and the walls are painted white. Light Armstrong ceiling panels are installed on the ceiling. We will assume that the reflection coefficient from the walls ρ_c is 50%, and the reflection coefficient from the ceiling ρ_n is 50%.

For the values **i**, ρ_c , and ρ_n in accordance with the table of luminous flux utilization coefficients from the manual [47], we will assume that the luminous flux

utilization coefficient η is equal to 0.34 (for a ceiling lamp of any type at $i = 1,5$) [47].

According to the table of stock coefficients of luminaires with fluorescent lamps [44], the stock coefficient k for rooms with low dust emission is 1,5. The correction factor z for fluorescent lamps is 1,1.

There are 12 lamps in the workplace, each lamp consists of 4 lamps. The type of lamps is LVO 4×18 CSVT with fluorescent lamps of type L 18W/640 with a flow of $\phi = 1200$ lm.

According to the formula (6.4), the illumination of the room for the computer class (auditorium No. 417) E_{ϕ} is equal to:

$$E_{\phi} = \frac{12 \cdot 4 \cdot 0,34 \cdot 1200}{29,5 \cdot 1,5 \cdot 1,1} = 402 \text{ lux}$$

The illumination of the workplace meets the requirements of SP 52.13330.2016.

6.2.2.3. Exceeding the noise level

The main sources of noise in a computer auditorium are computer components. These elements include the following:

- fans that are located in the power supply, processor, and video card;
- any additional fan in the system unit case.

To reduce the noise level, it is necessary to reduce the temperature inside the system unit, which can be achieved by ensuring proper ventilation. To cool the system unit, it is required to provide at least 20-30 cm of free space from the ventilation openings. In addition, foreign objects that reduce heat transfer should be removed from the equipment, and ventilation openings should be cleaned of dust.

6.2.2.4. Increased magnetic field strength

It is recommended to connect no more than two computers to one outlet, install a reliable grounding and connect the computer to the outlet through an electric field neutralizer.

A liquid crystal monitor should also be used in the workplace to provide less electromagnetic radiation. It is advisable to use two monitors. It is expected that this

will reduce the time spent on developing and testing the software being developed while working with a computer.

Ways to ensure electromagnetic safety:

- PC workstations should not be located near sources of electromagnetic fields (transformers, high-power electricity consumers, switchboards, cable connections, radio transmitting equipment);
- positioning the display screen at least 50 cm away from the user's eyes;
- Not using extension cords (carriers) and surge protectors made in the form of carriers;
- grounding the massive metal elements of the room equipment;
- placing the power wires, if possible, in shielding metal shells or pipes.

6.3. Environmental safety

Software development takes a lot of energy. A software developer cannot avoid using energy. The employees, on the other hand, can guarantee that they, if possible and if available, use renewable energy sources. In general, renewable energy comes from the sun, wind, tides, and geothermal heat, and it is spontaneously renewed.

There is a requirement to dispose of waste generated by personnel during software development. First of all, it is paper garbage, plastic waste, faulty computer parts, and other types of computers.

All waste generated in accordance with their hazard class is transferred to specialized enterprises for processing, disposal, or disposal, as required by law. Landfills or soil should be used to dispose of waste that cannot be recycled or reused.

When the life of the PC expires, it must be disposed of. Recycling is carried out by separating them into homogeneous components that can be reused (for example, silicon, aluminum, and rare metals) [45].

6.4. Safety in emergency situations

6.4.1. Analysis of probable emergencies that may occur during the development and operation of the software

Most of the software development is done in office buildings. The most common occupational risks, such as thunderstorms, hurricanes, landslides, and fire risks, pose the greatest threat to most IT workers. A fire may start in the event of an unexpected electrical malfunction. Fire has the ability to spread quickly. It is recommended to carry out preventive work aimed at reducing risk factors. A fire in an office can lead to significant material damage and even death.

The main causes of fire [46]:

- technical malfunction of the equipment, resulting in the release of flammable vapors or liquids;
- careless handling of an open fire or gas appliance;
- storage of fire-hazardous materials in violation of regulations;
- overload of electrical networks;
- violation of the standards of operation of electrical equipment or its operation in a faulty condition;
- the use of substandard lighting fixtures, electrical wiring, and devices that cause sparking, short circuit, and other problems;
- spontaneous combustion of substances and materials.

