Центральноукраїнський національний технічний університет
Центр заочної та дистанційної освіти
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2026 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за другим (магістерським) рівнем вищої освіти
на тему
“Дослідження та програмна реалізація системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії”

Виконав здобувач вищої освіти
ІІ курсу, групи КН-25МЗ
ОПП «Комп’ютерні науки»
спеціальності F3 «Комп’ютерні науки»
_____ Міносьян Р.І.
«____» _____ 2026 р.

Керівник проекту
кандидат технічних наук, доцент
_____ Смірнова Т.В.
«____» _____ 2026 р.

Нормоконтроль проекту
_____ Коваленко А.С.
«____» _____ 2026 р.

Рецензент _____
_____

Підписано КЕП
Центральноукраїнський національний технічний університет
Міносьян Ростислав Ігорович
Сертифікат: 3C080BB109AB2598040000000A3603A005121F700
Дійсний з 06.10.2026 13:21:31 по 06.10.2028 13:21:31

Центральноукраїнський національний технічний університет
Центр *Заочної та дистанційної освіти*
Кафедра *Кібербезпеки та програмного забезпечення*
Рівень вищої освіти *магістр*
Галузь знань _F_____*"Інформаційні технології"*_____
Спеціальність F3 *"Комп'ютерні науки"*
Освітньо-професійна (освітньо-наукова) програма _*"Комп'ютерні науки"*_

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 1 » вересня 2026 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ДРУГИМ (МАГІСТЕРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Міносьяну Ростиславу Ігоровичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження та програмна реалізація системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії*

2. Керівник роботи *Смірнова Тетяна Віталіївна, кандидат технічних наук, доцент*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 44-13 від 17.08.2026 року

3. Строк подання студентом роботи до захисту *2.12.2026 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою розробки є дослідження та програмна реалізація системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії*

5. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Призначення та область використання.	*5. Впровадження системи в промислову експлуатацію*
2. Перегляд аналогічних існуючих систем.	*6. Наукова новизна.*
3. Опис і обґрунтування проектних рішень.	*7. Заходи з охорони праці та техніки безпеки.*
4. Етапи програмування системи.	*8. Висновки.*

6. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Наукова новизна	*1 аркуш*
Структурна схема системи	*1 аркуш*
Функціональна схема системи	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*

7. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Марченко К.М., к.т.н., доцент	06.10.2026	16.11.2026

8. Дата видачі завдання « 1 » вересня 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.10.2026 р.	
2.	Постановка задачі, оформлення ТЗ	15.10.2026 р.	
3.	Розробка моделі компонента	20.10.2026 р.	
4.	Розробка структур даних	25.10.2026 р.	
5.	Розробка алгоритмів системи	30.10.2026 р.	
6.	Програмування алгоритмів	10.11.2026 р.	
7.	Розрахунки з охорони праці та техніки безпеки	15.11.2026 р.	
8.	Оформлення ПЗ	17.11.2026 р.	
9.	Попередній захист роботи	2.12.2026 р.	

Дата видачі завдання
« 1 » вересня 2026 р.

Підпис керівника

_____ _____
(прізвище та ініціали)

Завдання прийнято до виконання
« 1 » вересня 2026 р.

Підпис здобувача

_____ _____
(прізвище та ініціали)

АНОТАЦІЯ

Міносьян Р.І. Дослідження та програмна реалізація системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії. ФЗ Комп'ютерні науки. Центральноукраїнський національний технічний університет. Кропивницький. 2026.

В даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Метою розробки є дослідження та програмна реалізація системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Об'єктом дослідження є процес ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Предметом дослідження є методи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Методи дослідження базуються на методах обробки та зберігання інформації, методах оптимізації, методах математичної статистики, методах розробки програмного забезпечення.

Результат роботи – програмна реалізація системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на ПЕОМ з ОС Windows 10/11.

Програму розроблено в середовищі Python.

Ключові слова: комп'ютерні науки, ієрархічне зберігання даних, моделі багатокритеріальної оптимізації

ABSTRACT

Minosian R.I. Research and Software Implementation of a Hierarchical Data Storage System Based on Multi-Criteria Optimization Models for Cost and Performance. F3 Computer Science. Central Ukrainian National Technical University. Kropyvnytskyi. 2026.

This final qualification thesis for the second (master's) level of higher education presents the development of software designed for a hierarchical data storage system based on multi-criteria optimization models for cost and performance.

The objective of this development is the research and software implementation of a hierarchical data storage system based on multi-criteria optimization models for cost and performance.

The object of the research is the process of hierarchical data storage based on multi-criteria optimization models for cost and performance.

The subject of the research consists of methods for hierarchical data storage based on multi-criteria optimization models for cost and performance.

The research methodology draws upon techniques for information processing and storage, optimization methods, mathematical statistics, and software development practices.

The outcome of the work is a software implementation of a hierarchical data storage system based on multi-criteria optimization models for cost and performance.

During the development of the software model, an analysis of existing hardware and software solutions was conducted. All components of the developed software are described in detail.

A user-friendly interface has been developed, and instructions for operating the software are provided. The program can be used on PCs running Windows 10/11.

The program was developed in a Python environment.

Keywords: computer science, hierarchical data storage, multi-criteria optimization models

ВІДОМІСТЬ ПРОЕКТУ

		Текстові документи			
1	А4	Пояснювальна записка	110	–	
2	А4	Робоча програма	25	–	Додаток А
		Графічні документи			
3	А4	Наукова новизна	1	–	
4	А4	Структурна схема	1	–	
5	А4	Функціональна схема роботи системи	1	–	
6	А4	Діаграма взаємодії процесів	1	–	
7	А4	Блок-схема основної програми	1	–	
8	А4	Блок-схема роботи підпрограми	1	–	

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ 9
ВСТУП..... 10
1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ 12
1.1 Призначення системи..... 12
1.2 Область застосування 12
2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ 14
2.1 Огляд існуючих систем, технологій, архітектур та програмних рішень за профілем теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти..... 14
2.2 Обґрунтування вибору засобів для побудови системи та мови програмування 35
2.3 Розгорнута постановка завдання 36
3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ 38
3.1 Опис функціонування системи 38
3.2 Розробка структурної схеми..... 40
3.3 Розробка функціональної схеми 51
3.4 Розробка діаграми процесів 54
4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ 56
4.1 Розробка блок-схем та опис алгоритмів функціонування системи 56
5 ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ..... 84
6 НАУКОВА НОВИЗНА 89

7 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ 90
7.1 Вступ 90
7.2 Шкідливі і небезпечні фактори при роботі з комп’ютером 91
7.3 Аналіз санітарно-гігієнічних умов праці на робочому місці програміста ... 93
7.4 Розробка заходів з умов поліпшення охорони праці 96
7.5 Розрахункова частина 97
8 ОСНОВНІ ВИСНОВКИ 100
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 102

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

LZW – Алгоритм Зіва-Лемпела

ВСТУП

Актуальність теми. Багаторівневе або іншими словами ієрархічне сховище розподіляє дані за різними класами сховищ на основі їхньої цінності, використання та вимог до доступу. Щоб зрозуміти, чому це важливо, подумайте про три ключові моменти телеметрії: обсяг, різноманітність та цінність.

Дані дуже різняться за типом, форматом та структурою, і до них слід ставитися відповідно. Це не лише питання безпеки чи управління. Це практична основа для вашої стратегії роботи з даними цього року та надалі.

Цінність даних також різна. Для сповіщень у режимі реального часу їхня цінність може зникнути за лічені хвилини. Для дотримання вимог та розслідування інцидентів вони можуть стати цінними лише через місяці чи роки. Головне – визначити типи даних та їхню потенційну цінність, а потім зіставити кожен з них з правильним класом сховища для ваших IT-команд та команд безпеки.

Мета й завдання дослідження. Метою роботи є дослідження та програмна реалізація системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Огляд існуючих систем ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

– Дослідження системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

– Програмна реалізація системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Об'єктом дослідження є процес ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Предметом дослідження є методи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Методи дослідження базуються на методах обробки та зберігання інформації, методах оптимізації, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

– Розроблено вітчизняний продукт ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії, який має більш широкі можливості, на відміну від існуючих аналогів.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Достовірність наукових результатів підтверджена теоретичними викладеннями, даними комп'ютерного моделювання, коректними дослідженнями параметрів на функціонуючій обчислювальній мережі, а також відповідністю отриманих результатів окремим результатам, наведеним у науковій літературі.

Робота апробована на LVIII науково-технічній конференції здобувачів вищої освіти «Наука-виробництву», основні положення випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти надруковані у статті збірника праць молодих науковців ЦНТУ, випуск №17.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Цінність даних залежить від того, кому вони потрібні і коли. Якщо ви працюєте в команді SecOps, вам потрібні дані в режимі реального часу, щоб попередити вас про проблеми. Дані, яким кілька годин, марні, коли ви реагуєте на активну подію. Якщо ви працюєте в ITOps, години, дні, місяці чи роки історії можуть бути саме тим, що вам потрібно для аналізу тенденцій або дотримання вимог.

Цінність даних залежить від віку, доступності, обсягу та того, хто з ними взаємодіє. Деякі дані в режимі реального часу з невеликим обсягом, на які ніхто не звертає уваги сьогодні, можуть стати критично важливими доказами через місяць.

І масштаб цієї проблеми продовжує зростати. Згідно з аналізом галузевих даних, обсяг телеметричних даних збільшується приблизно на 29% на рік, що фактично означає, що вони подвоюються кожні 18 місяців. Бюджети не подвоюються кожні 18 місяців. Саме цей розрив є основною причиною існування багаторівневого сховища, і саме тому багаторівневе сховище даних стало критично важливим для команд безпеки та спостереження.

1.2 Область застосування

Багаторівневе управління даними балансує між витратами та складністю. Оскільки організації борються зі зростанням обсягів IT-даних та даних безпеки, багато хто з них запровадив стратегії, що переміщують дані у сховище на основі своєчасності: операційні випадки використання в одному місці, дослідницька робота в іншому, з урахуванням нормативних вимог та збереження даних у процесі. Цей підхід поєднує застарілі інструменти, хмарні сховища даних,

Lakehouse та озера даних для зберігання та аналізу даних відповідно до їх цінності, моделей використання та вимог до зберігання. Не існує єдиної відповіді, яка підходить для кожного випадку.

Механізм обробки даних забезпечує автоматизоване, гнучке розподілення даних на рівні, тому місце зберігання даних відповідає тому, як вони цінуються та використовуються. Дані, оптимізовані для продуктивності, надходять до ваших інструментів аналітики, моніторингу та кібербезпеки у формі, яку очікує кожна платформа.

Підтримка повноцінних наборів даних важлива так само, як і оптимізоване за витратами сховище. Повноцінні дані часто потрібні для дослідницької роботи та забезпечення відповідності вимогам, а багаторівневе розміщення розміщує їх у правильному місці для цільового використання. У поєднанні з розподіленим доступом та управлінням, вбудованим у механізм обробки даних, кожен рівень залишається доступним незалежно від того, де знаходяться дані.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти

Spectra StorCycle

Програмне забезпечення для керування життєвим циклом сховища, яке автоматично переносить дані на нижчі рівні, включаючи стрічкові, об'єктні сховища та хмарні.

Механізм політик переміщує або копіює файли між NAS, хмарним сховищем об'єктів, диском, стрічкою та цільовими пристроями Spectra BlackPearl.

StorCycle застосовує правила джерела, місця призначення, розкладу та дій до повторюваних передач файлів. Адміністратори можуть копіювати дані, перевіряти передачі та видаляти вихідні файли після успішного завершення. Журнали завдань, операції відновлення та доступ до REST API забезпечують контроль над керованими робочими процесами архівування.

