Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерних наук_____
(повна назва)
Кафедра_____Інформаційних управляючих систем_____
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____другий (магістерський)_____
_____Дослідження моделей черг у хмарних обчисленнях для ІС ІТ-компаній_
(тема)

Виконав:
студент _2_ курсу, групи _ІУСТм-22-1____
_____Обушко Данило Романович_____
(прізвище, ім'я, по батькові)
Спеціальність ___122 Комп'ютерні_науки__
(код і повна назва спеціальності)
Тип програми __освітньо-професійна____
(освітньо-професійна або освітньо-наукова)
Освітня програма Інформаційні управляючі системи та технології_____
(повна назва освітньої програми)
Керівник ____Олена МІХНОВА_____
(Власне ім'я ПРІЗВИЩЕ)

Допускається до захисту
Зав. кафедри  Костянтин ПЕТРОВ_
(Власне ім'я ПРІЗВИЩЕ)

2024 р

Харківський національний університет радіоелектроніки

Факультет_____Комп'ютерних наук_____

Кафедра_____Інформаційних управляючих систем_____

Рівень вищої освіти_____другий (магістерський)_____

Спеціальність_____122 Комп'ютерні науки_____
(код і повна назва)

Тип програми _____освітньо-професійна_____
(освітньо-професійна або освітньо-наукова)

Освітня програма___Інформаційні управляючі системи та технології_____
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри

«_20__»__листопада__ 2023_ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові_____Обушко Данилу Романовичу _____
(прізвище, ім'я, по батькові)

1. Тема роботи _Дослідження моделей черг у хмарних обчисленнях для ІС ІТ-компаній_ затверджена наказом університету від _16_ __листопада__2023_ р. № _1359Ст_
2. Термін подання студентом роботи до екзаменаційної комісії 15 січня 2024 р.
3. Вихідні дані до роботи Матеріали звіту з науково-дослідної практики, опис об'єкта досліджень – ІС ІТ-компаній, науково-технічні публікації та інтернет-джерела з тематики кваліфікаційної роботи
4. Перелік питань, що потрібно опрацювати в роботі формування проблеми щодо модифікації моделі черг для імплементації в ІС ІТ-компаній, які працюють на базі хмарних обчислень; огляд та загальний аналіз об'єкта та методів дослідження; вибір цілі та напрямків удосконалення; постановка завдань до вирішення даної проблеми; розробка модифікованої моделі систем масового обслуговування для хмарних обчислень ІТ-компаній; апробація та оцінка модифікованої моделі черг на тестових даних_____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз схеми організаційної структури, літератури та джерел	20.11.2023	Виконано
2	Опис постановки задачі дослідження	20.11.2023-22.11.2023	Виконано
3	Обробка матеріалів передатестаційної практики	22.11.2023-24.11.2023	Виконано
4	Аналіз існуючих моделей систем масового обслуговування	24.11.2023-26.11.2023	Виконано
5	Побудова Q-схем систем масового обслуговування	26.11.2023-28.11.2023	Виконано
6	Розробка модифікацій моделі систем масового обслуговування	28.11.2023-30.11.2023	Виконано
7	Модифікація моделі систем масового обслуговування	30.11.2023-01.12.2023	Виконано
8	Апробація модифікованої моделі системи масового обслуговування	01.12.2023-02.12.2023	Виконано
9	Підготовка та оформлення пояснювальної записки та графічного матеріалу до кваліфікаційної роботи	02.12.2023-05.12.2023	Виконано
10	Підготовка презентаційних матеріалів	05.12.2023-09.12.2023	Виконано
11	Подання студентом роботи для перевірки на плагіат	10.01.2024	Виконано
12	Подання роботи на підпис науковому керівнику	11.01.2024	Виконано
13	Попередній захист роботи	12.01.2024	Виконано
14	Надання роботи на рецензію	13.01.2024	Виконано
15	Надання роботи на підпис завідувачу кафедри	15.01.2024	Виконано
16	Захист кваліфікаційної роботи	17.01.2024	Виконано

Дата видачі завдання __20___ ____листопада___2023 р.

Студент _____
(підпис)

Керівник роботи _____ _доц. каф. ІУС Олена Міхнова
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи містить: _76_ с., _24_ рис., _1 табл., _25_ джерел, 1_ додаток.

БАЗА ДАНИХ, ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ, СИСТЕМИ МАСОВОГО ОБСЛУГОВУВАННЯ, ТЕОРІЯ МАСОВОГО ОБСЛУГОВУВАННЯ, ХМАРНІ ТЕХНОЛОГІЇ.

Об’єкт дослідження – процеси розміщення, впорядкування та обробки заявок у черзі ІС ІТ-компаній, працюючих з хмарними ресурсами в якості IaaS.

Предметом дослідження є моделі черг та їх використання при проектуванні або модернізації інформаційних систем ІТ-компаній задля підвищення ефективності функціонування останніх.

Мета роботи – вдосконалення моделі системи масового обслуговування для вирішення задач оптимального використання хмарних ресурсів ІТ-компаній. Створення моделі черг, що забезпечує високий рівень стабільності та надійності ІС.

Методи дослідження – системний аналіз, методи і засоби побудови схем систем масового обслуговування, методи і засоби Q-схем, методи і засоби моделювання структури програмного забезпечення.

В кваліфікаційній роботі виконано аналіз існуючих моделей систем масового обслуговування, структурний аналіз потоку черги, побудовано схеми алгоритму роботи таких систем, в тому числі модифікованої.

ABSTRACT

The explanatory note to the qualification paper contains: 76 p., _24_ fig., _1 table, _25_ sources, 1_ appendix.

DATA BASE, INFORMATION TECHNOLOGIES, MASS SERVICE SYSTEMS, MASS SERVICE THEORY, CLOUD TECHNOLOGIES.

The object of the study is the processes of placing, arranging and processing applications in the IT queue of IT companies working with cloud resources as IaaS.

The subject of research is queuing models and their use in the design or modernization of information systems of IT companies in order to increase the efficiency of the latter's functioning.

The purpose of the work is to improve the model of the mass service system for solving the problems of optimal use of cloud resources of IT companies. Creating a queue model that ensures a high level of IS stability and reliability.

Research methods - system analysis, methods and means of building schemes of mass service systems, methods and means of Q-schemes, methods and means of modeling the structure of software.

In the qualification work, an analysis of existing models of mass service systems, a structural analysis of the queue flow, and a diagram of the algorithm of the operation of such systems, including a modified one, were performed.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

БД – база даних

ІС – інформаційна система

ІТ – інформаційні технології

ОС – операційна система

ПЗ – програмне забезпечення

ПК – персональний комп'ютер

СМО – система масового обслуговування

IaaS - Infrastructure as a Service (інфраструктура як послуга)

PaaS – Platform as a Service (платформа як послуга)

SaaS – Software as a Service (програмне забезпечення як послуга)

ВСТУП

Хмарні обчислення є швидко зростаючою галуззю, яка пропонує IT-компаніям широкий спектр можливостей для оптимізації своїх операцій. Однією з ключових проблем, з якою стикаються IT-компанії при використанні хмарних обчислень, є ефективне управління чергами. Черги в хмарних обчисленнях виникають, коли множина запитів конкурує за доступ до обмежених ресурсів. Наприклад, коли користувачі намагаються отримати доступ до веб-сайту, їхні запити можуть бути поставлені в чергу, якщо веб-сервер не може обробити їх одразу.

Швидке зростання популярності хмарних обчислень створює новий виклик для IT-компаній. У міру того, як все більше і більше користувачів і пристроїв переходять у хмару, навантаження на IT-системи зростає. Ефективне управління чергами є ключовим фактором для забезпечення того, щоб IT-системи могли підтримувати це зростаюче навантаження. Ефективне управління чергами може допомогти IT-компаніям покращити ефективність своїх IT-систем, зменшивши затримки і втрати даних.

Необхідність розробки нових модифікацій моделей черг у хмарних обчисленнях також є актуальною. Існує багато різних моделей черг, і кожна модель має свої переваги та недоліки. Крім того, IT-компанії повинні мати можливість оптимізувати роботу моделей черг, щоб покращити їхню ефективність.

У результаті даного дослідження будуть розроблені модифікації для моделі системи масового обслуговування(СМО). Це може бути використано IT-компаніями для підвищення ефективності та продуктивності своїх IT-систем.

1 АНАЛІЗ ІСНУЮЧИХ МОДЕЛЕЙ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Хмарні обчислення в IT-компаніях

Хмарні обчислення – це парадигма надання IT-ресурсів, таких як обчислювальна потужність, зберігання даних, бази даних, мережеві послуги, програмне забезпечення, аналіз даних, штучний інтелект та машинне навчання, через Інтернет [1, 2]. Хмарні обчислення дозволяють компаніям отримувати доступ до цих ресурсів за потреби, без необхідності їх закупівлі та обслуговування [3].

Основні характеристики хмарних обчислень:

– підтримка з боку постачальника. Хмарні сервіси надаються постачальником, який відповідає за їхнє обслуговування та підтримку;

– платіж за використання. Хмарні сервіси оплачуються за фактичним використанням;

– гнучкість. Хмарні сервіси можна масштабувати відповідно до потреб компанії;

– доступність. Хмарні сервіси доступні через Інтернет, що забезпечує їхню глобальну доступність [4].

Впливом хмарних обчислень на сучасні IT-компанії є значним. Запровадження хмарних обчислень в сучасній IT-індустрії стало ключовою стратегією для багатьох компаній, що привело до значних змін у їхньому функціонуванні та виробничих процесах. Однією з ключових переваг є підвищена ефективність використання обчислювальних ресурсів. Завдяки можливості еластичного масштабування, компанії можуть легко адаптувати свою обчислювальну інфраструктуру до змінних потреб, що сприяє оптимізації витрат та підвищенню продуктивності.

Хмарні обчислення мають суттєвий вплив на діяльність сучасних IT-компаній, надаючи їм можливість знижувати витрати, збільшувати гнучкість та

покращувати доступність [5]. Однією з ключових переваг використання хмарних сервісів є їхня економічна вигода порівняно із традиційними IT-інфраструктурами. Компанії можуть оптимізувати свої витрати, використовуючи хмарні обчислення, що дозволяє ефективніше розподіляти ресурси та забезпечує фінансову стійкість. Додатково, хмарні сервіси забезпечують гнучкість у використанні обчислювальних ресурсів. Компанії можуть легко масштабувати свою IT-інфраструктуру відповідно до змін потреб, що є важливим фактором у змінливому інформаційному середовищі. Такий підхід дозволяє підприємствам швидко реагувати на змінні умови ринку та вдосконалювати свою конкурентоспроможність. Однією з істотних переваг є також покращена доступність до обчислювальних ресурсів через глобальну мережу. Це дозволяє компаніям обслуговувати клієнтів у всьому світі, забезпечуючи ефективність та якість обслуговування. Усі ці аспекти роблять хмарні обчислення ключовою парадигмою в IT-сфері, що впливає на стратегії та функціонування IT-компаній у сучасному цифровому віці. Поза основними характеристиками, хмарні обчислення поділяються на різні типи, такі як Інфраструктура як послуга (IaaS) та Платформа як послуга (PaaS), що доповнюють їхню універсальність та застосовність в різних областях IT [6].

Програмне забезпечення як послуга (SaaS): Надає доступ до готових до використання програмних продуктів, таких як електронна пошта, CRM-системи та системи управління контентом [7].

Хмарні обчислення широко застосовуються в різних галузях для вирішення різноманітних завдань та оптимізації бізнес-процесів. У фінансовому секторі, вони використовуються для ефективного зберігання фінансових даних, обробки платежів та надання різноманітних фінансових послуг. Роздрібна торгівля використовує хмарні обчислення для зберігання інформації про клієнтів, ефективного управління складами та підтримки електронної комерції. У галузі медицини хмарні обчислення забезпечують надійне зберігання медичних даних, допомагають у діагностиці захворювань та підтримують надання різноманітних

медичних послуг. В освітній сфері вони використовуються для зберігання навчального контенту, надання онлайн-курсів та управління різними освітніми установами. Ці та інші галузі використання хмарних обчислень свідчать про їхню універсальність та значущість у сучасному бізнес-середовищі, де вони допомагають підприємствам ефективно впроваджувати інновації та покращувати свою продуктивність.

Хмарні платформи можна поділити на три основні типи:

– інфраструктура як послуга (IaaS) - це найнижчий рівень хмарних послуг, який надає підприємствам доступ до віртуальних машин, сховищ даних, мереж і інших базових обчислювальних ресурсів [7];

– платформа як послуга (PaaS) - це середній рівень хмарних послуг, який надає підприємствам готові платформи для розробки, розгортання та управління веб-додатками та мобільними додатками [7];

– програмне забезпечення як послуга (SaaS) - це найвищий рівень хмарних послуг, який надає підприємствам готові програмні продукти, які можна використовувати через Інтернет [7].

Хмарні платформи IaaS пропонують підприємствам доступ до фізичних ресурсів, таких як віртуальні машини, сховища даних, мережі та інші. Клієнти IaaS відповідають за розгортання, управління та обслуговування своїх додатків і систем на цих ресурсах. Фізична інфраструктура. Хмарні платформи IaaS розміщуються на фізичній інфраструктурі постачальника хмарних послуг. Ця інфраструктура може бути розташована в одному або декількох дата-центрах. Хмарні платформи IaaS використовують віртуалізацію для створення віртуальних машин, які імітують фізичні сервери. Віртуальні машини можуть бути конфігуровані відповідно до потреб клієнта. Хмарні платформи IaaS дозволяють масштабувати ресурси вгору або вниз відповідно до потреб клієнта. Це забезпечує високу гнучкість і ефективність використання ресурсів [8].

Хмарні платформи PaaS пропонують підприємствам готові платформи для розробки, розгортання та управління веб-додатками та мобільними додатками. Клієнти PaaS можуть використовувати ці платформи для створення та розгортання своїх додатків без необхідності купувати та обслуговувати власне програмне забезпечення та апаратне забезпечення. Хмарні платформи PaaS включають платформне програмне забезпечення, яке забезпечує основні функції для розробки, розгортання та управління веб-додатками та мобільними додатками. Хмарні платформи PaaS використовують віртуалізацію для створення віртуальних машин, які імітують фізичні сервери. Віртуальні машини можуть бути конфігуровані відповідно до потреб клієнта. Хмарні платформи PaaS дозволяють масштабувати ресурси вгору або вниз відповідно до потреб клієнта. Це забезпечує високу гнучкість і ефективність використання ресурсів [9].

Хмарні платформи SaaS пропонують підприємствам готові програмні продукти, які можна використовувати через Інтернет. Клієнти SaaS не мають контролю над програмним забезпеченням, але можуть використовувати його без необхідності купувати та встановлювати його на власному апаратному забезпеченні. Особливостями побудови хмарних платформ SaaS є те, що хмарні платформи SaaS включають готове програмне забезпечення, яке можна використовувати через Інтернет. Хмарні платформи SaaS можуть використовувати віртуалізацію для створення віртуальних машин, які імітують фізичні сервери. Віртуальні машини можуть бути конфігуровані відповідно до потреб клієнта. Хмарні платформи SaaS дозволяють масштабувати ресурси вгору або вниз відповідно до потреб клієнта. Це забезпечує високу гнучкість і ефективність використання ресурсів [10].