6.4.2. Justification of emergency prevention measures and development of an action procedure in case of an emergency

Fire prevention is a set of organizational and technical measures aimed at ensuring the safety of employees. On the other hand, creating several conditions can significantly reduce the likelihood of a fire. Special fire emergency training can increase employee awareness of the problem. Employees in an emergency can use evacuation instructions and plans to safely leave the office building before help arrives. Fire extinguishers should be placed in almost every office to extinguish the flames in the early stages.

Since the premises with office equipment contain a significant number of plastic substances that emit volatile hazardous compounds and acrid smoke during combustion, effective smoke removal is required.

According to the Fire Safety Rules of the Russian Federation НПБ 01-2003 (item 16), evacuation plans for people in case of fire should be developed and posted in prominent places in buildings and structures (except residential buildings) with more than 10 people on the floor at a time. People should be evacuated from the fire zone in accordance with these evacuation plans [48].

Fire detectors are installed in the computer classroom (auditorium No. 417) of the TPU Cybernetic Center, that allow warning the staff on duty in the event of a fire. Smoke photovoltaic detectors are installed in the room as fire detectors. According to the current federal legislation, the head of the facility is responsible for any violations of fire safety rules.

6.5. Conclusion

In this section, the main problems related to workers' rights to work, compliance with labor protection standards, industrial safety, ecology, and resource conservation were discussed. During the implementation of the project, it was found that the researcher's workplace complies with occupational safety and health standards, and the harmful impact of the equipment used on the environment does not exceed the relevant norms.

Each employee must perform professional duties taking into account social, legal, environmental, and cultural factors, and demonstrate competence in occupational safety issues as a combination of knowledge and skills acquired through preparation, professional experience, and training. In particular, to avoid production risks, assess risks that cannot be eliminated; deal with sources of risks; take into account the technical condition of the organization; replace what is dangerous with what is not dangerous or less dangerous; and take individual and collective measures for safe working methods.

Conclusion

The design of a multi-stage program for modeling cloud computing infrastructure in software-defined networks has been described in detail. The design follows the object-oriented design patterns and principles to provide an extensible and modular tool, which can be for further development. Using the developed tool, it is possible to simulate a cloud infrastructure in SDN networks, where the SDN controller is responsible for network management operations. It controls multiple network elements (switches, routers, etc.), by managing their forwarding tables. The implemented SDN controller is able to parse the network topology to build the routing tables for all the nodes, discovering the shortest path with the minimum latency between different nodes. After building the network information model, the SDN controller is ready to deploy the SDN applications policy and requirements on the underlying network. The policy is provided by the client or managed by the ISP and its purpose is to control traffic flows. The designed controller is able to deploy a bandwidth policy, where the bandwidth requirements between source and destination nodes are specified.

The architecture and the design of the web interface for simulation experiments in the modified CloudSim Plus platform have been introduced. The design consists of multiple microservice components that communicate using synchronous and asynchronous APIs. The web interface allows the creation and managing of different modeling entities of the cloud that can be used to build experiment scenarios. Also, it provides an interface to display the simulation results using graphical charts.

List of student publications

1. Mouhamad I. Layout of state-ordered e-mail // Modern Technologies, Economics and Education: Proceedings of the II All-Russian Scientific and Methodological Conference. – Tomsk, 2020. – P. 42-43.
2. Mouhamad I. Using Modeling to Understand Bitcoin and Blockchain // Youth and Modern Information Technologies: Proceedings of XVIII International Scientific-Practical Conference of Students, Graduate Students and Young Scientists. – Tomsk, 2021. – P. 366-367.
3. Mouhamad I. Study of simulators for cloud computing // Eurasian scientific journal. NO.3, 2022 P. 14-19.
4. Mouhamad I. Comparison of auto scaling methods using different VM allocation algorithms in cloud computing // II International Scientific and Technical Conference "Actual problems of science and technology". – Sarapul, 2022 (in press).
5. Mouhamad I. Auto Cloud Simulator (AutoCS): A web-based framework for simulating cloud computing infrastructure and networking // IX International youth scientific conference "Mathematical and software of information, technological and economic systems". – Tomsk, 2022 (in press).