StorCycle не забезпечує багаторівневе розміщення на рівні блоків, тому середовищам, що потребують розміщення на підтомах, потрібен інший продукт. Команда постпродакшну може переміщувати завершені проекти з NAS до BlackPearl або хмарного сховища, зберігаючи при цьому робочий шлях відновлення.

Переваги:

– Вбудована інтеграція BlackPearl підтримує робочі процеси архівування на стрічках.

– Копіює або видаляє файли після перевірки передачі.

– REST API підтримує автоматизовані операції з політиками та завданнями.

– Зберігає метадані файлів під час передачі архівів.

Недоліки:

– Зосереджений на переміщенні файлів, а не на багаторівневому розподілі на рівні блоків.

– Швидкість відкликання залежить від затримки стрічки або хмарного цільового об'єкта.

– Розробка політик вимагає ретельних правил щодо джерела, місця призначення та видалення.

– Не забезпечує координацію транзакцій бази даних для даних програми.

Використовуйте сценарії:

– Команди медіа-продакшену.

Архів завершених проектів з NAS:

→ Повернено ємність основного NAS.

– Науково-дослідні установи.

Переміщення неактивних наборів даних до сховища об’єктів:

→ Менше навантаження на активне сховище.

Найкраще підходить для: коли командам зберігання даних потрібна міграція файлів з урахуванням правил між NAS, хмарою, стрічкою та BlackPearl.

Data Dynamics StorageX

Програмне забезпечення для керування неструктурованими даними, яке аналізує дані та автоматизує їх переміщення між рівнями сховища на основі політик та метаданих.

StorageX Analytics пов'язує аналіз шаблонів доступу з автоматизованими політиками розміщення файлів.

StorageX відображає неструктуровані файли в NAS-системах, файлових серверах та хмарних цільових пристроях, а потім представляє дані інвентаризації для прийняття рішень щодо ємності та власності. Міграція на основі політик може спрямовувати неактивні файли до вторинного сховища, зберігаючи при цьому знайомі шляхи через посилання на файли, які запускають відкликання. Аналітика на рівні файлів надає адміністраторам дані щодо політик

зберігання, розміщення та ємності.

Розгортання вимагає ретельного розроблення політик для спільних ресурсів з різними власниками, винятків та вимог до відкликання. StorageX підходить для центрів обробки даних, які консолідують застарілі потужності NAS або переміщують неактивні файли проектів до об'єктного сховища. Дизайн, орієнтований на файли, не спрямований на розміщення в блочному сховищі або елементи керування життєвим циклом бази даних, властиві додаткам.

Переваги:

– Аналітика на рівні файлів виявляє вік, розмір, власника та моделі доступу.

– Міграція на основі політик автоматизує розміщення даних між NAS та хмарними цільовими сховищами.

– Прозорі заглушки зберігають шляхи після переміщення відповідних файлів.

– Централізовані інформаційні панелі підтримують перевірку потужностей та відповідальності.

Недоліки:

– Розробка політик стає складною для багатьох правил спільного використання та винятків.

– Робочі навантаження блочного сховища виходять за межі його файлоорієнтованої операційної моделі.

– Продуктивність відкликання залежить від затримки цільового об'єкта та пропускної здатності мережі.

– Глибокі робочі процеси зберігання з урахуванням додатків можуть вимагати зовнішнього контролю.

Використовуйте сценарії:

– Адміністратори корпоративних сховищ.

Міграція застарілих спільних ресурсів NAS до об'єктного сховища:

→ Менше споживання основного накопичувача.

– Менеджери дослідницьких даних.

Керування файлами проектів для кількох сайтів:

Найкраще підходить для: коли корпоративним командам потрібна міграція з урахуванням політик між розподіленими сховищами NAS та хмарними сховищами.

Hammerspace

Глобальне програмне забезпечення для оркестрації даних, яке автоматизує розміщення файлів на рівнях флеш-сховища, об'єктів та хмари.

Глобальне середовище даних об'єднує розподілені NAS, паралельні файлові системи, сховище об'єктів та хмарні ресурси в одному просторі імен.

Глобальне середовище даних представляє існуючі системи зберігання даних через єдиний простір імен та координує переміщення без необхідності переписування програм. Hammerspace поєднує елементи керування політиками, адміністративні API та розміщення на рівні сайту для керування даними в географічно розподілених середовищах. Його підхід на рівні файлів підходить організаціям, яким потрібен спільний доступ до гетерогенного сховища, а не до єдиного архівного цільового об'єкта.

Архітектура вимагає ретельного планування метаданих, простору імен та мережі перед широким розгортанням. Розподілені команди з навчання ШІ можуть використовувати Hammerspace для розміщення активних наборів даних поблизу обчислювальних ресурсів, зберігаючи при цьому холодніші копії на дешевших сховищах.

Переваги:

– Глобальний простір імен охоплює NAS, паралельні файлові системи, сховища об'єктів та хмарні цільові системи.

– Автоматизація політик розміщує дані в гетерогенних середовищах зберігання.

– Програми можуть зберігати шляхи під час переміщення на серверній частині.

– Контроль локальності даних підтримує розподілений штучний інтелект та дослідницькі робочі процеси.

Недоліки:

– Розгортання вимагає ретельного планування метаданих, простору імен та мережі сайтів.

– Hammerspace не забезпечує рівнів на рівні блоків.

– Поведінка під час роботи залежить від роз'ємів та можливостей цільової системи.

– Тестування політик необхідне для контролю трафіку відкликання між віддаленими сайтами.

Використовуйте сценарії:

– Команди з інфраструктури штучного інтелекту.

Розміщуйте набори даних поблизу обчислювальних кластерів:

→ Коротші шляхи доступу до даних.

– Науково-дослідні установи..

Координація файлів досліджень з кількох центрів

→ Постійний міжсайтовий доступ.

Найкраще підходить для: коли підприємствам потрібне розміщення файлів на основі політик у розподілених NAS, об'єктних, хмарних та високопродуктивних обчислювальних середовищах.

FUJIFILM Object Archive

Програмне забезпечення для активного архівування, яке керує розміщенням даних між диском, об'єктом, стрічкою та хмарою для довгострокового зберігання даних.

Оркестрація відкликання Fujifilm пов’язана з регульованою політикою зберігання архівних активів.

FUJIFILM Object Archive – це пропозиція керування сховищем, побудована на основі робочого процесу архівування на основі об’єктів для організацій, які переміщують великі медіа- та файлові активи на довгострокове сховище. Основні можливості зосереджені на розміщенні архівів на основі політик, автоматизованих робочих процесах відкликання та контролі збереження для регульованого управління життєвим циклом.

Рішення також спрямоване на операційну видимість за допомогою моніторингу подій прийому та відкликання, що важливо для управління затримкою відкликання та очікуваною пропускною здатністю. Глибина інтеграції забезпечується переважно інтерфейсами екосистеми Fujifilm та перехоплювачами операційної автоматизації, а не узагальненими низькорівневими API.

Переваги:

– Розміщення архівів, кероване політиками, що підходить для регульованих життєвих циклів медіа.

– Робочий процес відкликання, розроблений для операційного відстеження під час прийому та вилучення даних.

– Контрольні засоби збереження підтримують узгоджені межі відповідності для архівних активів.

– Об'єктно-орієнтоване керування зменшує тертя під час роботи з серверних частин сховища об'єктів.

Недоліки:

– Інструменти адміністрування зосереджені на робочих процесах Fujifilm більше, ніж на узагальненій багаторівневій інженерії.

– Глибина автоматизації поза межами підтримуваних інтерфейсів може бути обмеженою.

– Детальні елементи керування для відкликань не надаються універсальним чином.

– Масова міграція та прозоре переміщення даних залежать від шляхів оркестрації Fujifilm.

Найкраще підходить для: коли організаціям потрібні робочі процеси архівування керованих об'єктів для колекцій з великим обсягом медіафайлів та передбачуваними операціями відкликання.

Quantum StorNext

Спільна файлова система та програмне забезпечення для керування даними з розподілом на рівні на основі політик для NVMe, дискового сховища,

об'єктного сховища, хмарного сховища та стрічкового сховища.

Механізм політик StorNext пов'язує розміщення на рівнях з поведінкою відкликання та відновлення простору, тому робочі процеси залишаються узгодженими на всіх носіях інформації.

Quantum StorNext використовує ієрархічні політики зберігання для робочих циклів життєвого циклу файлів і даних на дисках, стрічках і в хмарних цільових пристроях, а відкликання та міграція контролюються механізмом політик. StorNext зосереджується на інтеграції файлової системи та оркестрації переміщення даних, щоб видимі для програм шляхи та поведінка відкликання залишалися незмінними, навіть коли рівні ємності змінюються.

Quantum StorNext також надає адміністративні засоби для управління, аудиту та автоматизації дій зі зберігання, пов'язаних з подіями життєвого циклу. Порівняно з іншими стеками рівнів, глибина його робочого процесу досягається завдяки поєднанню контролю розміщення з поведінкою відкликання та відновлення простору для виробничих файлових систем.

Переваги:

– Поведінка відкликання, керована політиками, інтегрується з робочими процесами виробничих файлів.

– Рішення щодо розміщення рівнів залишаються централізованими в механізмі політики зберігання даних.

– Інтеграція стрічкової бібліотеки підтримує робочі процеси з автономними медіа.

– Хуки автоматизації підтримують повторювані дії життєвого циклу без ручного виконання кроків.

Недоліки:

– Розгортання вимагає ретельного налаштування цільових об'єктів сховища та дозволів.

– Для керування обмеженням відкликання та затримкою необхідне налаштування операційної системи.

– Багаторівневі конструкції можуть ускладнювати усунення

несправностей на етапах відкликання та міграції.

– Вища складність, ніж багаторівневе використання лише об'єктів, коли робочі процеси охоплюють багато серверних частин.

Найкраще підходить для: коли підприємствам потрібне відкликання та міграція файлових навантажень на диск, стрічку та в хмару, що регулюються політиками.

Hitachi Content Platform

Платформа об'єктного сховища з вбудованим керуванням життєвим циклом даних, яка автоматизує розподіл на рівні між класами сховищ.

Централізоване упорядкування політик, яке керує розміщенням сховища та поведінкою відкликання на всіх підключених рівнях.

Hitachi Content Platform – це ієрархічний варіант керування сховищами, призначений для підприємств, які стандартизують переміщення між онлайн-сховищами та офлайн-репозиторіями на основі політик.

Основні можливості включають керування простором імен, оркестрацію політик зберігання та орієнтовані на життєвий цикл робочі процеси, які контролюють розташування даних на рівнях медіа.

Продукт інтегрується з цільовими сховищами через визначені конектори та підтримує процеси міграції та відкликання, що зменшує ручну обробку.

Функції управління зосереджені на адміністративному контролі поведінки політик та операційному аудиті подій переміщення даних.

Переваги:

– Оркестрація на основі політик для робочих процесів міграції та відкликання.

– Шаблони адміністрування підприємства для управління поведінкою життєвого циклу.

– Шлях інтеграції для підключення цільових сховищ та медіа-репозиторіїв.

– Зручні для аудиту операційні записи про події переміщення на складі.

Недоліки:

– Розробка політики рівнів вимагає ретельного управління, щоб уникнути несподіванок від відкликання.

– Налаштування робочого процесу може стати складним для кількох серверних частин сховищ.

– Усунення несправностей у процесі експлуатації залежить від глибокого розуміння завдань переміщення.

– Автоматизація та охоплення API можуть на практиці відставати від спеціалізованих робочих процесів HSM.

Найкраще підходить для: коли великим підприємствам потрібне кероване політиками багаторівневе розміщення даних у гетерогенному сховищі та офлайн-репозиторіях.