Хмарні обчислення продовжують розвиватися, і їхній вплив на сучасні IT-компанії буде тільки зростати. Хмарні обчислення мають ряд переваг для IT-компаній, включаючи:

– зниження витрат. Хмарні сервіси часто є більш економічно вигідними, ніж традиційні IT-інфраструктури. Це пов'язано з тим, що компанії платять лише за фактичне використання, а не за закупівлю та обслуговування обладнання та програмного забезпечення;

– збільшення гнучкості. Хмарні сервіси дозволяють компаніям легко масштабувати свою IT-інфраструктуру відповідно до потреб. Це може бути особливо корисним для компаній, які мають сезонні або непередбачувані потреби в обчисленнях;

– покращення доступності. Хмарні сервіси забезпечують глобальну доступність, що дозволяє компаніям обслуговувати клієнтів у всьому світі.

Однак хмарне обчислення також має деякі обмеження, які слід враховувати, перш ніж приймати рішення про його використання. До них відносяться:

– залежність від постачальника. Компанії, які використовують хмарні сервіси, залежать від постачальника для обслуговування та підтримки своїх ресурсів. Це може бути проблемою, якщо постачальник не зможе забезпечити очікуваний рівень сервісу;

– безпека. Хмарні сервіси можуть бути більш вразливими до атак, ніж традиційні IT-інфраструктури. Це пов'язано з тим, що хмарне середовище є більш складним і розподіленим;

– контроль. Компанії, які використовують хмарні сервіси, мають менший контроль над своїми IT-ресурсами, ніж при використанні традиційних IT-інфраструктур. Це може бути проблемою для компаній, яким потрібний високий рівень контролю над своїми даними та системами.

Хмарні обчислення мають значний потенціал для IT-компаній. Вони можуть допомогти компаніям знизити витрати, збільшити гнучкість та покращити доступність. Однак важливо враховувати обмеження хмарного обчислення, перш ніж приймати рішення про його використання .

Окрім переваг та обмежень, хмарне обчислення також має ряд інших характеристик, які слід враховувати. До них відносяться:

– глобальність. Хмарні сервіси доступні в глобальному масштабі, що дозволяє компаніям обслуговувати клієнтів у всьому світі [11];

– швидкість. Хмарні сервіси часто забезпечують більшу швидкість, ніж традиційні IT-інфраструктури. Це пов'язано з тим, що хмарне середовище є більш масштабованим і ефективним [11];

– простота. Хмарні сервіси часто є більш простими в управлінні, ніж традиційні IT-інфраструктури. Це пов'язано з тим, що постачальники хмарних сервісів часто надають інструменти та послуги для управління хмарної інфраструктури [11].

На рисунку 1.1 зображений відсоток поділу у популярності використання

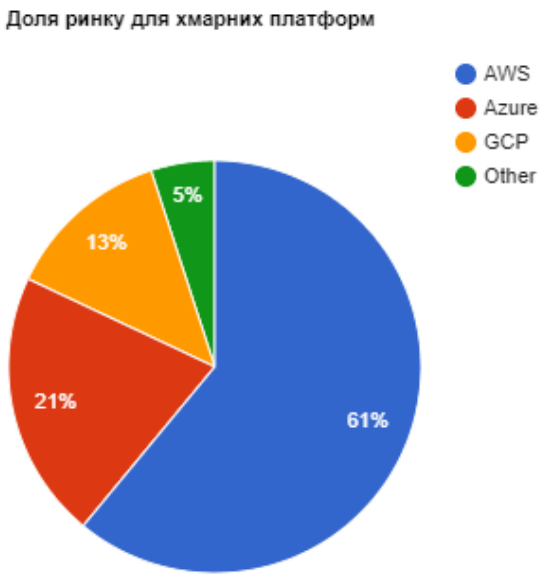

Рисунок 1.1 - Доля ринку для хмарних платформ

найпопулярніших хмарних платформ. З нього можна побачити що AWS є безперечним лідером на ринку хмарних платформ, з часткою у 61%, це означає, що AWS використовується більш ніж двома третинами всіх підприємств, які використовують хмари. AWS пропонує широкий спектр послуг, включаючи обчислювальні ресурси, зберігання даних, мережеві послуги та послуги машинного навчання [12].

Проте більш точну інформацію має діаграма зображена на рисунку 1.2. Порівнюючи їх можна зробити такі висновки. Зіставлення інформації про частку ринку та кількість підприємств, які використовують кожну платформу, дає кілька цікавих висновків [12].

По-перше, можна зробити висновок, що частка ринку не завжди є точним відображенням кількості підприємств, які використовують платформу. Наприклад, AWS має частку ринку 61%, але її використовують лише 61% підприємств, які використовують хмари. Це означає, що інші 39% підприємств, які використовують хмари, використовують платформи Azure, GCP або інші менші платформи.

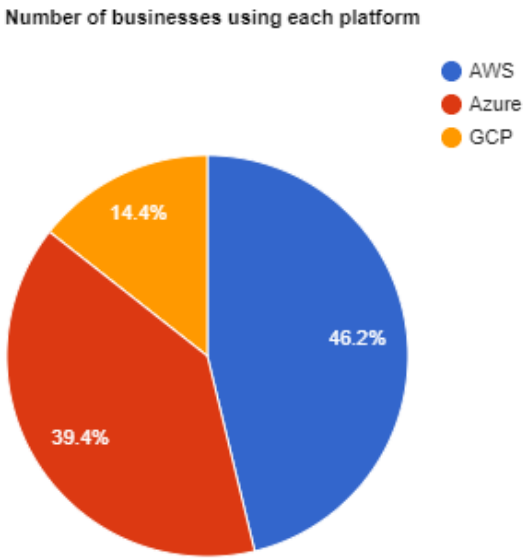

Рисунок 1.2 – Відсоток підприємств, які використовують кожну з платформ

По-друге, можна зробити висновок, що Azure набирає популярність швидше, ніж GCP. У 2022 році частка ринку Azure становила 19%, а кількість підприємств, які використовували Azure, становила 52%. Це означає, що Azure має більш високу частку ринку серед підприємств, які використовують хмари, ніж серед усіх підприємств. По-третє, можна зробити висновок, що багато підприємств використовують мульти-хмарні стратегії. Це означає, що вони використовують більше однієї хмарної платформи. Це може бути пов’язано з тим, що кожна платформа має свої сильні та слабкі сторони, і підприємства хочуть отримати найкраще з усіх світів.

1.2 Аналіз існуючих імовірнісних моделей черг для хмарних обчислень

У хмарних обчисленнях системи масового обслуговування (СМО) використовуються для моделювання потоків клієнтів, які надходять на ресурси хмари. Для аналізу таких систем використовуються імовірнісні моделі черг. Імовірнісні моделі черг описують СМО за допомогою математичної моделі, яка визначає розподіл ймовірностей кількості клієнтів у системі в будь-який момент часу. Ці моделі дозволяють оцінювати такі характеристики СМО, як середній час обслуговування клієнта, середній час перебування клієнта в системі та середня кількість клієнтів у системі. Існує безліч імовірнісних моделей черг, які можна використовувати для аналізу СМО в хмарних обчисленнях. Найбільш поширеними з них є:

– модель Галагера. Ця модель є найпростішою і найпоширенішою моделлю черг. Вона описує СМО, в якій потік клієнтів є постійним, а час обслуговування є постійним;

– модель M/M/1. Ця модель описує СМО, в якій потік клієнтів є розподілом Пуассона, а час обслуговування є розподілом експоненціального розподілу;

– модель M/M/c. Ця модель описує СМО, в якій є кілька серверів, які обслуговують клієнтів;

– модель G/G/1. Ця модель описує СМО, в якій потік клієнтів та час обслуговування є довільним розподілом [13].

Для порівняння імовірнісних моделей черг використовують такі характеристики:

– потік клієнтів. Потік клієнтів може бути постійним, розподілом Пуассона або довільним розподілом;

– час обслуговування. Час обслуговування може бути постійним, розподілом експоненціального розподілу або довільним розподілом;

– кількість серверів. У системі може бути один сервер або кілька серверів;

– складність реалізації. Модель може бути реалізована просто або складно;

– точність. Модель може бути більш точною або менш точною.

Таблиця 1.1 – Порівняння основних характеристик імовірнісних моделей черг

Характеристика	Модель Галагера [14]	Модель M/M/1 [14, 15]	Модель M/M/c [14, 15]	Модель G/G/1 [14, 15]
Потік клієнтів	Постійний	Пуассона	Пуассона	Довільний
Час обслуговування	Постійний	Експоненціальне	Експоненціальне	Довільний

Кінець таблиці 1.1

Кількість серверів	Один	Один	Кілька	Кілька
Складність реалізації	Проста	Середня	Складна	Складна
Точність	Невисока	Висока	Висока	Висока

На основі аналізу основних характеристик імовірнісних моделей черг можна зробити такі висновки, що модель Галагера є найпростішою і найменш точною моделлю. Вона описує систему, в якій потік клієнтів є постійним, а час обслуговування є постійним. Ця модель має такі переваги, як просту реалізацію та не значні ресурси на реалізацію. Проте ці переваги й призводять до недоліків, а саме невисокої точності моделі, це означає що модель може бути використана в разі, якщо потік клієнтів або час обслуговування не відповідають розподілу Пуассона. З іншого боку є модель M/M/1, що має більшу точність, аніж модель Галагера. Вона описує систему, в якій потік клієнтів є розподілом Пуассона, а час обслуговування є розподілом експоненціального розподілу. У цієї моделі є такі переваги як висока точність, але у разі, якщо потік клієнтів і час обслуговування відповідають розподілу Пуассона. В той час ця модель теж має недоліки такі як, складність її реалізації, порівняно з моделлю Галагера, а це означає що модель M/M/1 вимагає більше ресурсів, також порівняно з моделлю Галагера. Ще є модель M/M/c, яка є більш точною моделлю, ніж модель M/M/1 та модель Галагера. Вона описує систему, в якій в системі є кілька серверів. Ця модель має значну перевагу для системи з декількома серверами – високу точність. Але з цього й слідує ряд недоліків порівнюючи з моделями M/M/1 та Галагера: ще складнішу реалізацію та необхідність у більшій кількості ресурсів. Модель G/G/1 є найточнішої моделлю. Вона описує систему, в якій потік клієнтів і час обслуговування мають довільний

розподіл. Ця модель має найвищу точність, але в неї також найбільш складна реалізація та вимагає найбільше ресурсів.

На основі цих факторів можна зробити такі висновки:

– для систем з невеликим потоком клієнтів і простим розподілом часу обслуговування, наприклад, для зберігання даних, можна використовувати модель Галагера;

– для систем з великим потоком клієнтів або складним розподілом часу обслуговування, наприклад, для балансування навантаження, можна використовувати модель G/G/1;

– для систем з середнім потоком клієнтів і середнім розподілом часу обслуговування, наприклад, для обробки запитів, можна використовувати модель M/M/1 або модель M/M/c.

1.3 Теорія масового обслуговування як рішення для IT-компаній при використанні хмарних технологій

Теорія масового обслуговування є важливою і необхідною складовою для IT-компаній у сучасному цифровому світі. Ця теорія вивчає взаємодію між системою обслуговування та вхідними потоками, а також визначає оптимальні стратегії для обробки запитів та управління ресурсами в умовах зростаючого обсягу даних та транзакцій [15].

З одного боку, теорія масового обслуговування дозволяє IT-компаніям оптимізувати ефективність своїх обчислювальних ресурсів. З розвитком технологій та зростання обсягів даних, питання швидкості та ефективності обробки стає ключовим для компаній [16]. Застосування принципів теорії масового

обслуговування дозволяє оптимізувати час відгуку системи, зменшуючи час обробки запитів та покращуючи загальну продуктивність.

З іншого боку, теорія масового обслуговування стає критичною у ситуаціях, коли компанії намагаються підтримувати високу якість обслуговування для своїх клієнтів [17]. В IT-індустрії, де конкуренція жорстка, важливо забезпечувати надійність та доступність сервісів. Теорія масового обслуговування допомагає розробляти стратегії реагування на збільшення навантаження, що дозволяє компаніям уникати збоїв та забезпечувати стабільність в обслуговуванні своїх користувачів.

Теорія масового обслуговування є важливою і необхідною складовою для IT-компаній у сучасному цифровому світі. Ця теорія вивчає взаємодію між системою обслуговування та вхідними потоками, а також визначає оптимальні стратегії для обробки запитів та управління ресурсами в умовах зростаючого обсягу даних та транзакцій.

Модель Галагера має три основні функції розподілу [18]:

– функція розподілу числа клієнтів в системі. Вона визначає ймовірність того, що в системі буде n клієнтів. Для моделі Галагера ця функція має вигляд:

$$P(n) = \frac{\lambda^n e^{-\lambda}}{n!}, \qquad (1.1)$$

– функція розподілу часу очікування клієнта. Вона визначає ймовірність того, що клієнт, який прибув у систему, буде чекати t часу. Для моделі Галагера ця функція має вигляд:

$$P(t) = \frac{\lambda e^{-\lambda t}}{t}, \qquad (1.2)$$

– функція розподілу часу перебування клієнта в системі. Вона визначає ймовірність того, що клієнт, який прибув у систему, буде перебувати в ній t часу. Для моделі Галагера ця функція має вигляд:

$$P(t) = \frac{\lambda e^{-\lambda t}}{t} + \frac{\lambda^2 e^{-\lambda t}}{2t^2}. \quad (1.3)$$

З одного боку, теорія масового обслуговування дозволяє IT-компаніям оптимізувати ефективність своїх обчислювальних ресурсів. З розвитком технологій та зростання обсягів даних, питання швидкості та ефективності обробки стає ключовим для компаній. Застосування принципів теорії масового обслуговування дозволяє оптимізувати час відгуку системи, зменшуючи час обробки запитів та покращуючи загальну продуктивність.

З іншого боку, теорія масового обслуговування стає критичною у ситуаціях, коли компанії намагаються підтримувати високу якість обслуговування для своїх клієнтів. В IT-індустрії, де конкуренція жорстка, важливо забезпечувати надійність та доступність сервісів. Теорія масового обслуговування допомагає розробляти стратегії реагування на збільшення навантаження, що дозволяє компаніям уникати збоїв та забезпечувати стабільність в обслуговуванні своїх користувачів.

Поважаючи принципи теорії масового обслуговування, IT-компанії можуть ефективно планувати свої ресурси, забезпечуючи оптимальне використання обчислювальної потужності та високу якість обслуговування. Ця теорія стає ключовою для розуміння та прогнозування обсягів роботи, розробки ефективних алгоритмів обробки даних та підтримки стабільності та надійності IT-систем. У сумі, вона є невід'ємною частиною стратегічного управління технологічними ресурсами в умовах швидко змінюваного та конкурентного IT-середовища.

1.4 Постановка задачі

Мета дослідження полягає в розробці та оптимізації моделей черг для вдосконалення роботи інформаційних систем IT-компаній. Для досягнення цієї мети необхідно вирішити завдання. Потрібно провести аналіз уже існуючих моделей черг для хмарних обчислень. Це завдання необхідно для того, щоб визначити сильні та слабкі сторони існуючих моделей, а також виявити можливості для їхнього вдосконалення. Аналіз дозволить розробникам нових моделей уникнути помилок, що вже були допущені в існуючих моделях, а також розробити більш ефективні моделі. Також потрібно розробити нові моделі черг, які враховують особливості інформаційних систем IT-компаній. Це завдання необхідно для того, щоб розробити моделі, які будуть відповідати специфічним вимогам IT-компаній. Такі моделі будуть більш ефективними та точними, ніж існуючі моделі, які не враховують особливості IT-компаній. А також оптимізувати існуючі моделі черг для підвищення їхньої ефективності. Це завдання необхідно для того, щоб підвищити ефективність існуючих моделей. Оптимізація може бути проведена шляхом уточнення припущень, що використовуються в моделях, вибору оптимальних значень вихідних параметрів моделей або використання нових методів моделювання.