References

1. vSphere // VMware, Inc. [2022]. URL: <https://www.vmware.com/products/vsphere.html> (date of access: 28.04.2022).
2. OpenNebula 6.2 // OpenNebula Systems. [2002-2021]. URL: <https://opennebula.io/> (date of access: 28.04.2022).
3. Eucalyptus Cloud Platform // GitHub, Inc. [2022]. URL: <https://github.com/eucalyptus/eucalyptus> (date of access: 28.04.2022).
4. VMware ESXi: The Purpose-Built Bare Metal Hypervisor // VMware, Inc. [2022]. URL: <https://www.vmware.com/products/esxi-and-esx.html> (date of access: 28.04.2022).
5. The Xen project brings the power of virtualisation everywhere // The Linux Foundation. URL: <https://xenproject.org/> (date of access: 28.04.2022).
6. Kernel Virtual Machine // Red Hat, Inc. [2022]. URL: https://www.linux-kvm.org/page/Main_Page (date of access: 28.04.2022).
7. Hyper-V technology overview // Microsoft. [2022]. URL: <https://docs.microsoft.com/en-us/windows-server/virtualization/hyper-v/hyper-v-technology-overview> (date of access: 28.04.2022).
8. Use Amazon EC2, S3, and more - free for a full year // Amazon Web Services, Inc. [2022]. URL: <https://aws.amazon.com/> (date of access: 28.04.2022).
9. Keep your source code options open // Microsoft. [2022]. URL: <https://azure.microsoft.com/en-us/> (date of access: 28.04.2022).
10. SDN architecture // Tr, O. N. F. [2016]. URL: https://opennetworking.org/wp-content/uploads/2013/02/TR_SDN_ARCH_1.0_06062014.pdf (date of access: 28.04.2022).
11. Kreutz D., Ramos F. M., Verissimo P. E., Rothenberg C. E., Azodolmolky S., Uhlig S. Software-defined networking: A comprehensive survey // Proceedings of the IEEE. – 2014. – 103(1). – P. 14-76.
12. Nadeau T. D., Gray K. SDN: Software Defined Networks // O'Reilly Media, Inc. – 2013. – 352 p. – ISBN 978-1-449-34230-2.

13. Hwang R. H., Tseng H. P., Tang Y. C. Design of SDN-Enabled cloud data center // IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity). – 2015. – P. 950-957.
14. Todorov M. Analysis of Software-Defined Wide-Area Networking // Aalborg University Copenhagen. [2018]. URL: https://projekter.aau.dk/projekter/files/281292955/Masters_thesis_Mario_Todorov.pdf (date of access: 28.04.2022).
15. MPLS Applications User Guide // Juniper Networks. [2018]. URL: <https://www.juniper.net/documentation/us/en/software/junos/mpls/> (date of access: 28.04.2022).
16. Powersim Studio modelling tools // Powersim Software AS. [2021]. URL: <http://powersim.com/modelling-tools/> (date of access: 28.04.2022).
17. ExtendSim // Imagine That Inc. [1987-2022]. URL: <https://extendsim.com/> (date of access: 28.04.2022).
18. Experience the Future // INCONTROL. URL: <https://www.incontrolsim.com/> (date of access: 28.04.2022).
19. AnyLogic // The AnyLogic Company. URL: <https://www.anylogic.ru/> (date of access: 28.04.2022).
20. AnyLogic Cloud – a web service for operational use of models // The AnyLogic Company. URL: <https://www.anylogic.com/> (date of access: 28.04.2022).
21. CloudSim: a framework for modeling and simulation of cloud computing infrastructures and services // cloudbus.org. URL: <http://www.cloudbus.org/cloudsim/> (date of access: 28.04.2022).
22. Graduate Study Expo // The University of Melbourne. URL: <https://www.unimelb.edu.au/> (date of access: 28.04.2022).
23. CloudSim Plus // cloudsimplus.org. URL: <https://cloudsimplus.org/> (date of access: 28.04.2022).
24. manoelcampos / cloudsimplus // GitHub, Inc. [2022]. URL: <https://github.com/manoelcampos/cloudsimplus> (date of access: 28.04.2022).