Oracle Hierarchical Storage Manager

Система керування сховищем на основі політик, яка переносить файли між кешем дисків та бібліотеками стрічок, раніше відома як SAM-QFS.

Оркестрація відкликання даних за допомогою HSM для офлайн-медіа, скоординована для забезпечення передбачуваних шаблонів доступу програм.

Oracle Hierarchical Storage Manager зосереджений на багаторівневому розподілі на стрічки та диски підприємства з відкликанням даних на основі політик та автоматизованим керуванням розміщенням. Його глибина інтеграції спрямована на екосистеми Oracle та загальні стеки сховищ, що зменшує обсяг роботи з налаштування для організацій, які вже стандартизують інструменти зберігання даних Oracle.

Переміщення даних регулюється робочими процесами міграції та відкликання, керованими HSM, які підтримують повернення офлайн-медіа з контрольованою затримкою. Оперативне покриття зосереджується на адміністративному управлінні щодо доступу до медіа в стилі резервного копіювання та забезпечення життєвого циклу для керованих наборів даних.

Переваги:

– Робочі процеси відкликання на основі політик для офлайн-медіа з

контрольованою поведінкою.

– Глибока відповідність середовищам зберігання даних та операційним інструментам, орієнтованим на Oracle.

– Автоматизація багаторівневого розміщення на стрічках та дисках зменшує обсяг роботи з ручної міграції.

– Контроль життєвого циклу підтримує передбачувані цикли рекультивації простору.

Недоліки:

– Вищі операційні витрати, ніж у продуктів для архівування лише файлів.

– Дисципліна управління необхідна для уникнення штормів відкликання під навантаженням.

– Поверхня автоматизації вужча для нестандартних серверів зберігання даних.

– Налаштування міграції потребує ретельного тестування для робочих навантажень, чутливих до затримки.

Найкраще підходить для: коли підприємствам, що працюють на базі Oracle, потрібне автоматизоване відкликання стрічки та забезпечення рівності для керованих наборів даних.

QStar Archive Manager

Програмне забезпечення для архівування та HSM, яке переносить файлові дані між дисками, об'єктними сховищами, стрічками та хмарними сховищами.

Оркестрація відкликання на основі політик, яка керує маршрутизацією відкликання для зменшення затримки відкликання та конфліктів у черзі під час активного відновлення.

QStar Archive Manager – це програмне забезпечення для керування ієрархічним сховищем, яке зосереджено на робочих процесах архівування на основі політик, пов'язаних зі стрічками та іншими довгостроковими сховищами. Воно обробляє правила розміщення, архівування файлів і каталогів, а також

оркестрацію відкликання, щоб архівований вміст можна було повернути з контрольованою затримкою.

Інструмент робить акцент на автоматизації процесів прийому, переміщення та зберігання даних, а не на ручних операціях на основі заявок. Глибина інтеграції зосереджена на підключенні архівних цілей та шляхів передачі до механізму виконання, який виконує ці політики від початку до кінця.

Переваги:

– Оркестрація архівування та відкликання на основі політик для зберігання на плівку.

– Автоматизує прийом, переміщення та дії життєвого циклу як єдиний робочий процес.

– Підтримує контрольовані шляхи відкликання для зменшення піків затримки відкликання.

– Операційна спрямованість на завдання довгострокового зберігання, а не на загальну синхронізацію файлів.

Недоліки:

– Менше точок інтеграції, ніж у корпоративних пакетах HSM для гетерогенних серверних частин.

– Управління залежить від дисциплінованого розроблення політики, щоб уникнути неправильного використання.

– Налаштування відтворення може вимагати методу спроб і помилок, коли існує конфлікт пропускної здатності.

– Обмежена видимість у лінійці даних вищого рівня порівняно зі спеціалізованими шарами метаданих.

Найкраще підходить для: коли операційним командам потрібне автоматизоване архівування та відкликання даних на стрічках із суворою дисципліною зберігання.

HPE Data Management Framework

HPE Data Management Framework автоматизує переміщення файлів між рівнями сховища для високопродуктивних обчислювальних середовищ.

Інтегрована координація робочого процесу заглушення та відкликання, яка керує затримкою відкликання файлів під час змін політики рівнів.

HPE Data Management Framework координує переміщення даних на основі політик між ієрархічними рівнями сховища, включаючи автономні носії та дискові репозиторії. Основна цінність фреймворку полягає в оркестрації навколо класифікації рівнів, заглушення та поведінки відкликання, що зменшує ручну обробку під час змін життєвого циклу.

Робочі процеси адміністратора зосереджені на встановленні правил переміщення та зберігання, а також на відстеженні активності операцій, пов'язаних зі сховищем. Глибина інтеграції зосереджена на стеках сховищ HPE та пов'язаних службах обробки даних, з використанням перехоплювачів автоматизації, які підтримують шаблони розгортання, специфічні для середовища.

Переваги:

– Оркестрація на основі політик для переходів рівнів між дисками та офлайн-носіями.

– Підтримка заглушення зменшує обсяг сховища перед операціями відкликання HSM.

– Поведінку відкликання можна налаштувати для керування затримками доступу до файлів.

– Відстеження активності забезпечує операційну видимість під час переміщень по рівнях.

Недоліки:

– Найкращі результати залежать від тісної узгодженості з інтеграціями сховищ HPE.

– Складність операційної діяльності зростає, коли взаємодіють кілька правил рівневої структури та міграції.

– Налаштування поведінки відтворення може бути важко перевірити без контрольованого тестування.

– Глибина автоматизації найвища в екосистемах HPE, а не в багатопостачальних комплексах.

Найкраще підходить для: коли команди використовують ієрархії сховищ, орієнтовані на HPE, і їм потрібна оркестрація відкликання на основі політик із заглушенням.

Dell PowerScale SmartPools

PowerScale SmartPools розміщує файли в пулах сховищ відповідно до політик, атрибутів файлів та шаблонів доступу.

Класифікація файлів на основі правил SmartPools, яка пов'язує розміщення пулу з адміністративною політикою без використання зовнішніх інструментів міграції.

Dell PowerScale SmartPools централізує керування політиками багаторівневого розподілу для файлових систем PowerScale OneFS, зіставляючи класи сховищ із правилами SmartPools. Він автоматизує рішення щодо розміщення даних за допомогою призначення на основі політик, тому файли потрапляють у потрібний пул без ручного виконання операцій для кожного шарду окремо.

Він також підтримує поведінку керування потужностями, таку як балансування та відновлення простору, пов'язані з членством у пулі. Адміністративні робочі процеси зосереджені на послідовному управлінні класифікацією рівнів, а не на наданні окремого стеку оркестрації міграції.

Переваги:

– Правила розміщення на основі політик для класифікації рівнів файлів PowerScale.

– Централізоване управління пулом для зменшення дрейфу ручного розміщення даних.

– Вбудовані моделі поведінки для балансування та управління ємністю пулу.

– Працює в операціях OneFS без потреби в зовнішніх оркестраторах.

Недоліки:

– Обмежено екосистемами PowerScale та семантикою сховища OneFS.

– Охоплення робочих процесів залишається зосередженим на правилах пулу, а не на широких каналах відкликання.

– Контроль затримки відкликання залежить від базових функцій та налаштувань OneFS.

– Управління чутливе до впорядкування правил та операційної дисципліни.

Найкраще підходить для: коли адміністраторам PowerScale потрібне автоматизоване розміщення файлів у пулах з узгодженим управлінням рівнями та мінімальним використанням зовнішніх інструментів.

Висновок

Після оцінки 10 рішень для переміщення сховищ, Spectra StorCycle виділяється як наш найкращий вибір – він отримав найвищий бал за нашими сукупними критеріями функціональності, простоти використання та вартості, тому він посідає перше місце у вищезазначеному рейтингу.

Як вибрати правильне програмне забезпечення для ієрархічного управління сховищами

Програмне забезпечення для керування ієрархічним сховищем автоматизує розміщення та відкликання рівнів на диску, у хмарному сховищі об'єктів та офлайн-носіях, тому метою є передбачуване переміщення даних в рамках управління сховищем. Цей розділ охоплює Spectra StorCycle, Hammerspace та Spectra Logic BlackPearl, а також Hammerspace, Quantum StorNext, Oracle Hierarchical Storage Manager та інші іменовані платформи політик та відкликання з наданих карток інструментів. Кожна картка інструментів зосереджена на певному операційному механізмі, такому як міграція файлів на основі політик, розміщення в глобальному просторі імен або оркестрація відкликання зі стрічки.

Розділи оцінювання, що йдуть далі, відстежують глибину інтеграції між гетерогенними серверними системами, широту поверхні автоматизації політик та адміністративні елементи керування, що запобігають штормам неправильного розміщення та відкликання. Spectra StorCycle закріплює переміщення політик файлів між NAS, хмарним сховищем об'єктів, диском, стрічкою та Spectra BlackPearl, тоді як Hammerspace зосереджується на єдиному просторі імен у розподілених NAS, паралельних файлових системах та сховищах об'єктів. Інші інструменти охоплюють вужчі області застосування, такі як відкликання стрічки, вирівняне з Oracle, заглушення HPE для переходів між рівнями або класифікація пулів PowerScale SmartPools для OneFS.

Програмне забезпечення для ієрархічного керування сховищами даних для розміщення та відкликання рівнів на основі політик у гетерогенному сховищі

Програмне забезпечення для керування ієрархічним сховищем застосовує політики рівнів та міграції для розміщення файлів між рівнями, координації затримки або відновлення простору, де це можливо, та організації відкликання, щоб відновлення відповідало контрольованим робочим процесам. Spectra StorCycle поєднує механізм політик, який переміщує або копіює файли між NAS, хмарним сховищем об'єктів, диском, стрічкою та Spectra BlackPearl, із шлюзами перевірки передачі, які впливають на те, коли файли видаляються або копіюються.

Hammerspace використовує Global Data Environment для об'єднання розподілених NAS, паралельних файлових систем, сховищ об'єктів та хмарних ресурсів в одному просторі імен, а потім запускає автоматизацію політик для розміщення даних у різнорідних середовищах. Quantum StorNext також пов'язує поведінку відкликання та повернення простору зі своїм механізмом політик розміщення на рівнях, щоб робочі процеси залишалися узгодженими на дисках, стрічках та в хмарі для файлових робочих навантажень.

Оцінка категорії: глибина інтеграції, поверхня автоматизації та засоби управління

Багаторівневе керування сховищем працює лише тоді, коли продукт поєднує фактичні шляхи переміщення та шляхи відкликання між диском, NAS, хмарним сховищем об'єктів та офлайн-носіями. Spectra StorCycle побудовано на основі переміщень, контрольованих політиками, між NAS, хмарним сховищем об'єктів, диском, стрічкою та Spectra BlackPearl, тому оркестрація охоплює як розміщення, так і дії життєвого циклу в одному робочому процесі.

Гетерогенне інтеграційне покриття з чітко визначеними цілями

Spectra StorCycle переміщує або копіює файли між NAS, хмарним об'єктним сховищем, диском, стрічкою та Spectra BlackPearl за допомогою свого механізму політик. Hammerspace застосовує автоматизацію політик у розподілених NAS, паралельних файлових системах, об'єктних сховищах та хмарних цільових системах у рамках Global Data Environment.

Розміщення, зумовлене політикою, пов'язане з відкликанням та рекультивацією простору

Quantum StorNext пов’язує свій механізм політик розміщення на рівнях із поведінкою відкликання та відновлення простору для узгоджених робочих процесів з файлами на диску, стрічці та в хмарі. Spectra StorCycle також пов’язує дії політик із шлюзами перевірки переміщення, які впливають на те, коли файли видаляються або копіюються.