Очікуваними результатами дослідження є оптимізована існуюча модель черг з метою підвищення її ефективності черг для вдосконалення роботи інформаційних систем IT-компаній.

Розроблені та оптимізовані моделі черг можуть бути використані для вдосконалення роботи інформаційних систем IT-компаній у таких аспектах:

- зменшення часу обслуговування клієнтів;
- зниження витрат на обслуговування черги;
- покращення якості обслуговування клієнтів;

– оптимізація витрат на ресурси.

Таким чином, дослідження має важливе значення для підвищення ефективності роботи інформаційних систем ІТ-компаній.

У рамках цього напрямку буде проведено аналіз існуючих моделей черг для хмарних обчислень. Аналіз буде включати в себе такі аспекти:

– типи моделей черг. Типи моделей черг визначаються тим, як вони описують поведінку черги. Існує багато різних типів моделей черг;

– припущення, що використовуються в моделях. Припущення, що використовуються в моделях, є обмеженнями, які накладаються на поведінку черги;

– вихідні параметри моделей. Вихідні параметри моделей є величинами, які можна виміряти або оцінити з допомогою моделей;

– результати моделювання. Результати моделювання є величинами, які вимірюються або оцінюються за допомогою моделей.

Аналіз існуючих моделей черг дозволить визначити їхні сильні та слабкі сторони, а також виявити можливості для їхнього вдосконалення.

У рамках цього напрямку будуть розроблені нові моделі черг, які враховують особливості інформаційних систем ІТ-компаній. При розробці моделей буде враховуватися такі фактори:

– типи завдань, які виконуються в ІС ІТ-компаній;

– навантаження на ІС ІТ-компаній;

– вимоги до якості обслуговування клієнтів.

Розроблені нові моделі черг будуть перевірятися на відповідність поставленим вимогам.

У рамках цього напрямку будуть оптимізовані існуючі моделі черг для підвищення їхньої ефективності. Оптимізація буде проводитися за такими напрямками:

– уточнення припущень, що використовуються в моделях;

– вибір оптимальних значень вихідних параметрів моделей;

– використання нових методів моделювання.

У межах цієї моделі буде проведений аналіз існуючих досліджень у галузі хмарних обчислень та моделей черг. Також буде проведений аналіз статистичних даних про навантаження на ІС ІТ-компаній та вимоги до якості обслуговування клієнтів. Ще у рамках магістерської роботи будуть вдосконалені та протестовані моделі черг.

Таким чином, дослідження має на меті розробку та оптимізацію моделей черг для вдосконалення роботи інформаційних систем ІТ-компаній. Очікувані результати дослідження мають важливе значення для підвищення ефективності роботи ІС ІТ-компаній.

Популярність та необхідність використання різних моделей теорії масового обслуговування стають невід'ємною частиною стратегічного управління для ІТ-компаній у сучасному бізнес-середовищі. Розуміння та ефективне впровадження цих моделей дозволяє підприємствам гнучко реагувати на зміни у завданнях та транзакційних потоках.

Однією з популярних моделей є модель M/M/c, яка вивчає систему обслуговування з обслуговуючими каналами та покликана оптимізувати роботу системи при різних рівнях завантаження. Ця модель особливо ефективна в умовах коливань обсягів роботи та може бути використана для прогнозування необхідності розширення або скорочення обчислювальних ресурсів в залежності від попиту [19]. Модель M/G/1, яка розглядає систему з одним обслуговуючим каналом та генеральним розподілом часів обслуговування, стає корисною для врахування неоднорідності у величинах обслуговування, що може виникати в реальних обчислювальних системах. Застосування моделей таких, як M/G/∞ або M/D/c, також є важливим для вирішення конкретних завдань у сферах фінансів, медицини чи роздрібної торгівлі, де потрібно враховувати специфічні аспекти обслуговування та управління даними. Однак, важливо зауважити, що жодна модель не є універсальною. Ефективність використання конкретної моделі залежить від

конкретних умов та завдань, що ставить перед собою кожна ІТ-компанія. Таким чином, правильний вибір моделі та її адаптація до конкретного контексту дозволяють досягти оптимальних результатів у реальних умовах роботи системи обслуговування.

2 РОЗРОБКА ВДОСКОНАЛЕНОЇ МОДЕЛІ ЧЕРГ ДЛЯ ВИРІШЕННЯ ЗАДАЧ ОПТИМАЛЬНОГО ВИКОРИСТАННЯ ХМАРНИХ РЕСУРСІВ ІТ-КОМПАНІЙ

2.1 Підхід до вирішення задачі модифікації моделі системи масового обслуговування

Проблематика задачі описана в підрозділі 1.4 може бути вирішена за допомогою Q-схем та її декомпозиції, кінцевим результатом повинна бути побудована система масового обслуговування та протестована її ефективність. Для побудови системи масового обслуговування (СМО) потрібно визначитись з їх класифікацією та обрати найбільш близьку для вирішення поставленої задачі.

Системи масового обслуговування можна поділити за кількістю каналів самої системи та наліченням відмов або відсутністю відмов. За кількістю каналів СМО можуть бути одноканальні, двоканальні, тощо та багатоканальні [20]. Зазвичай у багатоканальних систем їх кількість каналів позначається латинською літерою «n», тому надалі інколи багатоканальна система буде означатися як n-канальна система. У випадку встановленому для вирішення задачі поставлена система налічування каналів якої є не вирішеною і може відрізнятися в процесі її розробки та в залежності від чинників що не можна передбачити, то найближчою підходящою системою за кількістю каналів можна вважати багатоканальну або n-канальну систему. Системи без відмов є скоріше теоретичними аніж практичними у використання. Головна відмінність СМО без відмов до СМО з відмовами полягає у тому як будуть поводити себе заявки у випадку перенавантаження головного сервера або інших елементів системи. Заявки у випадку перенавантаження для СМО без відмов не будуть викинуті з системи, а лише очікуватимуть моменту у часі коли з’явиться місце для того, щоб вони могли перейти до іншого стану системи, тобто допоки не з’явиться місце в переповненому накопичувачі, тощо. В реальних

же системах у разі переповнення кешу або бази даних зберігати заявки не буде можливості, і в такому разі системі доведеться або зупиняти весь процес обробки або відхилити заявку, тобто видалити з системи. Найбільш близький варіант для вирішення поставленої задачі буде відкидання заявки з системи у разі переповнення накопичувача або перенавантаження серверу (якщо заявку не буде можливості десь зберігати). У випадку відкидання заявки можливо буде уникнути ситуації зупинки обробки всієї системи, що є більш прийнятним. Тому обрана система повинна мати можливість відкидання заявок, тобто необхідною системою є СМО з відмовами. Також є необхідність вказати про СМО без відмов зазвичай передбачає буфер заявок в якому заявка може очікувати допоки не з’явиться місце в черзі, проте у необхідної до реалізації системи не має бути великого значення для кожної заявки системи. Заявки, як вже було згадано у першому розділі, це лише запити на виконання якихось дій, вони можуть бути повернуті користувачеві з повідомленням про перевантаженість системи, після чого в разі потреби він може повторно відправити запит на обробку, а в разі не великої значимості системи може зачекати допоки система не зможе працювати в належному стані. Передбачена система повинна акцентувати увагу на надійності роботи самої системи, до того ж моменти перенавантаження системи матимуть значний вплив хоча і очікувана частота таких подій є незначною.

Розглянемо структурну схеми загальної простої СМО (рисунок 2.1), тобто одноканальної. Ця структурна схема складається з ДЗ, що є джерелом заявок на їх подальше обслуговування, Н – накопичувач, що представляє собою чергу заявок; ОП – обслуговуючого пристрою, він повинен обробляти заявки що в нього потрапляють; С – споживача заявок, в нього потраплятимуть заявки що були оброблені належним чином. Заради простоти в системі не налічується ніяких розгалужень, блокувань на видачі, тощо [21].

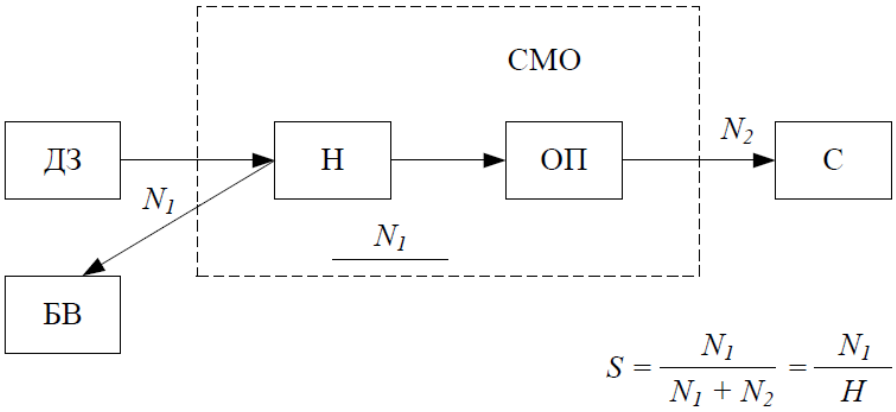

Рисунок 2.1 – Загальна структурна схема просто СМО [21]

У відповідності до вказаних вимог поставленої задачі представленої у підрозділі 1.4 найближчим рішенням матиме місце побудова Q-схем. Також є необхідність для декомпозиції тобто опису і розбиття Q-схеми на блоки роботи системи, а також опис алгоритму роботи цих блоків представлених у виді схем, що буде реалізовано у 3-му розділі.

Є декілька способів побудови моделюючих алгоритмів Q-схем, їх можна класифікувати за такими ознаками як:

а) залежність від випадкових величин:

1) детерміновані алгоритми не використовують випадкові величини. Вони описують поведінку системи в термінах ймовірностей, але не генерують випадкові події;
2) стохастичні алгоритми використовують випадкові величини для генерування випадкових подій, які впливають на поведінку системи;

б) часовий хід моделі:

1) асинхронні алгоритми генерують події в будь-який момент часу;
2) синхронні алгоритми генерують події тільки в дискретні моменти часу;

в) часовий горизонт моделі:

1) циклічні алгоритми моделюють систему в циклі з кінцевим числом періодів;
2) спорадичні алгоритми моделюють систему в нескінченному циклі.

Класифікація способів побудови моделюючих алгоритмів Q-схем представлена на рисунку 2.2.

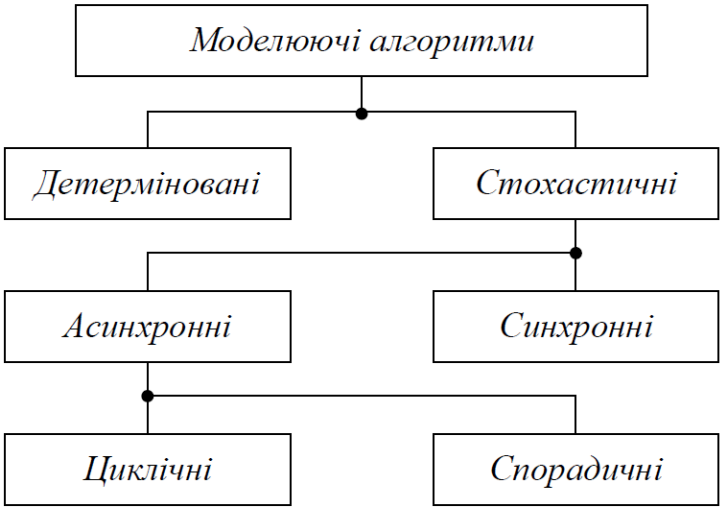

Рисунок 2.2 – Класифікація способів побудови моделюючих алгоритмів Q-схем [21]

2.2 Удосконалена модель СМО для вирішення поставленої задачі

Виходячи з вказаних вимог до поставленої задачі у підрозділі 1.4 можна зробити такі висновки, що стосуються заявок.

Великий обсяг заявок може призвести до значного навантаження на систему, а також до затримки виведення результатів. Це, у свою чергу, може спричинити на задовільних досвід використання системи для користувачів, а також можу

збільшити час обробки запитів та позначитись на стабільності роботи системи, тобто ускладнити якість роботи системи.

Щоб вирішити цю проблему, запропоновано модифікації, які значно зменшать витрати, та покращать стабільність та якість роботи системи.

Ця модифікація полягає в тому, що до моделі додається додаткове обмеження, яке встановлює максимальний час перебування заявки в системі. Це обмеження дозволяє забезпечити достатній рівень якості обслуговування заявок, а також гарантує, що заявки будуть обслуговуватися в передбачені терміни.

При наявності обмеження за часом перебування заявок в системі необхідно в якості особливих станів Q-схеми розглядати не лише моменти надходження нових заявок та моменти закінчення обслуговування заявок, але й моменти закінчення припустимого часу перебування (очікування, обслуговування) заявок в майбутніх Q-схемах.

Це означає, що якщо заявка не була обслужена до закінчення встановленого обмеження, вона повинна бути відхилена або переправлена в іншу систему.

Обмеження за часом перебування заявок в системі є важливим для систем, де заявки обробляються в реальному часі. Воно допомагає забезпечити, щоб заявки не затримувалися в системі надовго, що є критичним для таких систем, як плановане розроблене програмне забезпечення, та схожі системи.

Нижче наведено додаткові пояснення щодо модифікації:

3) додаткове обмеження встановлюється на початку роботи програмного продукту. Це дозволяє гарантувати, що всі заявки будуть обслуговуватися в рамках встановленого обмеження;

4) при наявності обмеження за часом перебування заявок в системі необхідно враховувати не лише моменти надходження нових заявок та закінчення обслуговування заявок, але й моменти закінчення припустимого часу перебування заявок в майбутніх Q-схемах. У заявок має бути властивість, що буде відображати

час знаходження заявки в системі. Сама ж система повинна мати можливість відстежувати значення цієї властивості;

5) якщо заявка не була обслужена до закінчення встановленого обмеження, вона повинна бути відхилена або переправлена в іншу систему. Це дозволяє запобігти ситуації, коли заявка залишається в системі надовго, що може призвести до негативних наслідків для системи;

6) у разі поломки накопичувача або каналу, а також у разі отримання помилки, всі елементи каналу переходять у режим “down” (recovery). Це означає, що канал припиняє свою роботу та переходить до виконання процедур відновлення.

Після переходу у режим відновлення система за можливості перевіряє стан заявки, яка була в процесі обробки і за можливості встановлює заявку в кінець черги, де вона буде оброблена працюючою частиною системи.

Якщо ж заявка пошкоджена, вона не повертається у чергу. Замість цього вона очікує, поки канал із заявкою не вийде з режиму відновлення. Після цього заявка повертається у кінець черги на повторну обробку належним чином.

Таке рішення дозволяє запобігти пошкодженню інших заявок, які знаходяться в системі. Воно також дозволяє відновити роботу системи якомога швидше.