25. Cloudslab / CloudSimEx // GitHub, Inc. [2022]. URL: <https://github.com/Cloudslab/CloudSimEx> (date of access: 28.04.2022).
26. CloudSim and CloudSimEx [Part 3] - Delayed VM and Cloudlet actions // nikolaygrozev.wordpress.com. URL: <https://nikolaygrozev.wordpress.com/2014/09/13/cloudsim-and-cloudsimex-part-3-delayed-vm-and-cloudlet-actions/> (date of access: 28.04.2022).
27. Cloudslab / cloudsimsdn // GitHub, Inc. [2022]. URL: <https://github.com/Cloudslab/cloudsimsdn> (date of access: 28.04.2022).
28. Alshammari D. Evaluation of cloud computing modelling tools: simulators and predictive models. PhD diss. / Alshammari Dhahi; University of Glasgow. – Glasgow, 2018. – 190 p.
29. Sonmez C., Ozgovde A., Ersoy C. Edgecloudsim: An environment for performance evaluation of edge computing systems // Transactions on Emerging Telecommunications Technologies. – 2018. – 29(11). – e3493.
30. Teixeira Sá T., Calheiros R. N., Gomes D. G. CloudReports: An extensible simulation tool for energy-aware cloud computing environments // In cloud computing Springer, Cham. – 2014. – P. 127-142.
31. Wickremasinghe B., Calheiros R. N., & Buyya R. Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications // 24th IEEE international conference on advanced information networking and applications. – 2010. – P. 446-452.
32. Zhihua, J. Greencloud for simulating qos-based naas in cloud computing // Ninth International Conference on Computational Intelligence and Security. – 2013. – P. 766-770.
33. Maarouf A., Marzouk A., & Haqiq A. Comparative study of simulators for cloud computing // International Conference on Cloud Technologies and Applications (CloudTech). – 2015. – P. 1-8.
34. Ahmed A., Sabyasachi A. S. Cloud computing simulators: A detailed survey and future direction // IEEE international advance computing conference (IACC). – 2014. – P. 866-872.

35. Bashar A. Modeling and simulation frameworks for cloud computing environment: A critical evaluation // International Conference on Cloud Computing and Services Science. – 2014. – P. 1-6.
36. Byrne J., Svorobej S., Giannoutakis K.M., Tzovaras D., Byrne P.J., Östberg P.O., Gourinovitch A. Lynn T. A review of cloud computing simulation platforms and related environments // International Conference on Cloud Computing and Services Science. – 2017. – Vol. 2. – 679-691.
37. Garg S. K., & Buyya R. Networkcloudsim: Modelling parallel applications in cloud simulations // *Fourth IEEE International Conference on Utility and Cloud Computing*. – 2011. – 105-113.
38. Medina A., Lakhina A., Matta I., Byers J. An approach to universal topology generation // In Proc. IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS). – 2001.
39. Weisstein W. Floyd-warshall algorithm. // MathWorld--A Wolfram Web Resource. [2022]. URL: <https://mathworld.wolfram.com/Floyd-WarshallAlgorithm.html> (date of access: 28.04.2022).
40. Cloud Providers' Investments in Edge Computing, AI, and 5G to Magnify Global Data Center Market by 2025 // Frost & Sullivan [2021]. URL: <https://www.frost.com/news/press-releases/cloud-providers-investments-in-edge-computing-ai-and-5g-to-magnify-global-data-center-market-by-2025/> (date of access: 10.05.2022).
41. SanPiN 1.2.3685-21 «Hygienic standards and requirements for ensuring the safety and (or) harmlessness of environmental factors for humans». – URL: https://biotorg.com/upload/medialibrary/039/SanPiN-1.2.3685_21.pdf (date of access: 10.05.2022).
42. GOST 12.1.003-83 «Occupational safety standards system. Noise. General safety requirements» // URL: <https://docs.cntd.ru/document/5200291> (date of access: 10.05.2022).
43. GOST 12.1.002-84 «Electric fields of industrial frequency» // URL: <https://docs.cntd.ru/document/5200271> (date of access: 10.05.2022).