Зворотній зв'язок від аналітики до політики для автоматизованого розміщення даних

Data Dynamics StorageX пов’язує аналіз шаблонів доступу з автоматизованими політиками розміщення файлів, щоб цілі міграції відповідали спостережуваному використанню. Hitachi Content Platform зосереджується на централізованому управлінні політиками міграції та відкликання на всіх підключених рівнях, а не на аналітиці використання.

Уніфікований контроль простору імен для кількох типів сховищ

Hammerspace використовує Global Data Environment для об'єднання розподілених NAS, паралельних файлових систем, сховищ об'єктів та хмарних ресурсів в одному просторі імен. Dell PowerScale SmartPools зосереджує правила класифікації та розміщення пулів у семантиці сховищ PowerScale та OneFS, а не в кросплатформному просторі імен.

Координація заглушення та відкликання для контрольованих переходів між рівнями

HPE Data Management Framework забезпечує підтримку заглушення, що зменшує обсяг сховища перед операціями відкликання HSM. Oracle Hierarchical Storage Manager зосереджується на оркестрації відкликання під керуванням HSM для автономних носіїв, щоб забезпечити передбачуваність шаблонів доступу до програм.

Як вибрати: зіставити площину керування продуктом із робочим процесом розподілу на рівні

Почніть з визначення того, чи потреби в багаторівневому розподілі полягають переважно в міграції на рівні файлів відповідно до механізму політик, чи в оркестрації відкликання архівів на основі стрічки під суворішим контролем носіїв. Spectra StorCycle та Quantum StorNext побудовані на основі переміщення файлів та дій життєвого циклу, керованих політиками, тоді як Oracle Hierarchical Storage Manager та QStar Archive Manager зосереджені на поведінці відкликання носіїв офлайн.

Виберіть площину керування, яка відповідає домінантному робочому процесу

Якщо розміщення файлів та дії життєвого циклу мають охоплювати NAS, диск, хмарне сховище об'єктів та стрічку, оберіть Spectra StorCycle, оскільки він переміщує або копіює файли між цими цільовими сховищами та може інтегруватися зі Spectra BlackPearl. Якщо середовище перевантажене стрічками зі суворою дисципліною маршрутизації відкликання та збереження, оберіть QStar Archive Manager, оскільки він автоматизує прийом, переміщення та дії

життєвого циклу як один робочий процес із маршрутизацією відкликання для зменшення затримки та конфліктів у чергах.

Вирішіть, чи має аналітика використання визначати правила розміщення

Якщо вимірювання шаблонів доступу має безпосередньо впливати на рішення щодо рівнів, оберіть Data Dynamics StorageX, оскільки StorageX Analytics пов’язує аналіз шаблонів доступу з автоматизованими політиками розміщення файлів. Якщо автоматизація політик має бути централізовано організована без аналітичного зворотного зв’язку, оберіть Hitachi Content Platform, оскільки вона виконує централізоване управління політиками для міграції та відкликання на всіх підключених рівнях.

Виберіть для об'єднання простору імен або класифікації, специфічної для середовища

Якщо єдиний логічний простір імен має охоплювати розподілену мережеву мережу (NAS), паралельні файлові системи, сховище об'єктів та хмару, оберіть Hammerspace, оскільки Global Data Environment об'єднує ці середовища в одному просторі імен. Якщо розміщення в першу чергу пов'язане з класифікацією файлів PowerScale в пули з мінімальним використанням зовнішніх інструментів, оберіть Dell PowerScale SmartPools, оскільки воно забезпечує класифікацію файлів на основі правил, пов'язану з адміністративною політикою.

Узгодьте поведінку заглушення та відкликання з очікуваннями щодо доступу до додатків

Якщо переходи між рівнями вимагають затримки, щоб зменшити обсяг сховища перед відкликанням, оберіть HPE Data Management Framework, оскільки вона підтримує затримку та координує час робочого процесу відкликання. Якщо відкликання офлайн-носія має відповідати поведінці доступу, керованої HSM, для передбачуваних шаблонів доступу до програм, оберіть Oracle Hierarchical Storage Manager, оскільки його оркестрація відкликання, керована HSM, координує шаблони доступу до офлайн-носія.

Перевірте, чи відповідає показник запам'ятовуваності цільовій затримці медіа

Якщо швидкість відкликання обмежена затримкою стрічки або хмарного цільового об'єкта, швидкість відкликання Spectra StorCycle залежить від затримки вибраного стрічкового або хмарного цільового об'єкта. Якщо необхідно підтримувати послідовну поведінку відкликання та відновлення простору на диску, стрічці та в хмарі за допомогою зв'язування політик, оберіть Quantum StorNext, оскільки він пов'язує розміщення на рівнях з відкликанням та відновленням простору, тому робочі процеси залишаються узгодженими.

Кому потрібне програмне забезпечення для управління ієрархічним сховищем даних

Організації, які використовують кілька рівнів сховища та повинні керувати переміщенням, потребують цих інструментів, оскільки політики рівнів повинні керувати поведінкою розміщення та відкликання на фактичних серверах. Spectra StorCycle орієнтований на команди, яким потрібна міграція файлів з контрольованими політиками між NAS, хмарним об'єктним сховищем, диском, стрічкою та Spectra BlackPearl.

Команди з операцій зі зберігання даних, що працюють зі змішаними архівами NAS та стрічковими архівами

Spectra StorCycle підходить для операцій зберігання даних, які повинні переміщувати або копіювати файли між NAS, дисками, стрічками та хмарними об'єктними сховищами в рамках єдиного механізму політик зі шлюзами перевірки передачі. Quantum StorNext також підходить цим командам, коли відкликання та повернення простору повинні залишатися узгодженими в рамках централізованої політики.

Команди з корпоративної архітектури консолідують доступ до даних у гетерогенному сховищі

Hammerspace підходить для команд, яким потрібне глобальне середовище даних (Global Data Environment) для об'єднання розподілених NAS, паралельних файлових систем, сховищ об'єктів та хмари в одному просторі імен для

розміщення на основі політик. Hitachi Content Platform підходить для підприємств, яким потрібна централізована оркестрація політик на всіх підключених рівнях для міграції та відкликання.

Команди медіаархівів з регульованим зберіганням та відстежуваними операціями з відкликання

Архів об'єктів FUJIFILM підходить для колекцій з великим обсягом медіафайлів, які потребують оркестрації відкликання Fujifilm, пов'язаної з регульованими політиками зберігання. Spectra StorCycle також може підтримувати робочі процеси архівування на плівках завдяки вбудованій інтеграції BlackPearl, але FUJIFILM Object Archive більше відповідає операційному відкликанню Fujifilm.

Команди, що оптимізують рішення щодо рівнів, використовуючи виміряну поведінку доступу

Data Dynamics StorageX підходить корпоративним командам, яким потрібне розміщення політик на основі аналітики, оскільки StorageX Analytics поєднує аналіз шаблонів доступу з автоматизованими політиками розміщення файлів. Інші оркестратори політик більше зосереджуються на правилах та оркестрації, ніж на аналізі шаблонів доступу.

HPE-орієнтовані середовища, що керують відкликанням за допомогою заглушок

HPE Data Management Framework відповідає ієрархіям сховищ HPE, які потребують координації робочих процесів заглушення та відкликання для керування затримкою відкликання файлів під час змін політики рівнів. Інші продукти можуть підтримувати відкликання, але не позиціонують заглушення як основний механізм у наданих картках інструментів.

Типові помилки в проектах ієрархічного управління сховищами

Автоматизація рівнів не спрацьовує, коли розробка політики рівнів розглядається як одноразова конфігурація, а не як робочий процес управління з тестуванням за реального навантаження відкликання. Кілька інструментів явно позначають вимоги до дисципліни управління, оскільки шторми відкликання та

неправильне розміщення можуть виникати, коли політики не налаштовані.

Розгляд розробки політики як статичної вправи без управлінської дисципліни

Oracle Hierarchical Storage Manager попереджає, що для уникнення штормів відкликання під навантаженням необхідна дисципліна управління. Hitachi Content Platform також зазначає, що розробка політики рівнів вимагає ретельного управління, щоб уникнути несподіванок щодо відкликання.

Припускаючи, що продукт обробляє навантаження блочного сховища

Data Dynamics StorageX орієнтований на файли та явно виносить робочі навантаження блочного сховища за межі своєї операційної моделі. Spectra StorCycle зосереджується на переміщенні файлів та поведінці копіювання або видалення, а не на багаторівневому розподілі на рівні блоків.

Ігнорування того, як обмеження відкликання та затримка впливають на операції

Quantum StorNext вказує на необхідність операційного налаштування для управління обмеженням швидкості відкликання та затримкою. Spectra StorCycle також зазначає, що швидкість відкликання залежить від затримки стрічки або хмарного цільового об'єкта, тому планування пропускної здатності має враховувати обмеження носіїв та мережі.

Розгортання глобального інструменту простору імен без планування метаданих та структури мережі

Hammerspace зазначає, що розгортання вимагає ретельного планування метаданих, простору імен та мережі сайтів. Команди, які пропускають це планування, можуть зіткнутися з операційною складністю, навіть якщо політики правильні.

Вибір класифікатора, специфічного для серверної частини, коли потрібен широкий конвеєр відкликання

Dell PowerScale SmartPools обмежений екосистемами PowerScale та семантикою сховища OneFS, тому він зосереджений на правилах пулу, а не на

широких конвеєрах відкликання. QStar Archive Manager орієнтований на стрічки та має менше точок інтеграції, ніж корпоративні пакети HSM для гетерогенних серверів.

Як відібрали та оцінили ці інструменти

Оцінювали програмне забезпечення для ієрархічного керування сховищами на основі глибини інтеграції між цільовими системами сховища, зазначеними в кожній картці інструментів, та широти поверхні автоматизації політик для розміщення, відкликання та пов'язаних дій життєвого циклу. Функції становили 40% рейтингу, оскільки Spectra StorCycle поєднує переміщення або копіювання файлів, керовані політиками, між NAS, хмарним сховищем об'єктів, диском, стрічкою та Spectra BlackPearl, а також перевіряє передачі перед видаленням або копіюванням.

Зручність/цінність кожен внесок становив 30%, оскільки операційне налаштування, складність управління та зусилля налаштування впливають на те, чи можуть команди запускати робочі процеси відкликання без несподіванок із затримкою. Spectra StorCycle посіла перше місце, оскільки поєднує гетерогенне покриття переміщень із вбудованою підтримкою робочих процесів архівування на стрічках BlackPearl та шлюзами перевірки передачі, які безпосередньо впливають на результати життєвого циклу.

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Для реалізації програмних компонентів системи ієрархічного зберігання даних із використанням моделей багатокритеріальної оптимізації було обрано мову програмування Python. Вибір цього інструментарію зумовлений його високою продуктивністю розробки, гнучкістю, наявністю розвинених структур даних та потужної екосистеми бібліотек для математичного моделювання, аналізу даних і оптимізації.

Особливості мови та її застосування в задачах оптимізації сховищ

Python є високорівневою об'єктно-орієнтованою мовою загального призначення з відкритим вихідним кодом. Динамічна типізація та інтерпретований характер виконання коду забезпечують зручність прототипування складних алгоритмів багатокритеріальної оптимізації, які враховують параметри вартості зберігання та швидкодії компонентів ієрархічних систем.

У контексті дослідження систем ієрархічного зберігання даних ключовими перевагами Python є:

– Розвинена екосистема для аналізу й обчислень: наявність спеціалізованих бібліотек (NumPy, Pandas) дозволяє ефективно обробляти великі масиви метрик, що описують інтенсивність звернень до даних, вартість носіїв та часові характеристики їхньої продуктивності.