3 МОДЕЛЮВАННЯ ПРОЦЕСІВ ФУНКЦІОНУВАННЯ СИСТЕМ МОДИФІКОВАНОЇ ТА ОРИГІНАЛЬНОЇ ПУАССОНІВСЬКОЇ МОДЕЛІ

3.1 Моделювання процесів функціонування систем на базі Q-схем для n канальної СМО з відмовами

Для опису процесу функціонування n канальної системи масового обслуговування з відмовами було обрано моделювання процесів на базі Q-схем. Позначимо цю СМО латинською літерою S. У роботі цієї системи є характерна ситуація з появою заявок (вимог) для обробки, а також завершення обробки і ці дії відбуваються у випадкові моменти часу. З цього можна зробити висновок що система має стохастичний характер функціонування. Вхідний потік в загальному випадку таких систем утворюють моменти надходження заявок із зовнішнього середовища E до системи S, а вихідний потік утворюється моментами закінчення обробки цих заявок в системі. Формалізація на базі Q-схем. Формалізуючи систему використаної у цій роботі за допомогою Q-схеми, було побудовано структуру такої системи. В якості елементів структури Q-схем будемо розглядати елементи типів які є прийнятними для структур Q-схем, а саме:

7) Д – джерела – елементи, які генерують заявки на обслуговування;

8) Н – накопичувачі – елементи, які зберігають заявки на обслуговування, які не можуть бути обслужені одразу;

9) К – канали обслуговування заявок – елементи, які обслуговують заявки.

Джерело заявки генерує заявки з певною інтенсивністю. Інтенсивність генерації заявок являє собою середню кількість заявок, які генеруються джерелом за певну одиницю часу. Накопичувач заявок може зберігати певну кількість заявок. Коли накопичувач заповнюється, заявки, які надходять на нього, відхиляються. Канал обслуговування заявок обслуговує заявки з певною швидкістю. Швидкість

обслуговування заявок — це середня кількість заявок, які можуть бути обслужені каналом за одиницю часу [21].

На рисунку 3.1 представлена структура n канальної СМО стохастичної синхронної. Ця структура складається з наступних елементів:

10) джерело заявок (Q) генерує заявки з певною інтенсивністю;

11) накопичувач заявок (N) може зберігати певну кількість заявок;

12) n каналів обслуговування (K1, K2, ..., Kn) обслуговують заявки.

Вихідний потік обслугованих заявок (E) утворюється заявками, які були успішно обслуговані.

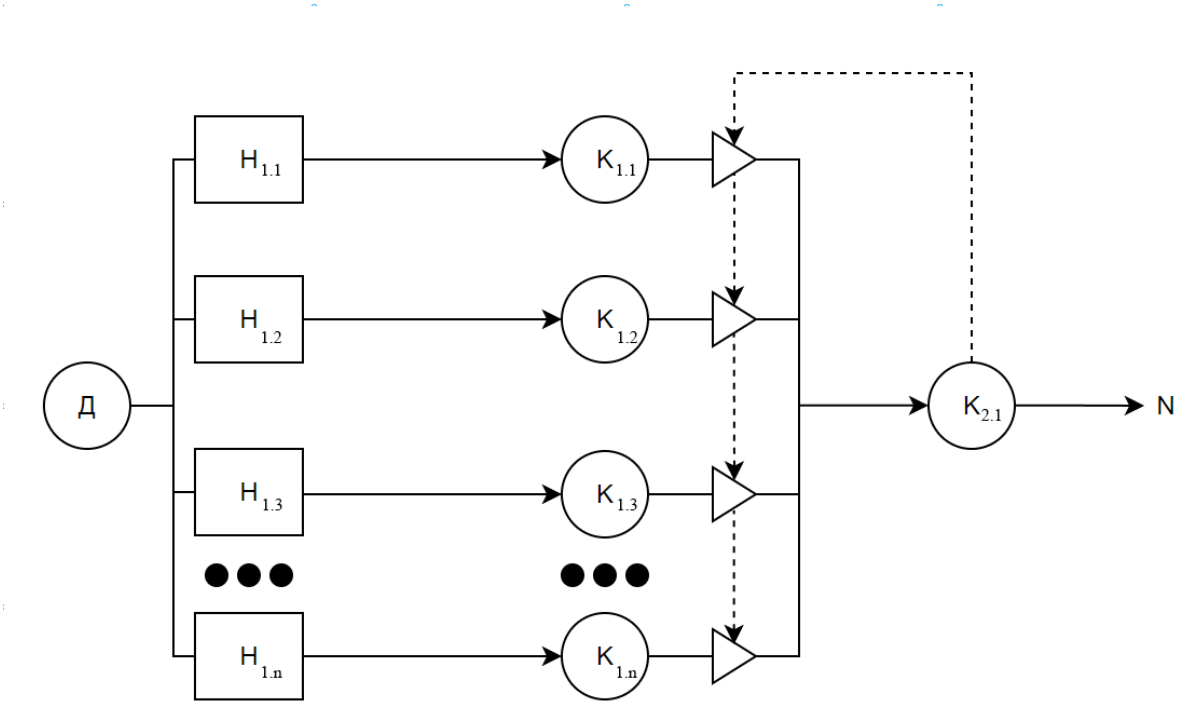

Рисунок 3.1 – Структура системи, представленої у вигляді Q-схеми

Керуючі зв'язки. На схемі зображені як зв’язки що відображають рух заявок в Q-схемі так і різні керуючі зв’язки (рух заявок представлений суцільними лініями). Різні блокування обслуговуючих каналів (за входом та виходом) є прикладом таких зв'язків. Блокування обслуговуючих зв’язків прийнято називати «клапанами» [21],

вони зображені у вигляді трикутників, а керуючі зв'язки — пунктирними лініями. Під блокуванням каналу мається на увазі що цей (блокований) канал відмикається від вхідного потоку заявок. Це може бути зроблено, наприклад, якщо канал перебуває у стані ремонту, що буде детальніше описано в модифікованій моделі підрозділу 3.2. У блокованому каналі заявка буде залишатися до моменту зняття блокування з цього каналу, у випадку якщо заявка вже повністю їм оброблена. Це може бути зроблено, наприклад, для того, щоб уникнути втрат заявок при переповненні накопичувача.

Втрати заявок. У випадку відсутності «клапана» заявка буде втрачена при його переповненості. Це означає, що деякі заявки будуть відкинуті, оскільки вони не зможуть потрапити до одного з каналів обслуговування.

Потоки подій Q-схеми описують, як заявки переміщуються між елементами системи. Вхідні потоки заявок (Д) представляють заявки, які надходять до системи з зовнішнього середовища. Потоки обслуговування представляють заявки, які обслуговуються каналами обслуговування. Структура системи визначає, які елементи входять до системи, як вони взаємодіють один з одним і як заявки переміщуються між ними. Кількість фаз (L*) визначає кількість різних станів, які може приймати система. Кількість каналів обслуговування (L^k) визначає, скільки каналів обслуговування одночасно може обслуговувати заявки. Кількість накопичувачів (L^H) визначає, скільки накопичувачів може зберігати заявки, які не можуть бути обслужені одразу. Зв'язок (Д, Н, К) визначає, як заявки переміщуються між елементами системи. Алгоритми функціонування системи визначають, як заявки обробляються в системі. Дисципліни очікування заявок в накопичувачах (Q) визначають, як заявки, які не можуть бути обслужені одразу, ранжуються в чергах. Дисципліни вибору на обслуговування (Р) визначають, які заявки будуть обслуговуватися першими. Правила відходу заявок з накопичувачів і каналів обслуговування визначають, як заявки залишають систему після обслуговування.

При моделюванні Q-схеми використовується стохастичний асинхронний алгоритм. Загальна схема алгоритму представлена на рисунку 3.2.

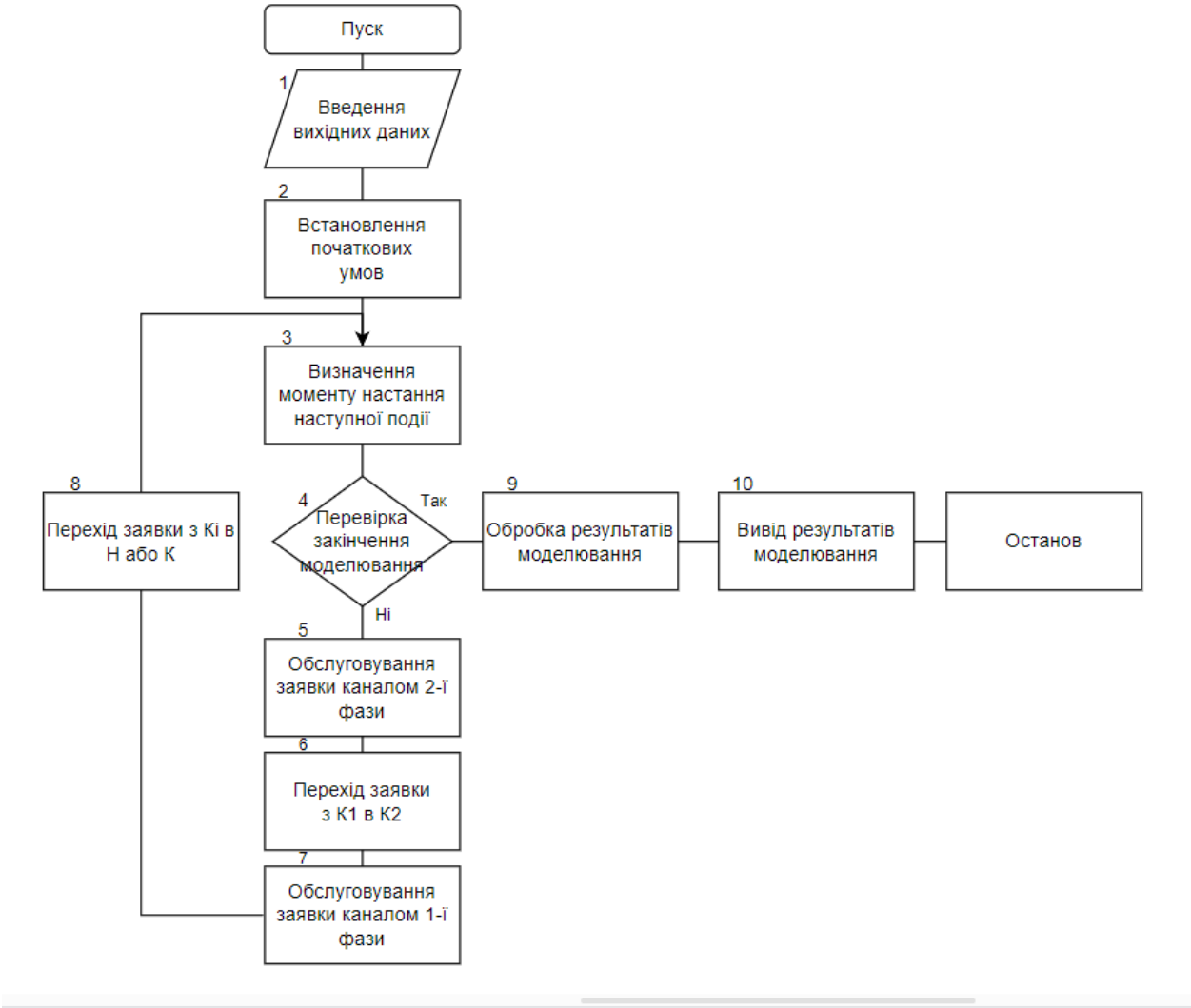

Рисунок 3.2 – Укрупнена схема стохастичного асинхронного алгоритму Q-схеми

Для зупинки алгоритму використовується перевірка певних умов (зображено на рисунку блоком 4). Допоміжні блоки (введення вхідних даних 1, встановлення початкових умов 2, обробка 9 та виведення результатів моделювання 10) не відрізняються від аналогічних блоків у алгоритмах обчислень на комп’ютері. На

рисунках 3.3-3.8 наведені деталізовані схеми для цих блоків. На цих і подальших схемах прийняті наступні позначення:

$$ZN(I) = z_1;\ \ Z(K,J) = z_{K,j};\ \ TM = t_m,\ \ TN = t_n,\ \ T(K,J) = t_{k,j},$$
$$LO(I) = L_i,\ \ PO = P,\ \ NO_1 = N_1,\ \ NO_3 = N_3,\ \ NO = N \qquad (3.1)$$