44. Life safety. Calculation of artificial lighting. Methodical instructions for individual assignments for full-time and part-time students of all directions and specialties of TPU. – Tomsk: Publishing house of Tomsk Polytechnic University. 2008. – 20 p. (In Russ.)
45. Valuev D.V., Gizatulina R.A. Technologies of metallurgical wastes processing: manual. – Tomsk: Tomsk Polytechnic University Press, 2012. – 196 p. (In Russ.)
46. Laboratory practical work on the discipline of "Safety of Life" for students of all specialties: teaching aid. V. Anischenko, A.N. Vtorushina, M.V. Gulyaev, M.E. Guselnikov, A.G. Dashkovsky, T.A. Zadorozhnaya, V.N. Izvekov, A.G. Kagirov, K.M. Kostyrev, V.F. Panin, A.M. Plakhov, S.V. Romanenko. – Tomsk: Tomsk Polytechnic University Publisher, 2010. – 236 p. (In Russ.)
47. Reference to MGSN 2.06-99 Calculation and design of artificial lighting of the premises of public buildings // URL: <https://docs.cntd.ru/document/1200005932> (date of access: 10.05.2022). (In Russ.)
48. PPB 01-03 Rules of fire safety in the Russian Federation // URL: <https://files.stroyinf.ru/Data1/11/11702/index.htm> (date of access: 10.05.2022). (In Russ.)
49. SP 52.13330.2016 «Natural and artificial lighting. Updated version of SNiP 23-05-95» // JSC "Codex". [2022]. URL: (date of access: 10.05.2022). (In Russ.)

Appendix A

1. Frontend image Dockerfile

```
1. # Copyright (C) 2022 Ibrahim Mouhamad
2. #
3. # SPDX-License-Identifier: MIT
4.
5. FROM node:lts-slim AS autocs-ui
6.
7. ENV TERM=xterm \
8.     LANG='C.UTF-8' \
9.     LC_ALL='C.UTF-8'
10.
11. # Install dependencies
12. COPY package.json /tmp/
13. COPY ui/package.json /tmp/ui/
14. COPY yarn.lock /tmp/
15.
16. # Install common dependencies
17. WORKDIR /tmp/
18. RUN yarn install --ignore-scripts
19.
20. # Build source code
21. COPY ui/ /tmp/ui/
22. RUN yarn run build:ui
23.
24. FROM nginx:mainline-alpine
25. # Replace default.conf configuration to remove unnecessary rules
26. RUN sed -i "s/}/application/wasm wasm;\n}/g" /etc/nginx/mime.types
27. COPY ui/react_nginx.conf /etc/nginx/conf.d/default.conf
28. COPY --from=autocs-ui /tmp/ui/build /usr/share/nginx/html/
```

2. Engine image Dockerfile

```
1. # Copyright (C) 2022 Ibrahim Mouhamad
2. #
3. # SPDX-License-Identifier: MIT
4.
5. FROM maven:3.8.5-openjdk-17 AS build-image
6.
7. COPY core /tmp/core/
8. WORKDIR /tmp/core
9. RUN mvn clean install
10.
11. COPY engine /tmp/engine/
12. WORKDIR /tmp/engine
13. RUN mvn clean install -Pprod
14.
15. FROM openjdk:17-slim-buster AS autocs_engine
16. COPY --from=build-image /tmp/engine/target/engine-0.0.1.jar app.jar
17. COPY data /tmp/data/
18.
19. RUN sh -c 'touch /app.jar'
20.
21. ENTRYPOINT ["java", "-Djava.security.egd=file:/dev/./urandom", "-jar", "/app.jar"]
```