– Підтримка методів машинного навчання та математичного моделювання: забезпечує можливість реалізації евристичних алгоритмів оптимізації, побудови прогностичних моделей завантаженості сховищ та прийняття рішень щодо міграції даних між рівнями ієрархії.

– Лаконічність і читаність коду: мінімізує витрати часу на написання супроводжуваного програмного забезпечення для моделювання та тестування стратегій розподілу даних.

Завдяки своїй універсальності та потужним аналітичним можливостям, Python забезпечує створення ефективного, гнучкого та легко адаптованого програмного ядра для моделювання і реалізації систем ієрархічного зберігання даних.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за другим (магістерським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи ієрархічного зберігання даних на

основі моделей багатокритеріальної оптимізації вартості та швидкодії.

В процесі розробки випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи контролю роботи технологічного обладнання на виробництві в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

в) розробити програмне забезпечення системи, що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів програми та підпрограми;

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про некоректні дії користувача та нестандартні ситуації в роботі технологічного обладнання;

д) розробити рекомендації по організаційних та методичних заходах, які забезпечать впровадження системи в промислову експлуатацію та її подальшу успішну експлуатацію;

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

з) сформувати висновки про виконаний обсяг робіт та одержані результати.

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Якщо ви все ще скидаєте все у свій SIEM, ваш бюджет скоро вичерпається, якщо він ще не вичерпався. Якщо ви фільтруєте або вибірково аналізуєте дані, щоб вони відповідали ліміту ліцензії, ви, ймовірно, підозрюєте, що пропускаєте щось важливе. А якщо ви відкладаєте переповнення SIEM у холодне сховище, щоб заощадити гроші, ви вже знаєте, що відновлення цих даних – справжній головний біль.

Щось із цього звучить знайомо? Тоді настав час для багаторівневої стратегії обробки даних. Почніть з одного питання: куди йдуть ваші повноцінні дані? Спочатку пропустіть їх через конвеєр для кращого управління та контролю витрат, щоб до вашої SIEM або системи аналізу потрапляли лише критично важливі, практичні дані. Потім структуруйте все інше за використанням та цінністю.

Дані, до яких рідко звертаються, потребують архіву з видимістю та доступом у режимі реального часу, без затримок регідратації. Пошук за принципом «голка в копиці сіна» потребує високошвидкісного стовпцевого сховища, яке забезпечує аналітику, інформаційні панелі та запити в режимі реального часу. Дані аудиту та звітності потребують доступного та відновлюваного сховища. Саме це пропонує розроблене програмне забезпечення: архітектура сховища, створена для управління телеметричними даними протягом наступного десятиліття.

Розроблене програмне забезпечення поєднує високошвидкісний механізм запитів із економією коштів на сховищі об'єктів, завдяки чому ваша команда отримує миттєві запити без прив'язки до постачальника чи зайвих витрат на зберігання. Завдяки маршрутизації та зберіганню даних в оптимальному форматі, це може знизити витрати на 50% порівняно з традиційними рішеннями.

Запити автоматично спрямовуються на найшвидше доступне джерело, а повноцінні дані залишаються для повторного відтворення, коли вам це потрібно.

Причини для впровадження багаторівневої стратегії обробки даних:

– Зберігайте дані на основі цінності, використання та потреб доступу. Спрямовуйте високоцінні, часто використовувані дані на оптимізовані для продуктивності рівні для аналітики в режимі реального часу. Зберігайте менш критичні або дослідницькі дані в економічно ефективних сховищах. Ви скорочуєте витрати та забезпечуєте доступність даних.

– Використовуйте відкриті формати для об'єднання даних. Консолідуйте фрагментовані дані в єдине джерело достовірної інформації на різних рівнях, щоб аналіз проводився без зайвого дублювання або переміщення між системами.

– Надайте пріоритет гнучкому та масштабованому аналізу даних. Поєднуйте обчислення на вимогу з економічно ефективним сховищем для аналізу великих обсягів без нескінченного розширення інфраструктури.

– Використовуйте економічно ефективне сховище для дотримання вимог щодо відповідності та аудиту. Зберігайте повноцінні дані в недорогому об'єктному сховищі для довгострокового зберігання, а потім витягуйте та відтворюйте їх у разі виникнення аудитів та розслідувань.

– Зменште поширення та складність інструментів. Консолідуйте платформи безпеки та спостереження, щоб зменшити залежність від спеціалізованих інструментів, мінімізувати фрагментацію набору навичок та спростити життєвий цикл даних.

Не дозволяйте своїй ліцензії SIEM вирішувати, які дані ви зберігатимете

Розроблене програмне забезпечення надає інструменти для керування та аналізу телеметрії без прив’язки до певних систем, втрати даних та залучення додаткових агентів. Розроблене програмне забезпечення – це центральний вузол, який зменшує обсяг і складність даних, скорочує витрати, підтримує міграцію SIEM та забезпечує відповідність вимогам без порушення роботи систем, які ви вже використовуєте.

Кожен продукт охоплює один рівень багаторівневої стратегії зберігання даних. Stream обробляє та маршрутизує дані в режимі реального часу, тому до вашої SIEM-системи потрапляють лише корисні дані. Edge збирає дані біля джерела. Search виконує федеративні запити на кожному рівні без переміщення або повторної гідратації даних. Lake разом з Lakehouse забезпечує багаторівневе сховище даних у озері даних, яке автоматично узгоджує вартість із цінністю.

Оскільки відокремлює обчислення від сховища та зберігає дані у відкритих форматах, ви можете впроваджувати нові інструменти, мігрувати платформи або переносити історичні дані на швидкий рівень без їхньої блокування чи втрати. Кожен байт залишається доступним, незалежно від того, чи він забезпечує роботу панелі інструментів у режимі реального часу сьогодні, чи аудиту через два роки.

3.2 Розробка структурної схеми

Зберігання даних стало справжнім вузьким місцем штучного інтелекту. Розкриття високопродуктивного штучного інтелекту в великих масштабах тепер вимагає архітектурних змін, а не кращих моделей.

Штучний інтелект не досяг межі моделі. Він досяг межі архітектури даних, особливо враховуючи, що робочі навантаження переходять від епізодичних навчальних запусків до безперервного розподіленого логічного висновку:

– Межі продуктивності ШІ зміщуються з моделей на архітектуру. (Або, з «чого?» на «де?» та «як?»).

– Традиційне сховище файлів не було розроблено для паралельності, розподілу та безперервного доступу до сучасних робочих навантажень штучного інтелекту.

– Зберігання об'єктів стає системою запису для високопродуктивного штучного інтелекту.

Протягом останніх кількох років у розмовах про інфраструктуру штучного інтелекту домінували моделі та обчислення. Більша кількість параметрів. Швидші графічні процесори. Щільніші кластери. Але у 2026 році інше обмеження стане неможливим ігнорувати: як дані зберігаються, отримують доступ до них та подаються системам штучного інтелекту у великих масштабах.

Глобальні витрати на інфраструктуру штучного інтелекту перевищили 250 мільярдів доларів у 2025 році , причому сховища та мережі зростають майже так само швидко, як і обчислювальні технології. Проте понад 50% організацій повідомляють про вузькі місця в даних та сховищах, які обмежують продуктивність та масштабованість ШІ. Водночас 57% підприємств стверджують, що їхні дані не готові до ШІ, навіть попри прискорення експериментів.

Наслідок очевидний: ШІ сповільнюється не тому, що моделі зупинилися або обчислювальна потужність недоступна, а тому, що архітектури даних, побудовані для ери до ШІ, виходять за межі своїх можливостей.

Архітектура сховища штучного інтелекту стосується проектування систем, які зберігають, керують та передають дані для машинного навчання та робочих навантажень штучного інтелекту.

На відміну від традиційних корпоративних сховищ, архітектури сховищ на основі штучного інтелекту повинні підтримувати:

– Масовий паралельний доступ за допомогою графічних процесорів та розподілених обчислювальних кластерів

– Різні архітектури сховища для навчання та логічного висновку

– Великі неструктуровані набори даних, такі як зображення, відео, журнали та навчальні корпуси

– Розподілені середовища, що охоплюють хмару, локальну інфраструктуру та периферійні розташування

Сучасні платформи штучного інтелекту все більше покладаються на високопродуктивне сховище об'єктів як центральний рівень даних, що дозволяє

обчисленням та сховищу масштабуватися незалежно, зберігаючи при цьому спільний доступ до наборів даних.

2026 рік є переломним моментом для штучного інтелекту в сфері зберігання даних, оскільки перехід від епізодичного навчання до безперервного розподіленого логічного висновку виявляє фундаментальні обмеження застарілих архітектур сховищ у масштабі підприємства.

Робочі навантаження штучного інтелекту (ШІ) зазнають структурних змін. Ранні системи ШІ були зосереджені на навчанні. Це означало великі, дискретні запуску, оптимізовані для пакетного доступу до даних. Ця модель швидко поступається місцем безперервному, ресурсомісткому логічному висновку, де моделі повинні обслуговувати та споживати дані постійно, а не ізольовано.

Вартість виводу на запит різко знизилася, але загальні витрати на штучний інтелект продовжують зростати, оскільки виведення є постійним та охоплює все підприємство. Моделі постійно запитуються, оцінюються, оновлюються та налаштовуються. Продуктивність тепер вимірюється в кількості токенів за секунду, а не в пакетній пропускній здатності. Дані більше не зчитуються та не архівуються один раз; до них постійно звертаються кілька систем, команд та агентів.

Однак майже дві третини організацій не успішно масштабували штучний інтелект по всьому підприємству, незважаючи на значні інвестиції та експерименти. Прогалина полягає в інфраструктурі, зокрема в системах зберігання даних, які ніколи не були розроблені для безперервних, спільних робочих навантажень ШІ.

Файлові системи залишаються ефективними для тісно пов'язаних навчальних кластерів, де обчислювальні та сховищні ресурси розташовані разом, а схеми доступу передбачувані. Напруження виникає, оскільки штучний інтелект розширюється за межі ізольованих навчальних завдань на загальнокорпоративні, спільні та постійно доступні набори даних.

Робочі навантаження штучного інтелекту є ефемерними, масово паралельними та все частіше розподіленими по хмарних, локальних та периферійних середовищах. Вони вимагають доступу до різнорідних джерел даних у режимі реального часу у великих масштабах. Оскільки продукти даних стають спільними та повторно використовуваними між командами, файлоорієнтовані архітектури стикаються з труднощами під спільним тягарем паралельності, управління метаданими та управління, особливо оскільки служби метаданих стають вузькими місцями за паралельного доступу.

Результатом є системне тертя: конфлікти за простір імен, накладні витрати на блокування, обмеження крихкого масштабування та зниження продуктивності під навантаженням. Файлові системи залишаються корисними для робочого простору та локалізованої обробки, але як система запису для ШІ, вони вводять обмеження саме там, де ШІ вимагає гнучкості. Сучасні шляхи передачі даних ШІ перепроектовуються навколо обходу ядра та передачі нульових копій зі сховища безпосередньо до пам'яті графічного процесора, шаблони, які принципово несумісні з файловим вводом-виводом.

Продуктивність ШІ тепер є проблемою архітектури даних

Однією з найдорожчих неефективностей інфраструктури штучного інтелекту сьогодні є недостатнє використання графічних процесорів. Тести показують, що оптимізовані архітектури сховищ можуть забезпечити до 5 разів більшу пропускну здатність, ніж звичайні S3 через HTTP, наприклад, перевищуючи сукупну пропускну здатність читання 100 ГБ/с порівняно з приблизно 20 ГБ/с у стандартних конфігураціях – різниця між графічними процесорами з живленням та тими, що простоюють. У міру масштабування моделей графічні процесори все частіше простоюють не через брак обчислювальних ресурсів, а тому, що системи зберігання даних та вводу-виводу не можуть передавати дані з пропускною здатністю та затримкою, які вимагають графічні процесори.