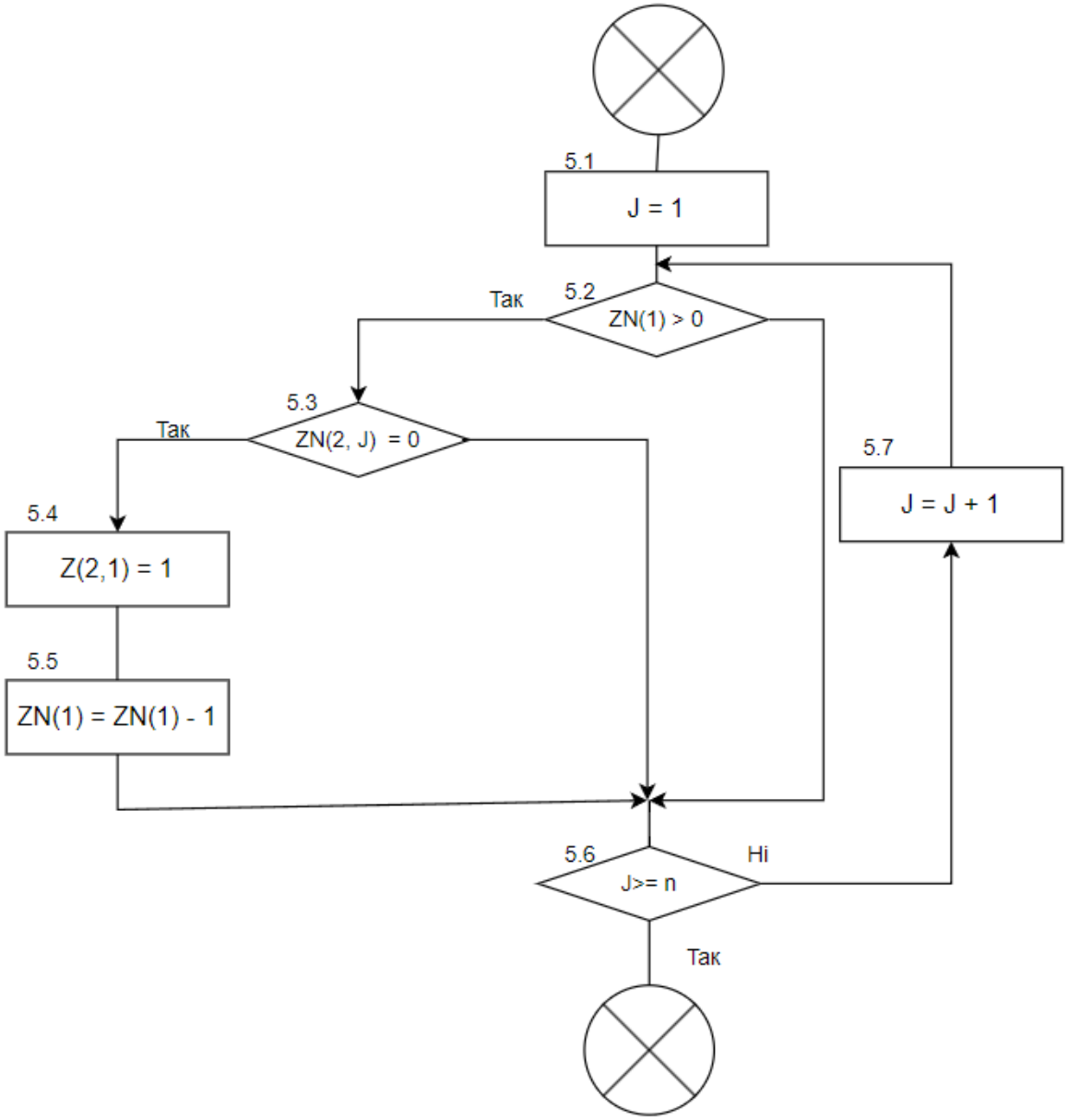

Рисунок 3.3 – Схема алгоритмів 5 блоку

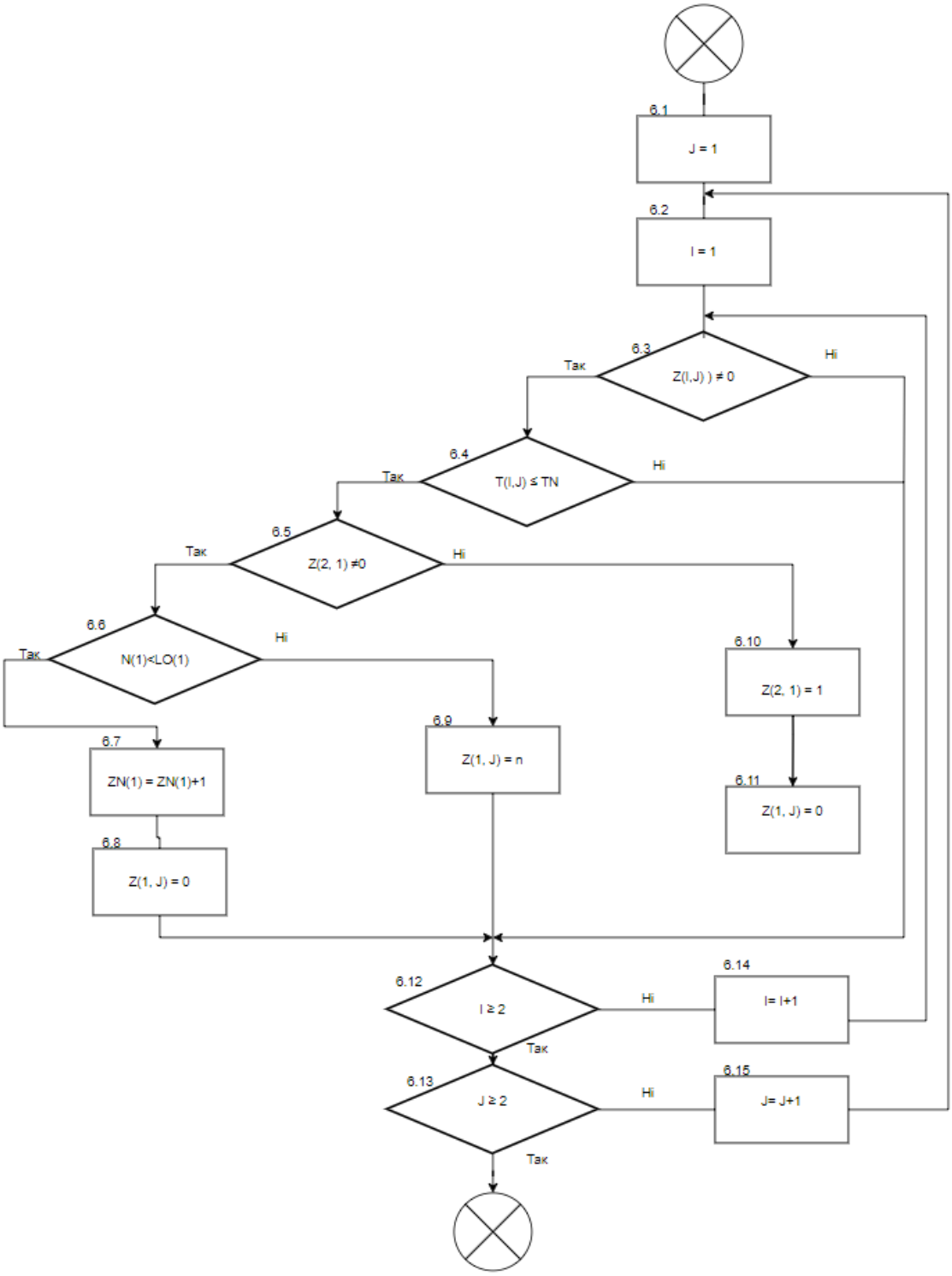

Рисунок 3.4 – Схема алгоритмів 6 блоку

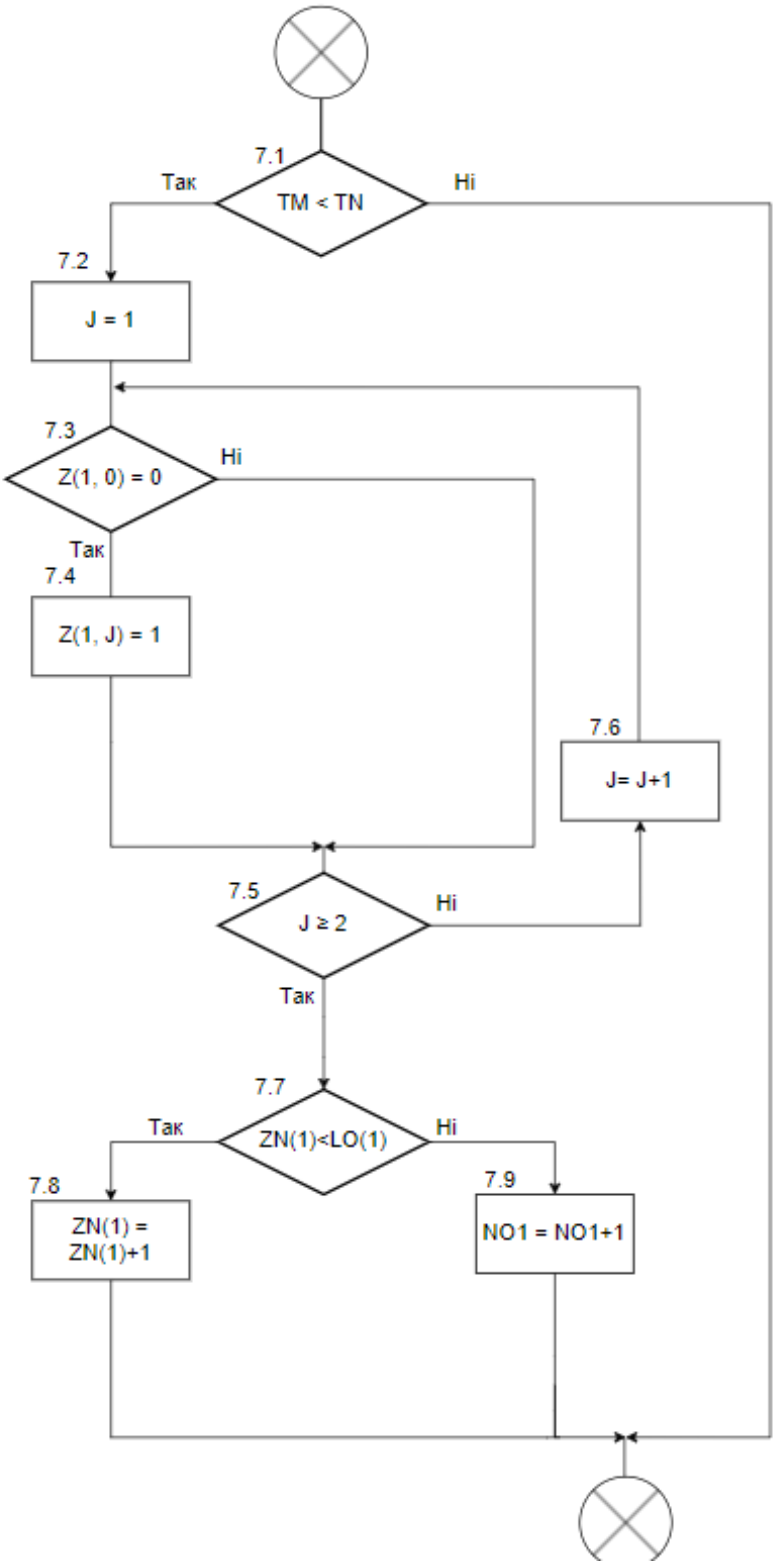

Рисунок 3.5 – Схема алгоритму блоку 7

Обслуговування заявок каналами $K_{j,k}$, яка звертається до генератора випадкових чисел з відповідним законом розподілу. Тривалість обслуговування чергової заявки $t_{k,j}$генерується за цим законом. Літерою «Д» позначається генерація заявок джерелом.

Після початку робот системи, установлення початкових умов, введення вхідних даних йду перевірка на закінчення моделювання системи, за це відповідають перші блоки зображені на рисунку 3.3. Далі на рисунку на рисунку 3.5 зображена схема алгоритмів 5-го блоку. В цій схемі першою дією виконується послідовний перегляд каналів цієї фази. Потім йде перевірка на знаходження заявок в каналах 1-ї фази, які б очікували на обслуговування в каналі $K_{2,1}$. Якщо на часовий момент t_n є заявки, що очікують на обслуговування каналом $K_{2,1}$, а до того ж канал не має зайнятості обслуговуванням іншої заявки, то обирається одна з них та поміщається в каналі $K_{2,1}$. Після цього вказується зайнятість каналу $K_{2,1}$. Якщо ж канал $K_{2,1}$ в цей момент часу зайнятий, тобто вже обслуговує іншу заявку, то в такому разі встановлюється значення блокування для каналу 2-ої фази.

У шостому блоці імітується взаємодія в процесі обслуговування заявок для накопичувачів та каналів 1-ої фази, у послідовності (рисунок 3.6). У випадку налічування заявки у накопичувачі $Н_1$ (зображено у операторі 6.2), а також канали першої фази є вільними (зображено у операторі 6.3), тоді виконується обробка заявки, після чого звільняється місце для накопичувача $Н_1$ (зображена у операторах 6.4-6.6).

Далі йде блок 7 (зображений на рисунку 3.7), на випадок незайнятості каналів виконується взаємодія з джерелом Д. В цьому операторі є допоміжні оператори циклів – це оператори 7.2, 7.5, 7.6.За наявності вільного каналу, при надходженні заявки в момент часу t_n вона буде обробленаодним з каналів $K_{1,j}$.Якщо ж канали э зайнятими то заявка може встати в чергу до накопичувача, проте якщо у самому

накопичувачі є місце. Якщо місця для нової заявки немає у жодному з накопичувачів то заявка буде відхилена (втрачена).

Потім управління знову передається блоку 4. На момент набору достатньої статистики цей блок виконує обробку та передачу результатів моделювання, після чого він же виконує зупинку моделювання (зображено блоками 9 та 10).

3.2 Моделювання процесів функціонування систем на базі Q-схем для модифікованої n канальної СМО з відмовами

На рисунку 3.9 представлена структура системи, представленої у вигляді Q-схеми для модифікованої n канальної СМО. Модифікації були виконані, як вже було вказано раніше з врахуванням виходу з ладу деяких елементів системи а також з обмеженням за часом перебування в системі для заявок.

Детальніше зупинимося на відмінностях в схемах між модифікованою та не оригінальною системами. На схемі видно появу нових елементів системи як: $K_{1.t}$, $K_{1.1.d}$, $H_{1.1.d}$, саме в цих елементів і є модифікація. $K_{1.t}$ – створений для заявок що вийшли за час обмежений на очікування, тобто в нього потрапляють заявки час обробки або очікування котрих вийшов за межі встановлені системою (T_timeout). На цій стадії заявки становляться втраченими через занадто довге очікування, а загальна кількість таких заявок збільшується на 1. Далі $K_{1.1.d}$, $H_{1.1.d}$ та їх n копії існують в системі на випадок виходу з ладу відповідно $K_{1.1}$, $H_{1.1}$ та n елементів. Елементи системи потрапляють у цей стан у разі виникнення неочікуваних поломок цих елементів, ці елементи системи призупиняють свою роботу, разом з пошкодженими елементами системи призупиняють обробку і заявки, в цей час інші, працюючі елементи системи продовжують свою роботу. Після того як ці частини ввійдуть в налагоджений стан, заявки, що не були відпрацьовані повернуться до

кінця черги та пройдуть цикл спочатку, а вже налагоджені елементи системи повернуться до нормальної роботи. Час знаходження заявок в режимі down (recovery) не буде враховуватися для перевірки на час заявки в очікувані.

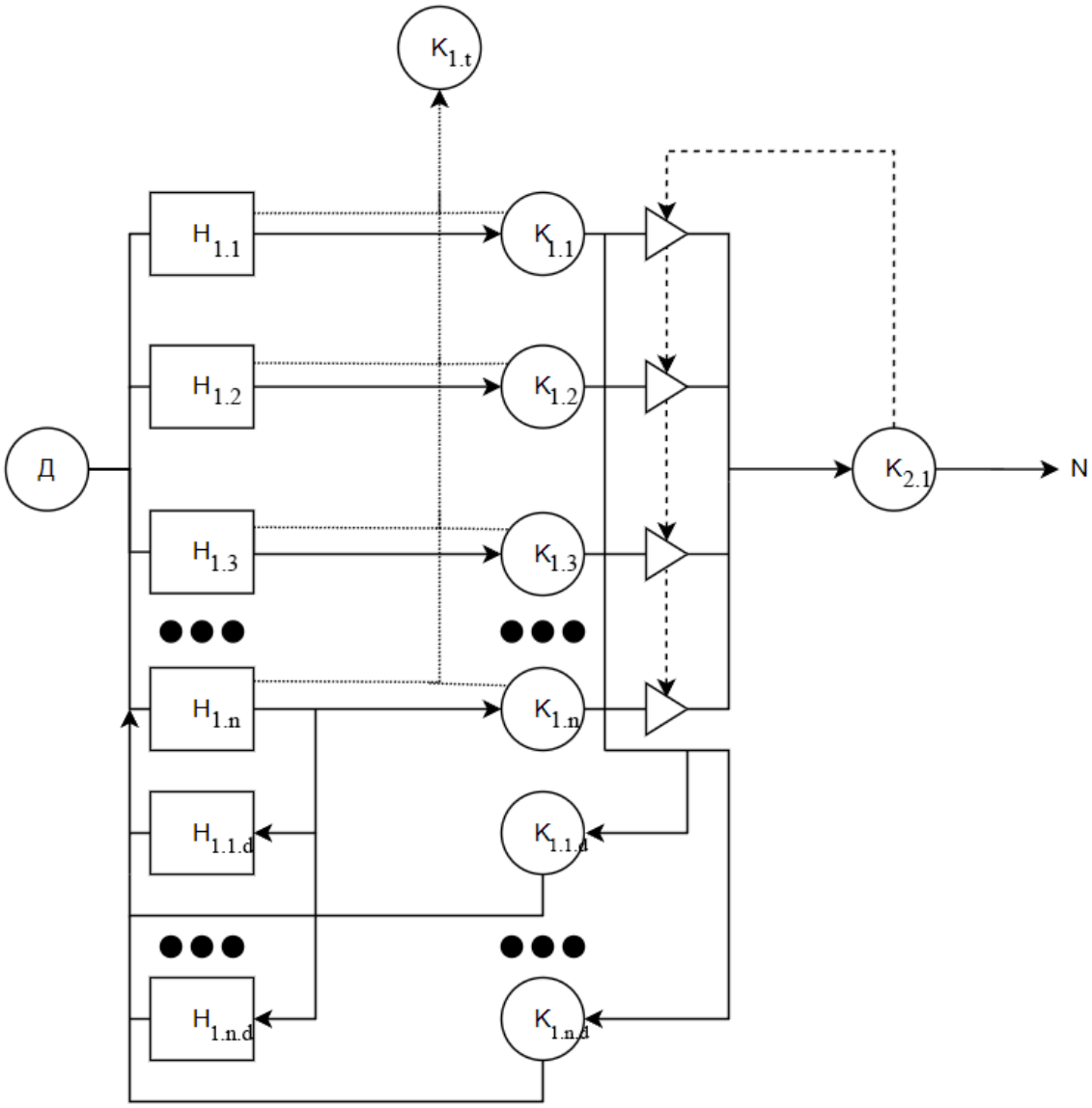

Рисунок 3.6 – Структура системи, представленої у вигляді Q-схеми для модифікованої n канальної СМО

На рисунку 3.8 представлена укрупнена схема модифікованої СМО алгоритму Q-схеми. Детальніше розглянемо її ключові відмінності, а саме блоки 8 та 10 на яких зображена модифікація врахування виходу елементів системи з ладу

та їх подальше відновлення. А також ключова відмінність є в декомпозиції блоків 6 та 7 представлені на рисунках 4.4 та 4.5 відповідно. Після 7-го блоку слідує перевірка виходу елементів системи з ладу, в якій буде зафіксовано стан елементу або елементів системи та переведено вийшовших з ладу елементів в стан down (recovery). Після відновлення нормальної роботи цих елементів заявка перейде в початковий стан і пройде повний цикл починаючи з блоку 3.