3. Backend image Dockerfile

```
1. # Copyright (C) 2022 Ibrahim Mouhamad
2. #
3. # SPDX-License-Identifier: MIT
4.
5. FROM maven:3.8.5-openjdk-17 AS build-image
6.
7. COPY core /tmp/core/
8. WORKDIR /tmp/core
9. RUN mvn clean install
10.
11. COPY backend /tmp/backend/
12. WORKDIR /tmp/backend
13. RUN mvn clean install -Pprod
14.
15. FROM openjdk:17-slim-buster AS autocs_server
16. COPY --from=build-image /tmp/backend/target/backend-0.0.1.jar app.jar
17. COPY data /tmp/data/
18.
19. RUN sh -c 'touch /app.jar'
20.
21. ENTRYPOINT ["java", "-Djava.security.egd=file:/dev/./urandom", "-
jar", "/app.jar"]
```

4. Docker compose file to run all system microservices

```
1. version: '3'
2. services:
3.   autocs:
4.     container_name: autocs
5.     image: autocs/autocs_server
6.     restart: always
7.     depends_on:
8.       - autocs_engine
9.       - rabbitmq
10.    labels:
11.      - traefik.enable=true
12.      - traefik.http.services.autocs.loadbalancer.server.port=8080
13.      - traefik.http.routers.autocs.rule=Host(`${AUTOCS_HOST:-
localhost}`) && PathPrefix(`/api/`)
14.      - traefik.http.routers.autocs.entrypoints=web
15.    volumes:
16.      - ./data:/tmp/data
17.      - autocs_workspace:/tmp/workspace
18.      - autocs_logs:/tmp/logs
19.    networks:
20.      - autocs
21.    logging:
22.      driver: "json-file"
23.      options:
24.        max-size: "100m"
25.        max-file: "5"
26.
27.   autocs_engine:
28.     container_name: autocs_engine
29.     image: autocs/autocs_engine
30.     restart: always
31.     depends_on:
32.       - rabbitmq
```

```

33.     volumes:
34.         - ./data:/tmp/data
35.         - autocs_workspace:/tmp/workspace
36.         - autocs_logs:/tmp/logs
37.     networks:
38.         - autocs
39.     logging:
40.         driver: "json-file"
41.         options:
42.             max-size: "100m"
43.             max-file: "5"
44.
45.     autocs_ui:
46.         container_name: autocs_ui
47.         image: autocs/autocs_ui
48.         restart: always
49.         depends_on:
50.             - autocs
51.         labels:
52.             - traefik.enable=true
53.             - traefik.http.services.autocs-
ui.loadbalancer.server.port=80
54.             - traefik.http.routers.autocs-ui.rule=Host(`${AUTOCS_HOST:-
localhost}`)
55.             - traefik.http.routers.autocs-ui.entrypoints=web
56.         networks:
57.             - autocs
58.
59.     traefik:
60.         image: traefik:v2.4
61.         container_name: traefik
62.         restart: always
63.         command:
64.             - "--providers.docker.exposedByDefault=false"
65.             - "--providers.docker.network=autocs"
66.             - "--entryPoints.web.address=:8080"
67.         # Uncomment to get Traefik dashboard
68.         # - "--entryPoints.dashboard.address=:8090"
69.         # - "--api.dashboard=true"
70.         # labels:
71.         # - traefik.enable=true
72.         # - traefik.http.routers.dashboard.entrypoints=dashboard
73.         # - traefik.http.routers.dashboard.service=api@internal
74.         # - traefik.http.routers.dashboard.rule=Host(`${AUTOCS_HOST:-
localhost}`)
75.         ports:
76.             - 8080:8080
77.             - 8090:8090
78.         volumes:
79.             - /var/run/docker.sock:/var/run/docker.sock:ro
80.         networks:
81.             - autocs
82.
83.     rabbitmq:
84.         container_name: rabbitmq
85.         image: rabbitmq:management
86.         ports:
87.             - "5672:5672"
88.             - "15672:15672"
89.         networks:
90.             - autocs
91.         logging:
92.             driver: "json-file"

```



```
93.             options:
94.                 max-size: "100m"
95.                 max-file: "5"
96.
97. volumes:
98.     autocs_workspace:
99.     autocs_logs:
100.
101.
102. networks:
103.     autocs:
104.
```