Галузеві дані показують, що пропускна здатність сховища, доступність даних та мережева взаємодія зараз є основними перешкодами для

продуктивності ШІ, поряд з обчислювальними можливостями. Водночас, пропускна здатність пам'яті та сховища стають жорсткими обмеженнями для масштабу ШІ, порівнянними з обмеженнями живлення та охолодження.

Це знаменує собою фундаментальний зсув. Ефективність інфраструктури штучного інтелекту більше не визначається щільністю обчислень. Вона визначається тим, чи можуть системи зберігання даних підтримувати паралельний високошвидкісний доступ до даних без затримки чи конфліктів.

Корпоративний штучний інтелект досягає успіху, коли команди можуть неодноразово створювати, оцінювати, розгортати та вдосконалювати моделі для мінливих наборів даних, вимог управління та версій моделей, без накопичення непотрібного розростання сервісів, додаткових переходів та операційних витрат.

Це вимагає єдиної бази даних, яка послідовно забезпечує:

– Швидка готовність даних без розростання платформи.

– Навчальні робочі процеси, що не пов'язані з операціями вводу/виводу.

– Передбачувана поведінка виводу в міру розширення вікон контексту та паралельності.

– Системи пошуку, які залишаються синхронізованими з вихідними даними.

– Масштабованість операційної діяльності, що збільшує потужність та пропускну здатність без ускладнення.

Сховище об'єктів побудовано навколо цих вимог та архітектурно узгоджено з шаблонами доступу та вимогами до масштабування сучасного штучного інтелекту.

На відміну від файлових систем, об'єктне сховище було розроблено з нуля для горизонтальної масштабованості, масивного паралельного доступу та розділених життєвих циклів обчислень та сховища. Програмно-визначене об'єктне сховище з апаратно-прискореними службами обробки даних тепер забезпечує доступ за менш ніж мілісекунду зі швидкостями, які раніше були винятковими для блоків та файлів. Воно підтримує відкриті формати даних та

дозволяє одночасно отримувати доступ до даних у розподілених середовищах без втрати масштабу.

Оскільки підприємства переходять до архітектур «дані як продукт», багаторазові та споживані набори даних стають важливими для масштабування ШІ. Аналітики прогнозують, що 50% великих підприємств до 2026 року позбудуться ізоляційних систем даних у рамках цього зрушення. Цей перехід залежить від систем зберігання даних, побудованих для сумісності, а не ізоляції.

Дані, готові до використання штучного інтелекту. Об'єктне сховище забезпечує цю основу.

Сховище об'єктів не є альтернативою файловому сховищу для ШІ. Це сховище даних ШІ.

Розподілений штучний інтелект робить сумісне сховище об'єктів обов'язковим.

Робочі навантаження ШІ за своєю суттю розподілені. Навчання, логічний висновок та підготовка даних все частіше відбуваються в різних середовищах, що зумовлено вимогами до вартості, продуктивності та суверенітету.

Понад 53% підприємств планують розгортати приватну або колокаційну інфраструктуру, виділену на основі штучного інтелекту , а 73% підприємств очікують, що навчання залишиться централізованим, тоді як логічний висновок все більше розподілятиметься географічно. Deloitte описує нову норму як трирівневу гібридну архітектуру штучного інтелекту, що охоплює хмару, ядро та периферію.

У цьому середовищі продуктивність залежить від сумісного доступу до даних через географічні та інфраструктурні кордони, а не від власних ізоляційних сховищ, обмежених одним середовищем. Дані повинні вільно переміщатися між середовищами, не змушуючи організації перепроектовувати конвеєри в міру зміни робочих навантажень. Архітектури сховищ, які прив'язують дані до певних платформ, підривають як продуктивність, так і гнучкість.

Що ламається без сховища об'єктів

Без сховища об'єктів як системи запису інфраструктура штучного інтелекту дає збій передбачуваним чином. Графічні процесори виснажуються, незважаючи на збільшення витрат. Набори даних фрагментуються в різних середовищах, оскільки системи зберігання не можуть забезпечити спільну, портативну систему запису. Проєкти штучного інтелекту зупиняються, оскільки робочі навантаження логічного висновку виявляють вузькі місця сховища, які були непомітними на етапах навчання. Витрати зростають, а використання ресурсів падає.

Ці збої посилюються фізичними обмеженнями. Обмеження живлення, пам'яті та охолодження вже обмежують розширення центрів обробки даних зі штучним інтелектом. Тим часом рентабельність інвестицій у штучний інтелект у підприємстві залишається недосяжною для більшості організацій.

Цього року сховище даних перестане бути невидимим шаром у системах штучного інтелекту. Воно стане центральним архітектурним рішенням.

Організації, які успішно масштабують штучний інтелект, переоцінять усталені припущення щодо систем обліку, нададуть пріоритет відкритим форматам даних, розроблять систему з урахуванням портативності та репатріації, а також відокремлять життєві цикли обчислень від життєвих циклів даних.

Файлові системи залишатимуться важливими інструментами. Але об'єктне зберігання буде основою. Майбутнє штучного інтелекту залежить від відкритих стандартів, сумісних архітектур даних та систем зберігання, спеціально розроблених для паралельного доступу та розподіленого масштабування.

Зверніть увагу на високу пропускну здатність, горизонтальну масштабованість, сумісність з S3, оптимізацію продуктивності та гнучкість розгортання.

По-перше, сховище має підтримувати високу пропускну здатність та паралельний доступ. Конвеєри штучного інтелекту часто включають тисячі

одночасних операцій читання та запису з графічних процесорів, конвеєрів навчання та служб логічного виводу.

По-друге, система повинна масштабуватися горизонтально. Набори даних штучного інтелекту швидко зростають, а сховище має розширюватися простим додаванням вузлів, а не вимагати переробки архітектури.

По-третє, сумісність з S3 є надзвичайно важливою. API S3 став стандартним інтерфейсом для сучасних платформ даних, конвеєрів машинного навчання та інструментів аналітики.

По-четверте, сховище об'єктів має бути оптимізоване для продуктивності, а не лише для ємності. Багато застарілих систем сховища об'єктів надають пріоритет варіантам використання архівів, а не високопродуктивним робочим навантаженням штучного інтелекту.

Зрештою, організаціям слід надати пріоритет гнучкості розгортання в хмарних, локальних та периферійних середовищах, щоб робочі навантаження штучного інтелекту могли переміщуватися без необхідності нових архітектур сховищ.

Високопродуктивні S3-сумісні платформи зберігання об'єктів, такі як MinIO, все частіше використовуються як система запису для сучасних конвеєрів штучного інтелекту.

Ключова відмінність зберігання об'єктів від зберігання файлів полягає в тому, як структуровані дані та як до них здійснюється доступ.

Файлове сховище організовує дані в ієрархічних каталогах за допомогою шляхів до файлів. Зазвичай воно спирається на централізовані служби метаданих і розроблено для передбачуваних шаблонів доступу в тісно пов'язаних системах.

Об'єктне сховище зберігає дані як об'єкти в плоскому просторі імен та отримує доступ до цих об'єктів через API, а не шляхи до файлів. Об'єкти також містять багаті метадані, які дозволяють системам керувати даними та отримувати їх у великих масштабах.

Для розподілених робочих навантажень штучного інтелекту, які потребують високої паралельності та спільних наборів даних, об'єктне сховище

забезпечує кращу масштабованість та архітектурну гнучкість, ніж традиційні файлові системи.

Об'єктне сховище краще для ШІ, оскільки воно підтримує масивний паралелізм, горизонтальну масштабованість та незалежне масштабування обчислювальних та сховищних ресурсів.

Робочі навантаження штучного інтелекту залежать від масового паралелізму, розподіленого доступу до даних та безперервного споживання даних.

Об'єктне сховище відповідає цим вимогам, оскільки воно дозволяє виконувати тисячі одночасних запитів, масштабується горизонтально між кластерами та дозволяє обчислювальним та сховищним ресурсам масштабуватися незалежно.

Він також природно інтегрується із сучасними архітектурами озер даних, відкритими форматами даних та фреймворками машинного навчання.

Ці характеристики роблять сховище об'єктів найкращою системою запису для наборів даних та конвеєрів штучного інтелекту.

Використовувати сховище файлів для робочих навантажень штучного інтелекту можливо, так як файлове сховище все ще може використовуватися для робочих навантажень штучного інтелекту, але зазвичай в обмеженій ролі, такій як тимчасовий простір для зберігання або локалізована обробка.

Сховище файлів часто корисне для тимчасового простору під час навчання моделі або локалізованих завдань обробки в тісно пов'язаних обчислювальних середовищах.

Однак, оскільки штучний інтелект поширюється між командами, платформами та географічними розташуваннями, файлові системи часто стають вузькими місцями через суперечки щодо метаданих, обмеження простору імен та обмеження масштабування.

Тому більшість сучасних архітектур штучного інтелекту використовують сховище об'єктів як основний рівень даних, а файлові системи зарезервовані для локалізованих обчислювальних робочих процесів.

API S3 є фактичним стандартним інтерфейсом для зберігання об'єктів, що забезпечує сумісність сучасних інструментів штучного інтелекту та обробки даних.

Багато сучасних інструментів штучного інтелекту та роботи з даними, включаючи фреймворки озер даних, платформи навчання машинному навчанню, аналітичні механізми та інструменти оркестрації, інтегруються з S3-сумісним сховищем.

Використання S3-сумісного об'єктного сховища забезпечує сумісність між інструментами та зменшує ризик прив'язки до певного постачальника.

Високопродуктивні S3-сумісні платформи, такі як MinIO, дозволяють організаціям розгортати цю архітектуру в хмарних, приватних інфраструктурах та периферійних середовищах.

Інфраструктура штучного інтелекту, що спирається на неправильну архітектуру сховища, зіткнеться з недовикористанням графічного процесора, фрагментованими наборами даних, виявленими вузькими місцями під час виведення даних та зростанням витрат зі зниженням ефективності.

Графічні процесори простоюють, оскільки системи зберігання даних не можуть достатньо швидко передавати дані. Набори даних фрагментуються в різних середовищах, що ускладнює обслуговування конвеєрів. Робочі навантаження логічного висновку виявляють вузькі місця, які не були помітні під час ізольованих навчальних запуску.

Результатом є зростання витрат на інфраструктуру разом зі зниженням ефективності.

У багатьох випадках обмежувальним фактором є не модель чи обчислювальна інфраструктура, а архітектура сховища, що підтримує дані.

Сучасні платформи штучного інтелекту використовують багаторівневу архітектуру з об'єктним сховищем як центральною системою запису, що підтримується кластерами обчислень на графічних процесорах, що працюють у гібридній інфраструктурі.