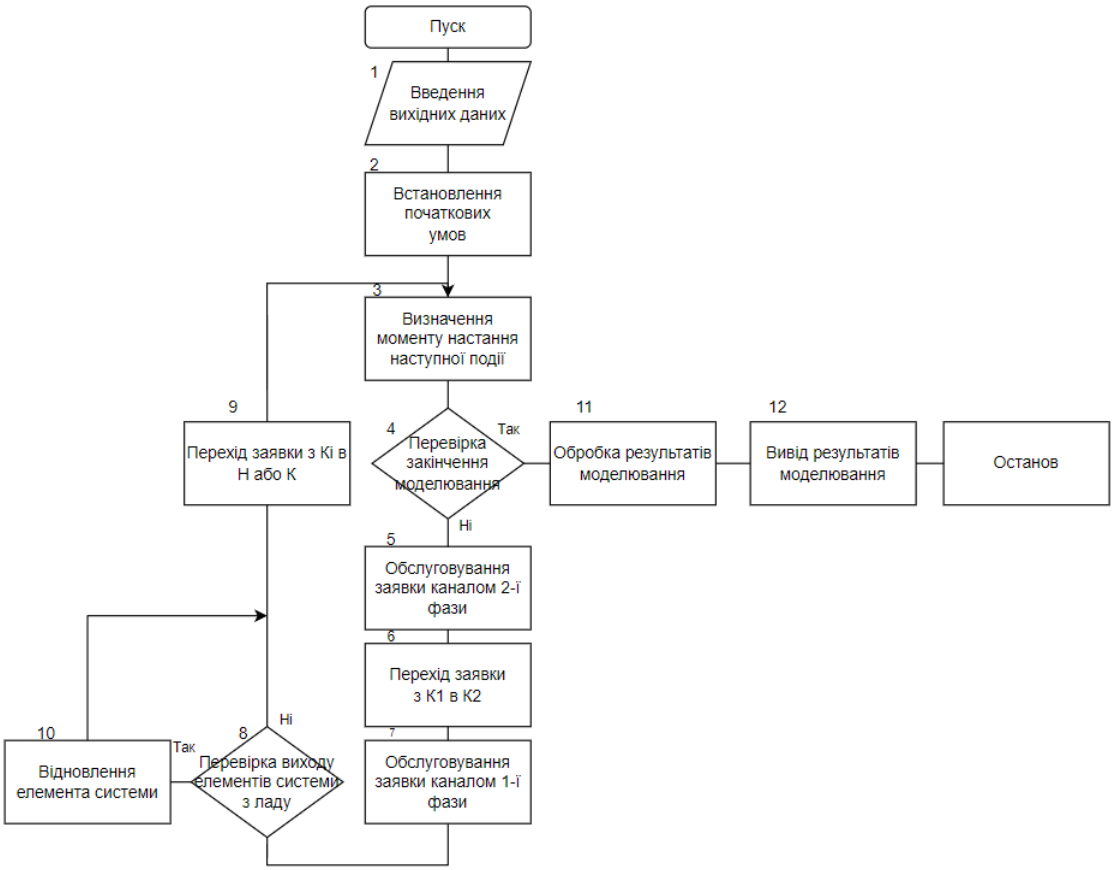

Рисунок 3.7 – Укрупнена схема модифікованої СМО алгоритму Q-схеми

На схемі алгоритму 5-го блоку ключовою відмінністю є перевірка часу очікування заявки на вихід за обмеження на час очікування заявок (T_timeout). У разі якщо заявка перевищила цей час то лічильник заявок відхилених через перевищення ліміту на час очікування збільшується на один, а сама заявка

видаляється з системи. Якщо ж заявка не перевищила T_timeout потік дій для заявки не відрізняється від немодифікованої схеми.

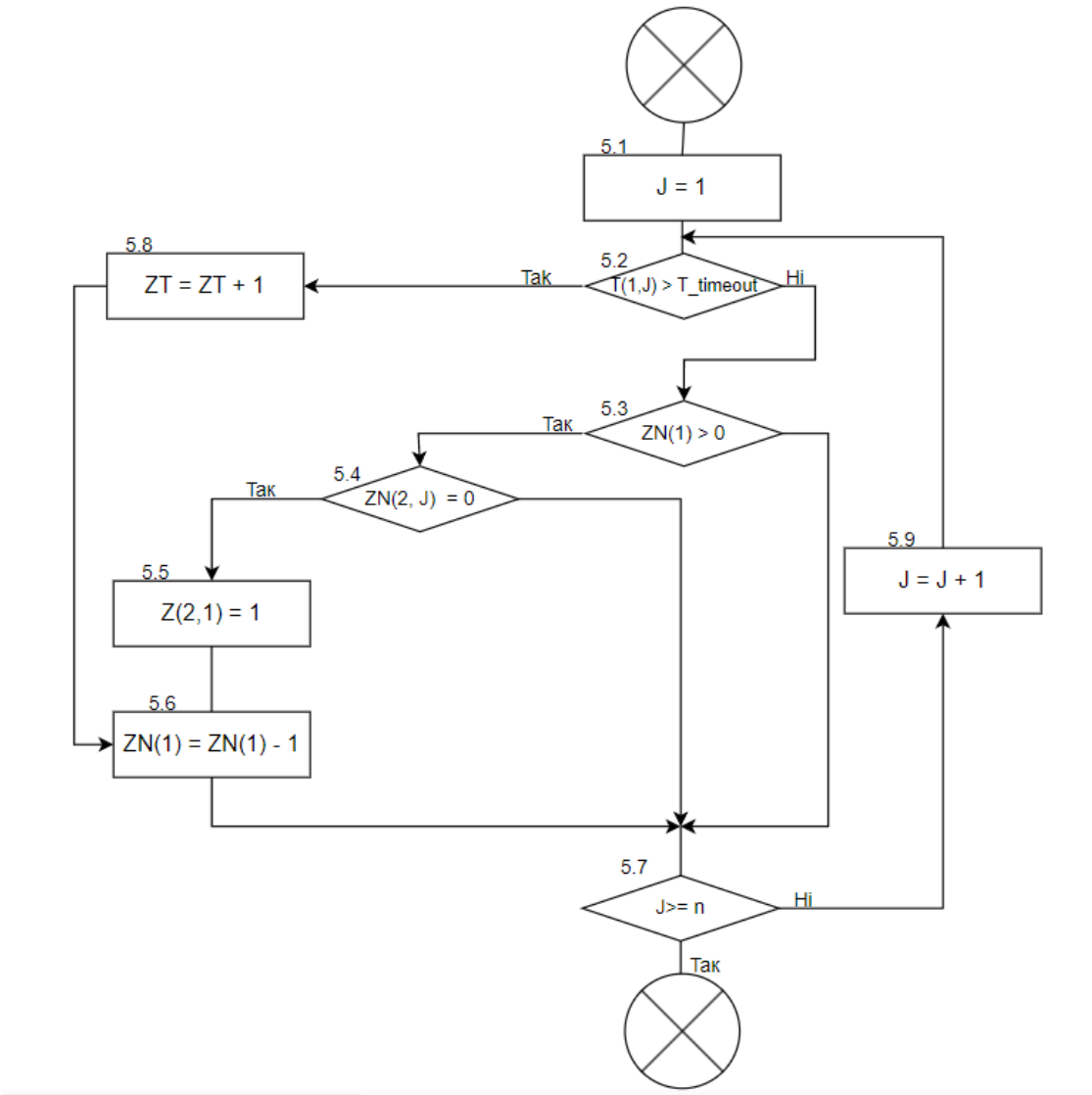

Рисунок 3.8 – Схеми алгоритмів 5-го блоку

Ключовою відмінністю для блоку 6, також як і для блоку 5 є наличення перевірки заявки на перевищення ліміту перебування в очікуванні. І сутність цієї перевірки за логічністю не відрізняється від блоку 5. Те ж стосується і 7-го блоку.

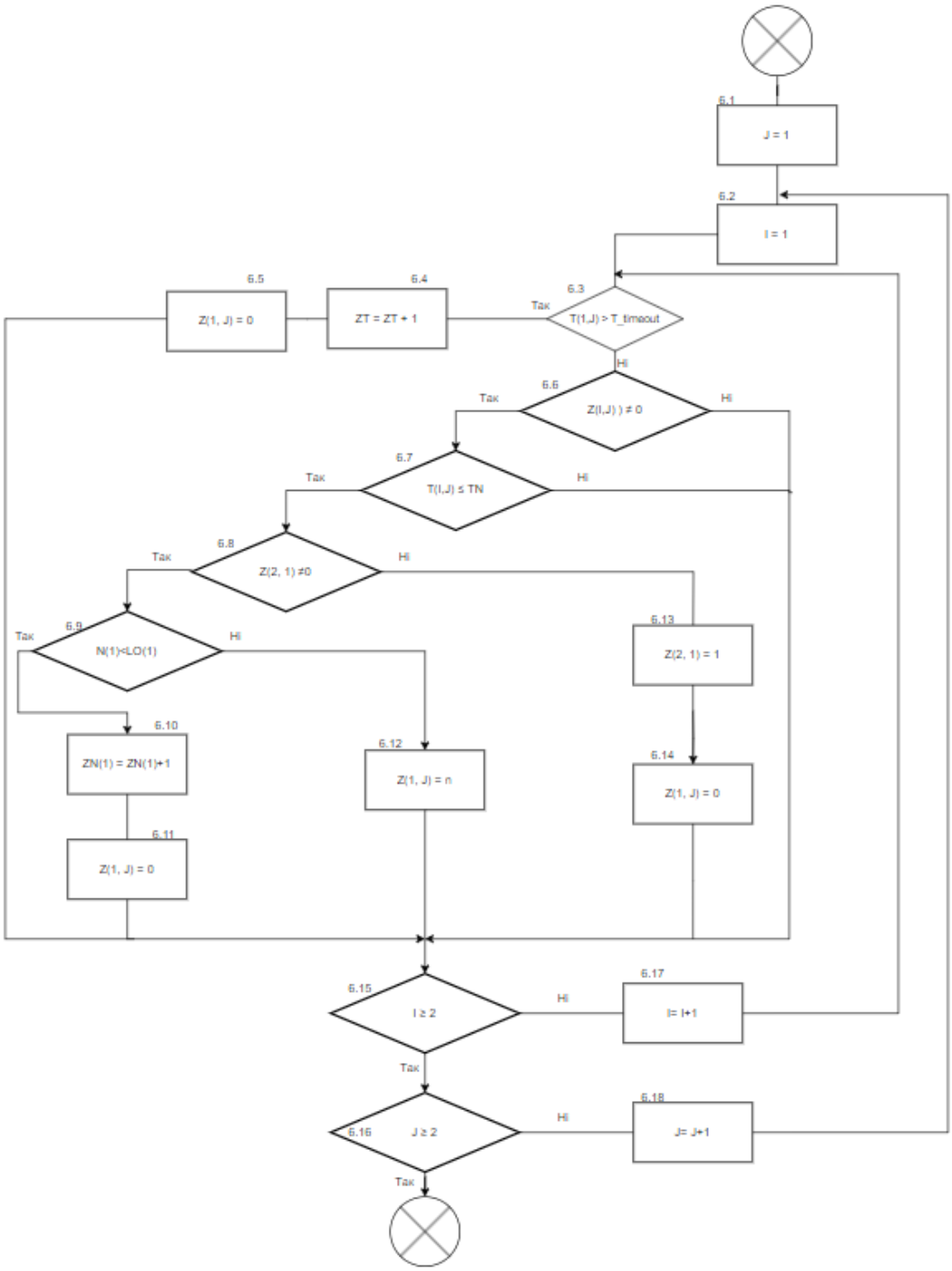

Рисунок 3.9 – Схема алгоритму 6 блоку

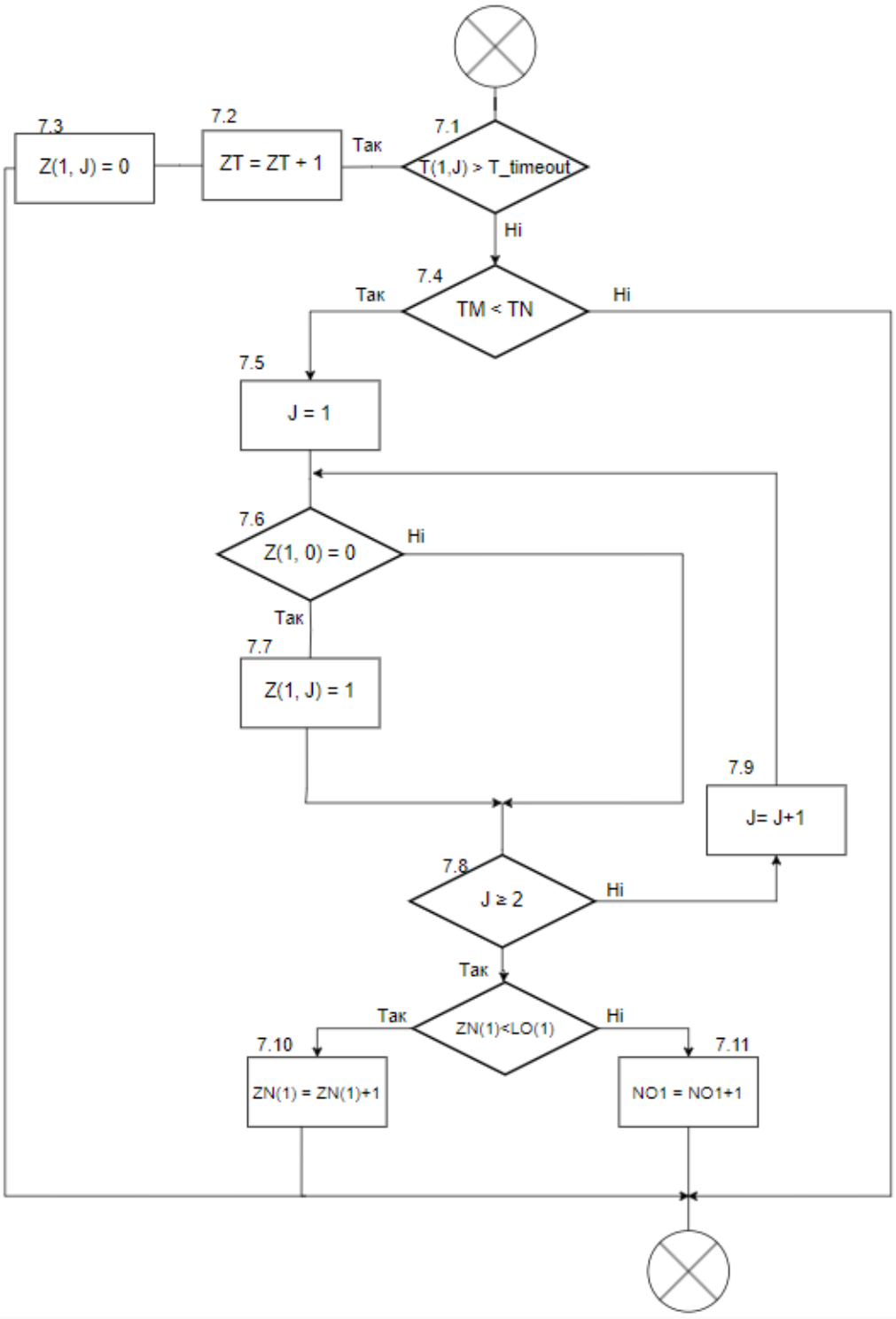

Рисунок 3.10 – Схема алгоритму 7 блоку

4 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАРНА ПЕРЕВІРКА

4.1 Програмна реалізація технології для модифікованої моделі СМО

Структура системи програмного продукту є досить складна, тому увагу в цій роботі буде загострено саме на частині системи, що безпосередньо пов’язана з дослідженням даної роботи. Одним з основних елементів системи є сутність “Item”, ця сутність в системі масового обслуговування має використовуватися як вхідні та вихідні повідомлення. Сутність “Item” може містити наступні дані:

– ідентифікатор сутності;

– стан сутності;

– дані, що зберігає сутність.

Інструкції до виконання певних дій. Система повинна отримати сутність Item від користувача системи, та виконати певні дії що прописані саме в тому, які дані зберігає ця сутність. Цими діями може бути: зміна стану сутності, зміна, заміна або видалення даних що зберігає Item, зміна інструкцій, а також зміна, заміна або видалення самої сутності. Тому заявками системи черги повинна бути саме Item вже у програмній реалізації.

Розроблена цільова програма повинна використовуватися користувачами цієї системі у хмарі. Сутності Item будуть відправлені користувачем до загальної системи та встановленні до черги. Проте ресурси сервера матимуть обмеження за навантаженням. Структура черг в програмному застосунку є невизначеною, як і середнє системне навантаження. Сутності Item самою системою попередньо заплановано не повинні оброблятися, проте в цілях розширення функціоналу, попередня обробка може бути передбачена. Передбачено що сервер буде відкриватися для одного регіону, тому сутності Item будуть оброблені лише одним сервером. Надалі систему можна було б розширити до використання в декількох

регіонах, з відповідно декількома серверами, проте це вже виходить за обсяги визначеного технічного завдання та сфери застосування.

У даній роботі буде представлено запланований вигляд програми та головні частини програмної реалізації представлені у вигляді скріншотів схеми програми та коду алгоритмів роботи програми. Візуальне представлення веб-додатку представлене на рисунку 4.1.

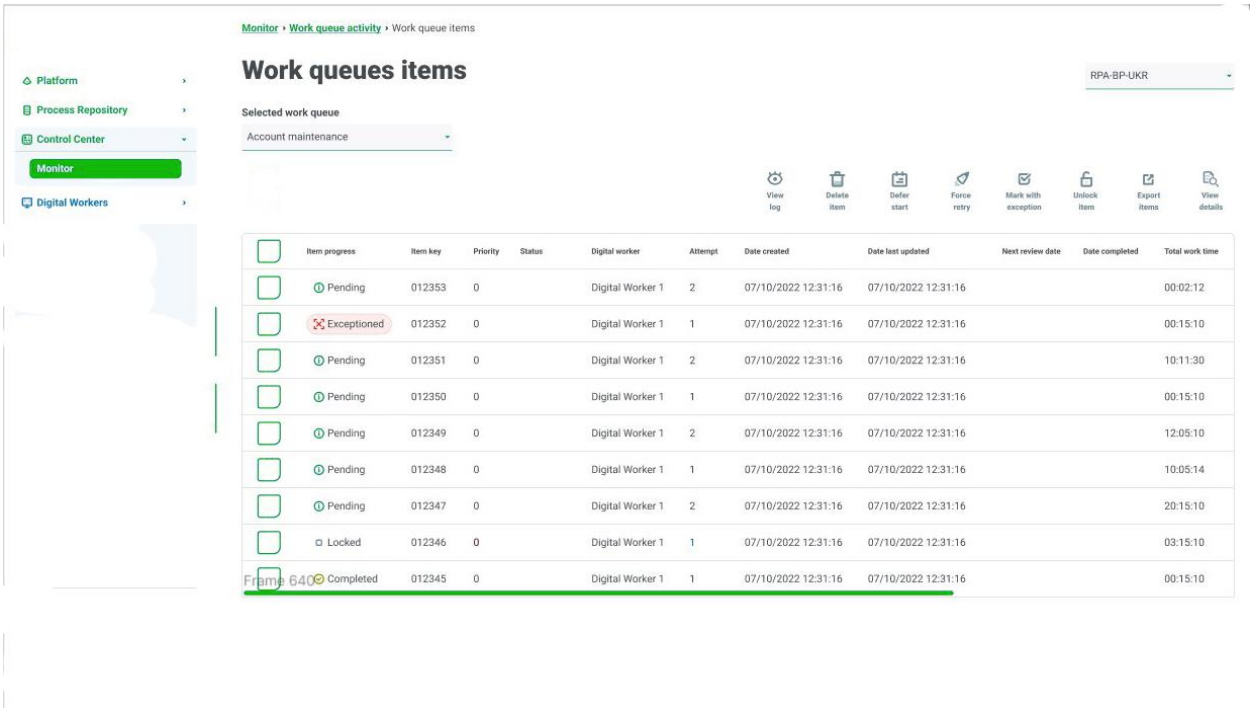

Рисунок 4.1 – Візуальне представлення програми (зображення таблиці items)

Головний елемент системи – item, що представляє собою сутність, яка може бути використана користувачем програми для зберігання певних даних у хмарі або для обробки за допомогою цих даних якоїсь частини коду тощо. Над сутністю системи item можуть бути виконані певні дії, такі як перегляд детальної інформації, видалення, зміна даних, що зберігаються в цій сутності.

Всі ці дії будуть оброблятися сервером (або серверами) суцільної системи для всіх користувачів системи одночасно, тому в даному прикладі потік сутностей item

до системи є заявками системи, передачу яких потрібно оптимізувати. Тому детальніше буде розглянуто алгоритм роботи системи масового обслуговування.

На рисунку 4.2 представлений скріншот алгоритму роботи програми, що відповідає за додання сутності item до черги. На початку кеш 1-го каналу перевіряється на наявність місця для збереження сутності item, а також йде перевірка, чи працює сам кеш. У разі якщо місця в кеші для тимчасового збереження сутності item немає або кеш знаходиться в стані down (recovery), то сутність зберігається в наступному вільному кеші. Якщо немає жодного вільного місця item вважається втраченим, про що й повернеться повідомлення.

На наступному рисунку 4.3 представлений скріншот алгоритму роботи перевірки сутностей item на вихід за максимальний допустимий час очікування

```
private async Task<ResultSaving> WorkItemModel(ItemModel item)
{
    logger.LogInformation("Attempting to save item with Id: {0}", item.Id);
    for (int i = 0; i < _cacheObj.Length; i++)
    {
        if (_cacheObj[i].Capacity < CacheConst.MaxCacheCapacity && _cacheObj[i].IsUp)
        {
            _cacheObj[i].Items.Add(item);
            var result = _cacheObj[i].ResultsOfSaving();
            return result;
        }
    }


    logger.LogInformation("Saving item data to queue failed!");
    return CacheConst.EmptyResult;
}
```

Рисунок 4.2 – Скріншот алгоритму збереження сутності item до тимчасового носія

```
private async Task<bool> OutOfTimeOut(ItemModel item, TimeSpan outOfTimeRange) =>
    DateTime.UtcNow - item.SystemTimeRunning > outOfTimeRange ? true : false;
```

Рисунок 4.3 – Скріншот алгоритму перевірки на перевищення часу очікування

«TimeOut». У разі якщо item перевищує час очікування, він буде видалений з системи та повернений з результатом інформації про перевищення часу очікування.

4.2 Експериментальна перевірка модифікованої моделі побудови СМО

Для експериментальної перевірки необхідно побудувати схему СМО, згенерувати тестові заявки (item) та перевірити результати роботи системи, отримати результати роботи програми та зробити висновки з результатів. Схема СМО побудована основуючись на підрозділі 3.2, для прикладу було реалізовано 4-х канальну версію системи. На схемі зображені усі основні необхідні компоненти для виміру результатів модифікації. У блоці черги було виставлено обмеження на кількість заявок для відповідної черги. У випадку якщо це число перевищує задану норму, тоді заявка відкидається, цей процес фіксується нижнім блоком черги, та відображається відповідно. Якщо час очікування певної заявки перевищує заданий час, то ця заявка потрапляє в заданий блок черги та відкидається. Цей «блок відкидання» знаходиться вище інших блоків черги.

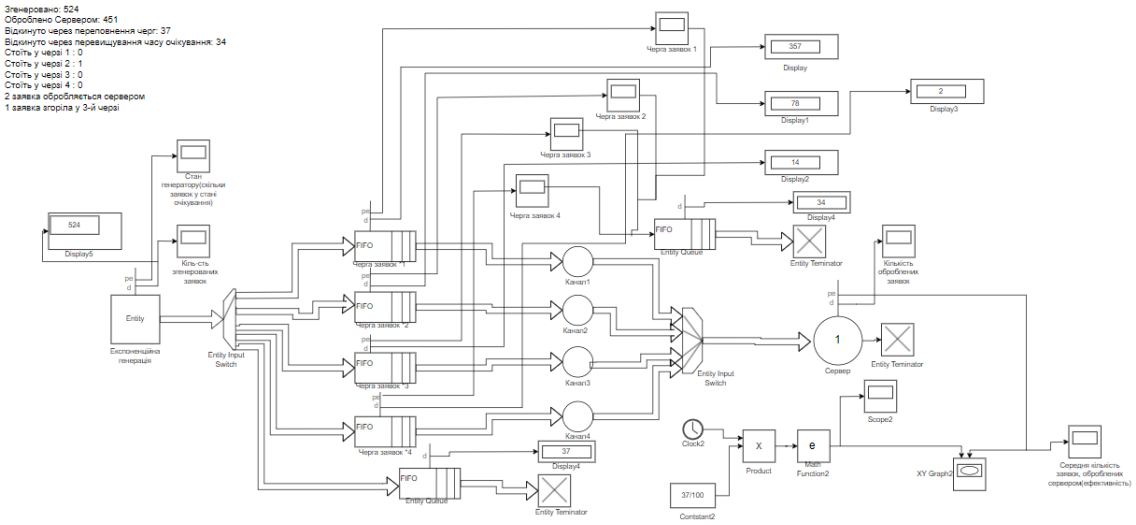

Рисунок 4.4 – Зображення зібраної схеми СМО, а також її обрахунки

На наступному графіку (рисунок 4.5) зображено процес генерації заявок, де n–позначає кількість згенерованих заявок, а tчас в секундах.

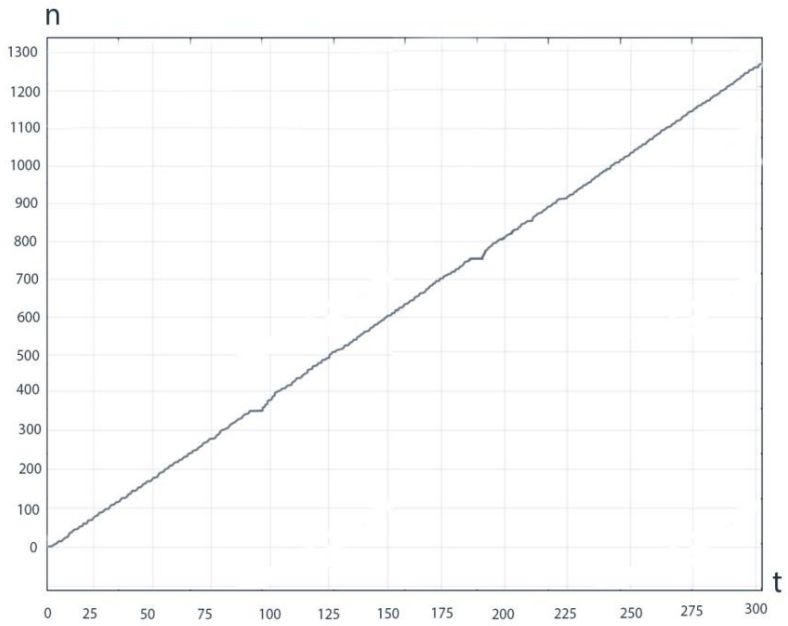

Рисунок 4.5 – Графік генерації заявок

Всього було згенеровано 1257 заявки за 5 хвилин (300 секунд), з яких 1236 були оброблені сервером. З цього виходить нестача 21 заявок, з яких: 2 відкинуто через переповнення черги, 14 відкинуто через перевищення часу очікування, 4 заявка залишилася в черзі, 1 заявки обробляються сервером. Уже з цих значень можна зробити висновки про успішно створену СМО, з низькою часткою втрачених заявок взагалі (16 заявок). Можна виключити заявки, що знаходяться в обробці, через особливість генерації ці заявки не були відкинуті а лише не встигли обробитися сервером. Надалі отримаємо графіки надійності та точності. Значення ефективності обозначатиме завантаженість серверу, а значення надійності ймовірність безвідмовної роботи. Отримані графіки зображені на рисунках 4.8 та 4.9. Показником ефективності становитиме завантаженість серверу, а показником надійності встановимо ймовірність безвідмовної роботи. Цей показник можна розрахувати за формулою

$$P = e^{-kt}, \tag{4.1}$$

де k – є інтенсивністю відмов;

t – це момент часу, тобто вісь абсцис.

Інтенсивність в свою чергу розраховується за такою формулою:

$$k = \frac{n}{T}, \tag{4.2}$$

де T – є часом моделювання;

n – кількість відмов за цей час.