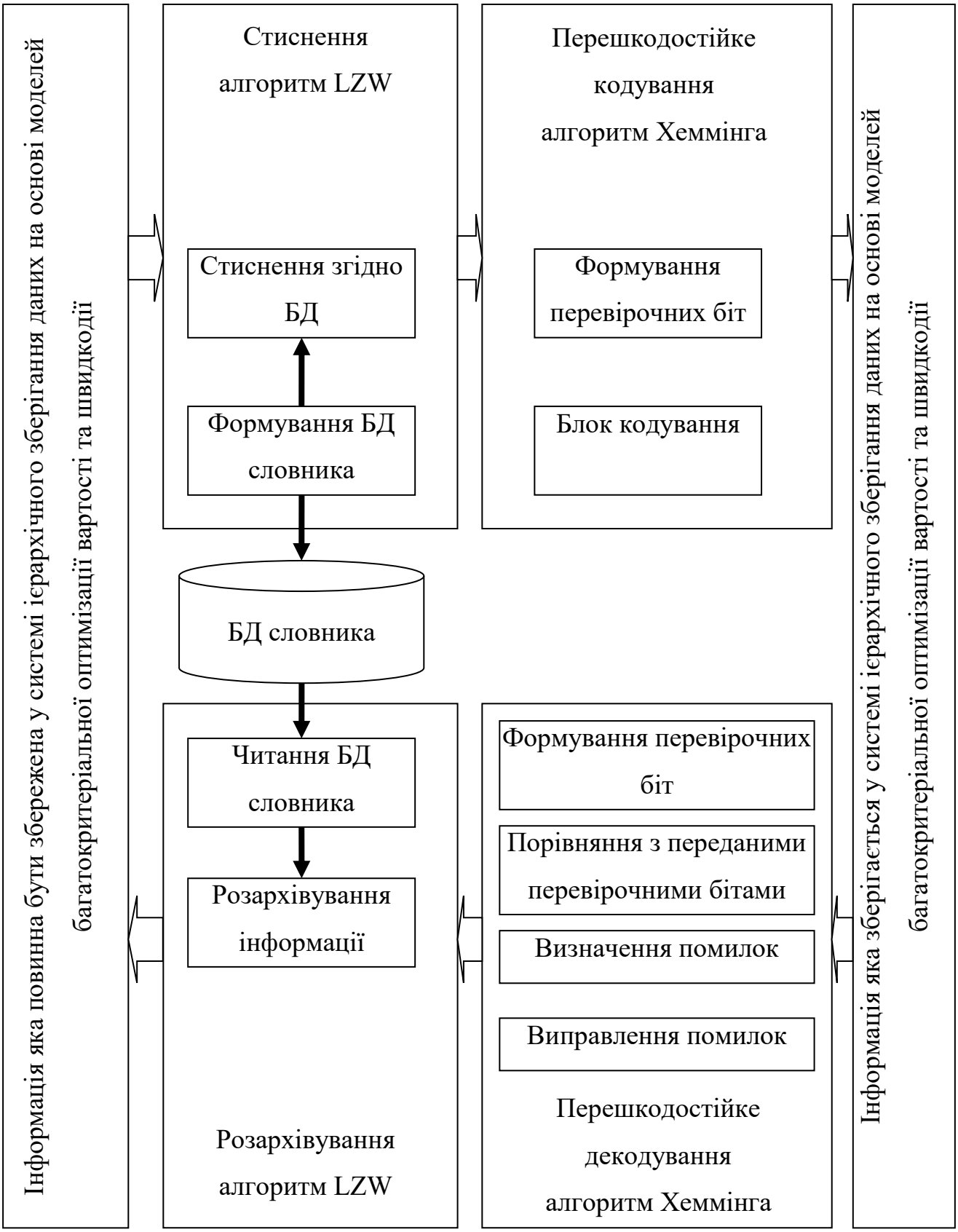


Рисунок 3.1 – Структурна схема системи

Сховище об'єктів служить центральною системою запису, зберігаючи набори даних, моделі та артефакти.

Обчислювальні кластери обробляють навантаження навчання та логічного висновку за допомогою графічних процесорів та прискорювачів.

Ці системи все частіше працюють у гібридній інфраструктурі, яка охоплює публічну хмару, приватні центри обробки даних та периферійні середовища.

Ця архітектура дозволяє організаціям масштабувати робочі навантаження штучного інтелекту, зберігаючи при цьому доступність, портативність та сумісність даних.

На рисунку 3.1 зображена структурна схема розробленого програмного забезпечення.

Для реалізації системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії, використовується процес архівування та розархівування інформації за алгоритмом LZW, та підданя заархівованої інформації перешкодостійкому кодуванню за алгоритмом Хеммінга.

3.3 Розробка функціональної схеми

На рисунку 3.2 наведено функціональну схему розробленої системи. Програмне забезпечення призначене для ефективного керування архівами даних, оптимізації процесів пакування та забезпечення зручності взаємодії з користувачем. До основних функціональних модулів комплексу належать:

– Модуль вибору ступеня стиску файлів: забезпечує гнучке налаштування алгоритмів компресії залежно від вимог до обсягу та швидкості обробки.

– Блок формування багатотомних архівів: призначений для розподілу великих масивів даних на окремі частини (томи) для зберігання на носіях з обмеженою місткістю.

– Блок добування інформації з багатотомних архівів: забезпечує коректне зчитування, збирання та відновлення даних із багатотомних структурованих файлів.

– Блок формування архівів, що саморозпаковуються (SFX): створює виконавчі файли (.exe), здатні до самостійної розпаковки без необхідності попереднього встановлення програми-архіватора.

– Модуль вибору режиму солід-архівів (solid): реалізує пакування файлів як єдиного безперервного потоку даних, що значно підвищує ступінь компресії за рахунок аналізу повторюваних послідовностей у масиві однотипних файлів.

– Блок моніторингу роботи архіватора: здійснює виведення протоколів та індикацію поточного стану виконання операцій.

– Блок вибору способу керування програмою: підтримує як інтерфейс командного рядка, так і вбудовану графічну оболонку з діалоговими панелями.

– Блок вибору мови інтерфейсу та довідкова система: забезпечують локалізацію програми та оперативне надання інформації щодо синтаксису й виконання команд.

– Особливості організації архівів та режимів стиснення

У ході проєктування особливу увагу приділено підтримці спеціалізованих форматів зберігання:

1. Багатотомні архіви: дозволяють ефективно оперувати об'ємними файлами шляхом їх фрагментації на зручні для транспортування чи зберігання частини.

2. SFX-архіви (Self-eXtracting): являють собою завантажувальні модулі, які після запуску в оперативній пам'яті автономно розпаковують і виконують закладені дані.

3. Солід-архіви (solid): забезпечують підвищений коефіцієнт стиснення завдяки спільній обробці сукупності файлів як єдиного потоку, що є оптимальним для архівування великих колекцій однорідних даних.

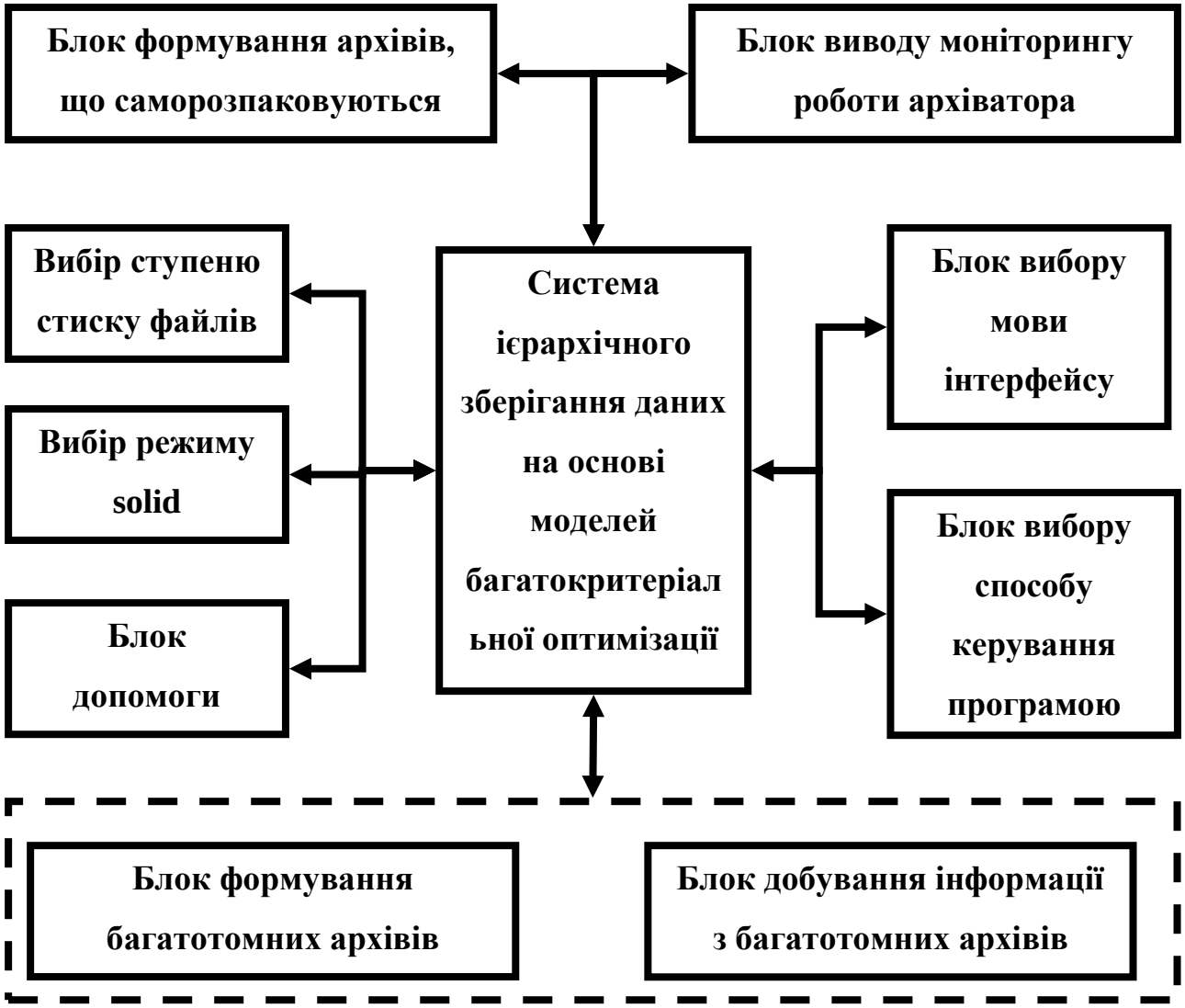


Рисунок 3.2 – Функціональна схема розробленої системи

Керування функціоналом програмного забезпечення може здійснюватися як за допомогою параметрів командного рядка (із застосуванням спеціальних ключів налаштування), так і через інтуїтивно зрозумілу вбудовану оболонку користувача, що забезпечує високий рівень комфорту та ефективності під час роботи з великими масивами даних.

Розглянувши усі блоки функціональної схеми перейдемо до розгляду діаграми взаємодії процесів, які відбуваються у системі.

3.4 Розробка діаграми процесів

Діаграму взаємодії процесів розробленої у межах магістерської роботи системи наведено на рисунку 3.3. Її детальний аналіз дозволяє простежити характер взаємодії внутрішніх компонентів та маршрутизацію інформаційних потоків у межах платформи ієрархічного зберігання даних.

У процесі проєктування застосовується модель графічного представлення потоків даних (Data Flow Diagram, DFD), яка наочно відображає рух, трансформацію та збереження інформації (зокрема, параметрів інтенсивності звернень до файлів, критеріїв вартості носіїв, часових характеристик швидкодії та результатів багатокритеріальної оптимізації).