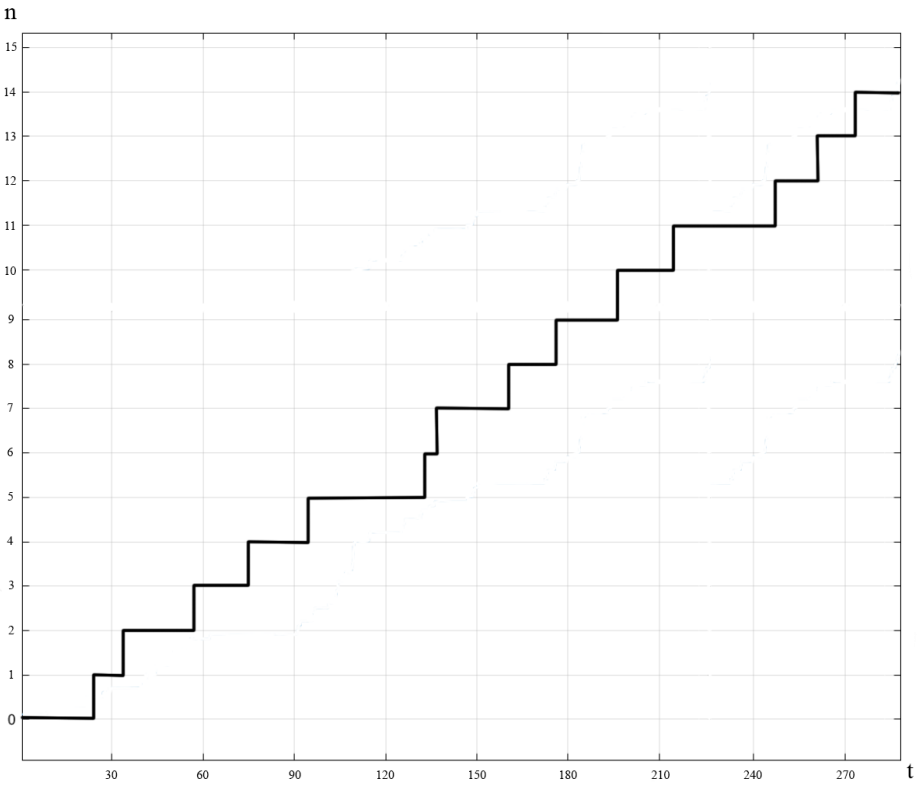

Рисунок 4.6 – Графік відкинутих заявок через перевищування часу очікування

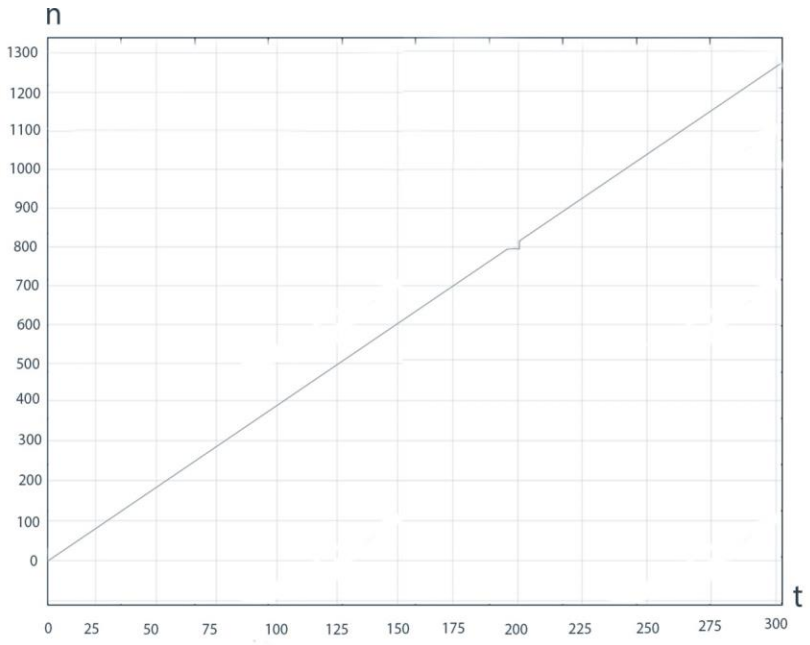

Рисунок 4.7 – Графік кількості оброблених заявок

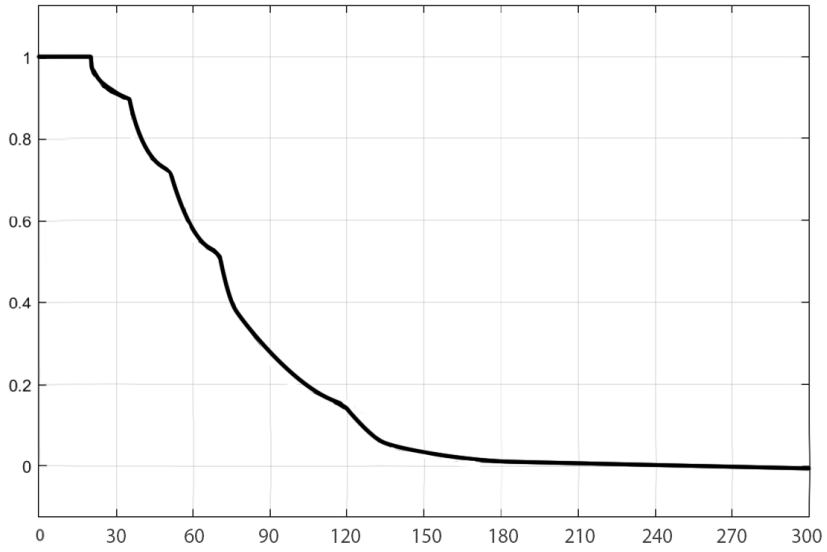

Рисунок 4.8 – Графік надійності системи

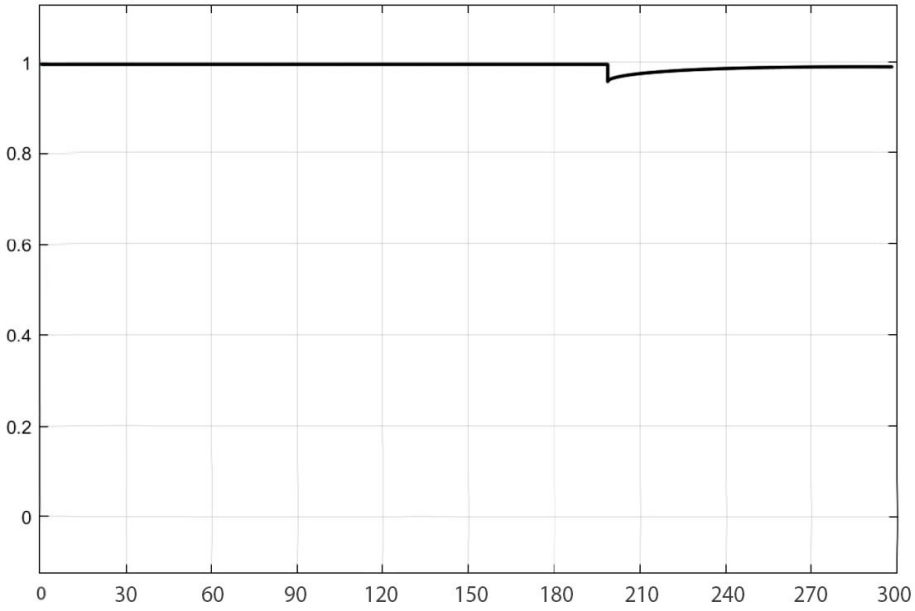

Рисунок 4.9 – Графік ефективності системи

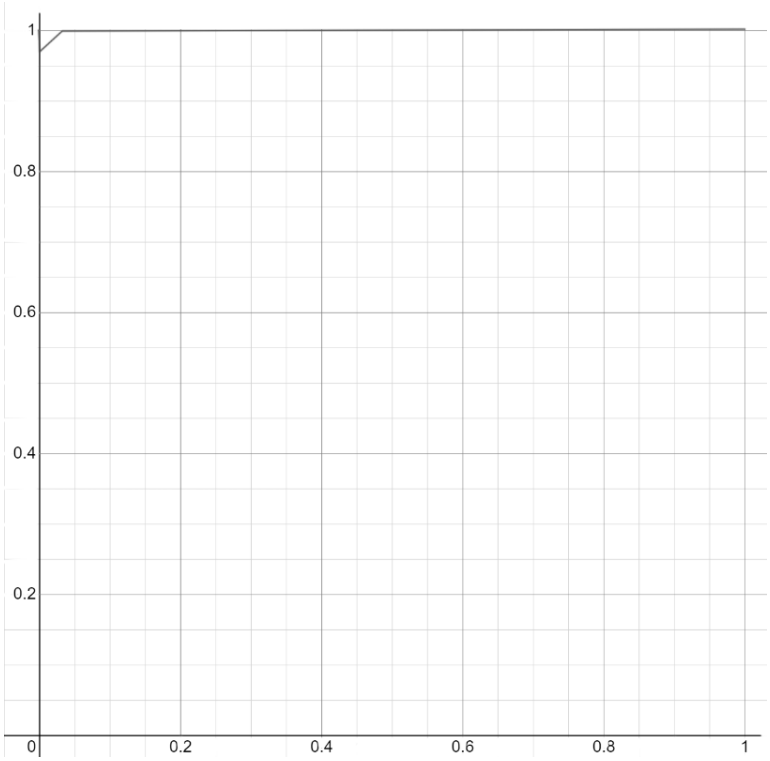

Рисунок 4.10 – Графік ефективності-надійності (вісь абсцис є значеннями ефективності, ординат – надійністю)

Отримані графіки є гарним результатом - ефективність даної СМО наближується до значення одиниці, кількість згенерованих заявок до відкинутих системою в відсотковому відношенні 98,72%, що наближується до одиниці, що теж є гарним результатом. Причинністю до просідання на графіку ефективності має просте пояснення – через особливу генерацію заявок на моменті часу між 175-ю та 200-ю секундами роботи генерації була відсутність нових заявок, через що як тільки сервер обробив заявки що знаходилися у накопичувачах, він почав простоювати, всього на декілька секунд але це мало свій вплив на графіку. Графік надійності має незвичну форму через налічування невеликої кількості відкинутих заявок з великими проміжками часу. Можна сказати що графік надійності досить довгий час утримував надійність близько одиниці та не наближався до нуля. Графік надійності свідчить про достатній рівень для поставленої задачі, враховуючи систему що

передбачає модифікацію збереження заявки за можливістю у разі поломки, можна сказати що побудована система є досить надійною.

Графік надійності-ефективності має теж гарний результат майже на протязі всієї вісі абсцис ефективність наближується або є одиницею, проте на початку графіку є просідання що пов’язане з відсутністю заявок для сервера, і саме його простоюванням про що вже було згадано. Можна зробити висновок що системи буде здатна пройти випробування на реальних даних з гарним результатом.

Порівнюючи результати у відсотковому відношенні відмов до загальної кількості оброблених або згенерованих заявок для систем масового обслуговування без модифікацій (він становить приблизно 5-10%) можна сказати що реалізована модифікована система масового обслуговування відповідає всім поставленим критеріям та займає не останнє місце серед інших модифікованих СМО.

ВИСНОВКИ

У рамках дослідження було розроблено нові модифікації моделей СМО з урахуванням обмежень для заявок на час очікування та можливого виходу з ладу певних елементів системи. Ці модифікації дозволяють значно зменшити кількість відмов заявок, що є важливим для систем масового обслуговування в хмарних обчисленнях.

У першому розділі роботи було визначено задачу. У другому розділі була проведена розробка вдосконаленої моделі черг. У третьому розділі відбувалось моделювання процесів функціонування систем модифікованої та Пуассонівської моделі. У четвертому розділі було проведено апробацію модифікованої моделі на тестових згенерованих даних та почато виконання програмної реалізації.

Таким чином, результати дослідження дозволяють зробити наступні висновки:

1) модифікація СМО шляхом встановлення обмежень для заявок на час очікування та врахуванням можливого виходу з ладу певних елементів системи є ефективною для зменшення відмов заявок;

2) модифікована модель СМО з обмеженнями для заявок на час очікування та врахуванням можливого виходу з ладу певних елементів системи займає не останнє місце серед інших модифікованих СМО з відмовами.

Найважливішою науковою новизною роботи є отримання досить успішного результату з приблизно 2% відмов заявок, завдяки розробленим модифікаціям. Ці модифікації дозволили значно зменшити кількість відмов заявок та можливого виходу з ладу певних елементів системи, що є важливим для систем масового обслуговування в хмарних обчисленнях.

Апробація наукового дослідження була виконана у вигляді тез [22].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Bondaroun T., Fisher S. Encyclopedia of Electronic HRM. Walterde Gruyter GmbH, Berlin/Boston, 2020. 323 p.
2. Shortle J.F., Thompson J.M., Gross D., Harris C. Fundamentals of Queueing Theory, 5th Edition, Boston, 2018. 213 p.
3. What Is Candidate Relationship Management? Definition, Process, Solutions, and Best Practices URL: https://less.worns/less/principles/ queueing_theory (дата звернення: 17.11.2023).
4. Queuing Theory Definition, Elements, and Example URL: https://www.investopedia.com/terms/q/queuing-theory.asp (дата звернення: 21.11.2023).
5. Stewart W.J. Probability, Markov Chains, Queues, and Simulation. The Mathematical Basis of Performance Modeling. S. R. Nova PvtLtd, Bangalore, India, 2008. 111 p.
6. Introduction to Queueing Theory, 2th Edition URL: https://www.cse.fau.edu/~bob/publications/IntroToQueueingTheory_Cooper.pdf (дата звернення: 22.11.2023).
7. Тихомиров А.А. Теорія масового обслуговування. Київ, 2004. 78 с.
8. Міхнова А.В., Міхнова О.Д. Особливості візуалізації даних у системах діагностики. Системи обробки інформації, 2 (100). Харків: ХУПС, 2012. С. 67-70.
9. Кендал Т.К., Мейер Г.Л. Системи масового обслуговування. Теорія та приклади. Harris: Fundamentals of Queueing Theory, 2004. 210 с.
10. Гніденко В.І. Удосконалення методів розрахунку відсотка відмов систем масового обслуговування. Київ: Київський національний університет імені Тараса Шевченка, 2016. 421 с.

11. El-Gohary M., Aly M. Modeling and Analysis of Cloud Computing Queues. ACM Computing Surveys, 2017. 230 p.

12. Bailey N.T.J. Some further results in the non-equilibrium theory of a simple queue. J. R. Stat. Soc. B19, 1994. 333 p.

13. Benson T., Akella A., Maltz D.A. Network traffic characteristics of data centers in the wild. In: Proceedings of the 10th Annual Conference on Internet Measurement. IMC ’10, New-York, NY, USA. ACM, 2010. 280 p.

14. Grassmann W.K. Transient and steady state results for two parallel queues. Omega 8, 1980. 112 p.

15. Hampshire, R.C., Harchol-Balter M., Massey W.A. Fluid and diffusion limits for transient sojourn time processor sharing queues with time varying rates. Queueing Syst. 53(2), 2006. 30 p.

16. Loboz, C. Cloud resource usage — heavy tailed distributions in validating traditional capacity planning models. Grid Computing, 10(1), 2012. P. 85-108 p.

17. Newell, G.F. Applications of Queueing Theory. Chapman & Hall, London, 1971. 115 p.

18. Rothkopf, M.H., Oren, S.S. A closure approximation for the non-stationary M/M/s queue. Manag. Sci. 25, 1979. 534 p.

19. Jinesh V. Architecting for the Cloud: Best Practices, 2010. Queueing Syst, 2006. 96 p.

20. Mikhnova O. Key frame extraction from video: framework and advances. International Journal of Computer Vision and Image Processing. Vol. 4, No. 2, 2014. P. 67-78.

21. Савчук О. В., Моргаль О.М., Моделювання процесів і систем. Київ, 2021. 220 с.

22. Mikhnova O., Obushko D. Queueing Models in Cloud Computing. Інформаційні системи та технології: матеріали 12-ї Міжнародної науково-технічної конференції. Частина 2. Молодіжна секція, Харків, 2023. С. 15-16.

23. Методичні вказівки щодо розробки та оформлення кваліфікаційної роботи (для студентів усіх форм навчання другого (магістерського) рівня програми "Інформаційні управляючі системи та технології") / Упоряд.: Петров К.Е., Левикін В.М., Чалий С.Ф., Євланов М.В., Саєнко В.І., Міхнов Д.К., Міхнова А.В., Чала О.В. – Харків: ХНУРЕ, 2021. – 30с.

24. ДСТУ 3008-2015 «Звіти у сфері науки і техніки» структура та правила оформлення. ДП «УкрНДНЦ», Київ, 2016. 31 с.

25. ДСТУ 8302-2015 «Бібліографічне посилання» загальні положення та правила складання. ДП «УкрНДНЦ», Київ, 2016. 20 с.