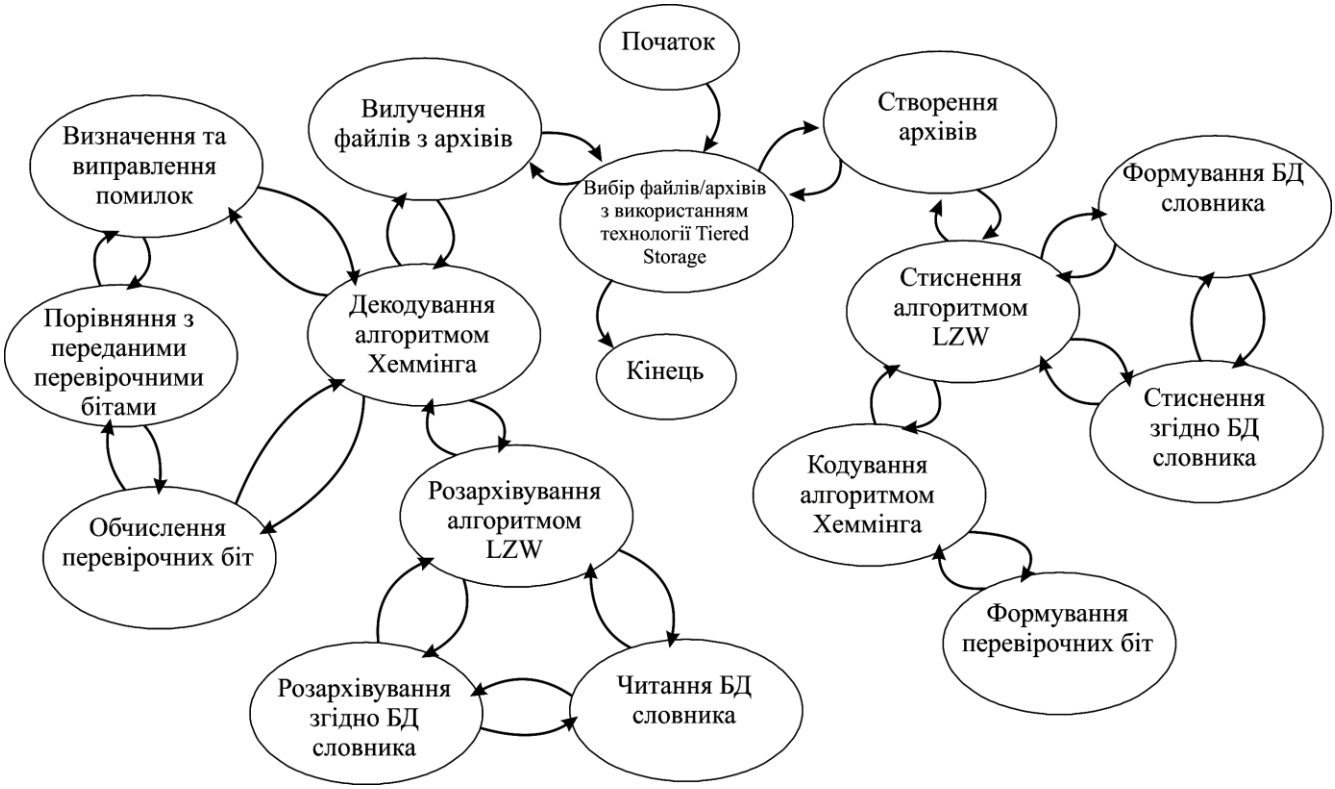


Рисунок 3.3 – Діаграма взаємодії процесів

Діаграми взаємодії процесів є базовим інструментом структурного проєктування для візуалізації обробки даних. Традиційно розробка розпочинається з побудови контекстної діаграми потоків даних, яка ілюструє загальну взаємодію розроблюваної системи ієрархічного зберігання зі зовнішнім

середовищем (користувачем, джерелами даних та сховищами різного рівня). У подальшому ця модель деталізується шляхом поетапної декомпозиції процесів і потоків даних з метою повної та всебічної демонстрації архітектури комплексу, що створюється.

Таким чином, розглянувши опис системи, структурну, функціональну схеми системи, та діаграму взаємодії процесів перейдемо до опису блок-схем основної програми, та підпрограм, які використовуються, для реалізації системи.

4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ПРАВИЛЬНІСТЬ ПРОЕКТНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Алгоритмічне забезпечення системи ієрархічного зберігання даних призначене для автоматизованого розміщення інформаційних об’єктів між рівнями сховища з різними характеристиками вартості, продуктивності та доступності.

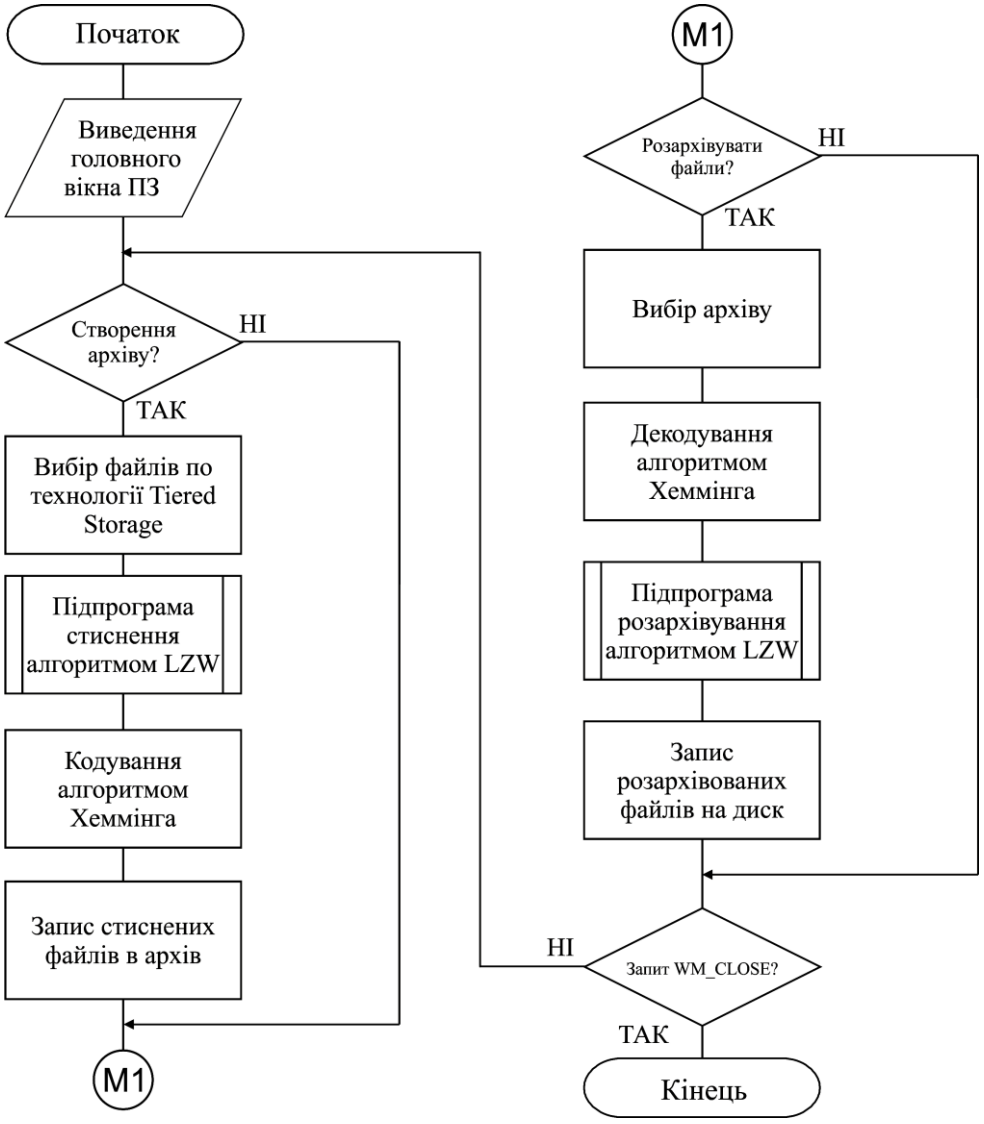


Рисунок 4.1 – Блок-схема основної програми

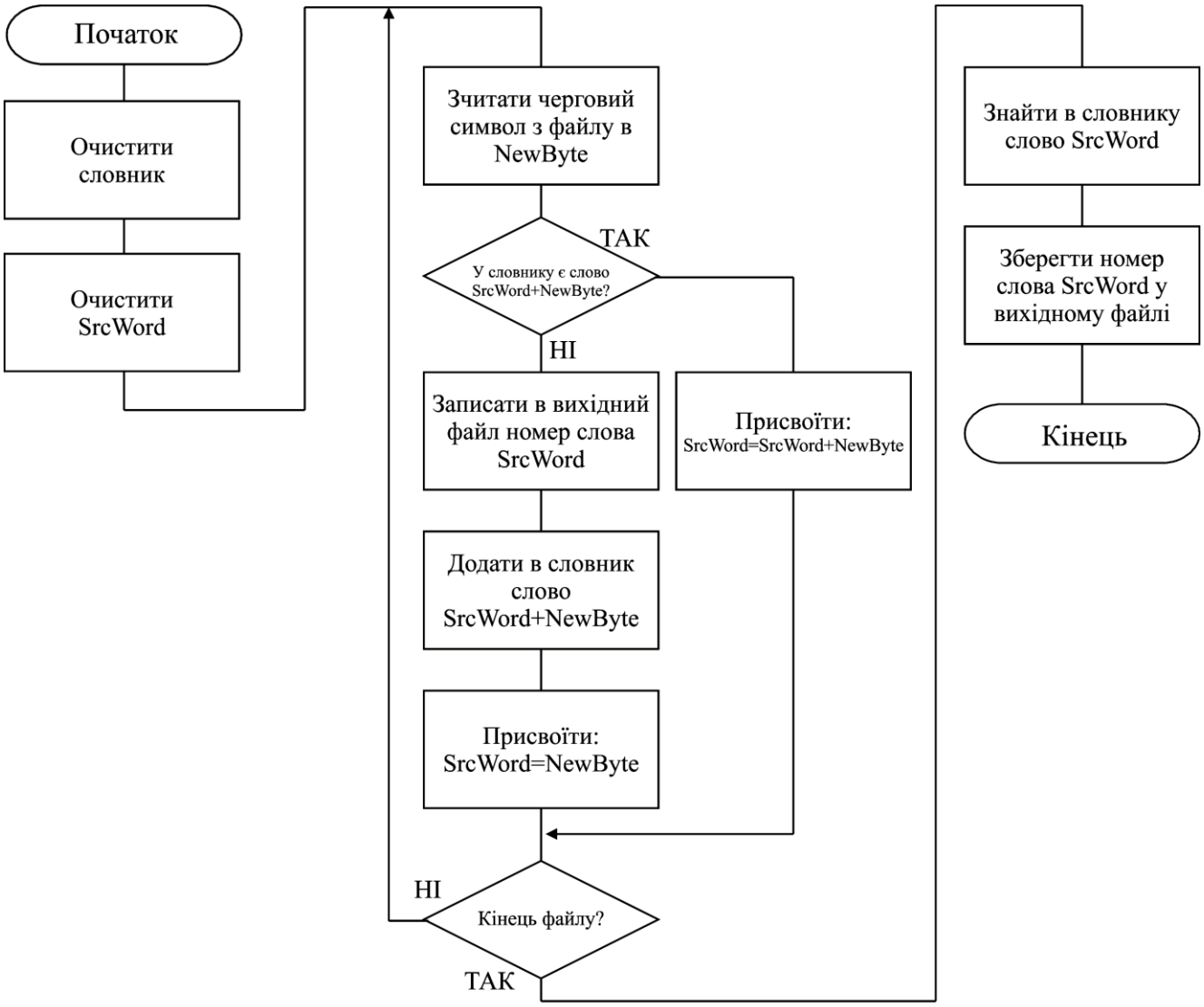


Рисунок 4.2 – Блок-схема роботи підпрограми

Основним завданням є визначення економічно доцільного способу зберігання даних за умови дотримання вимог до швидкості доступу. Для його виконання передбачається використання моделей багатокритеріальної оптимізації, які враховують обсяг інформації, частоту звернень, затримку операцій читання й записування, вартість зберігання та витрати на переміщення об’єктів.

Запропонований підхід дозволяє формувати індивідуальні рішення щодо розміщення даних відповідно до їхнього фактичного використання та встановлених експлуатаційних обмежень.

Загальну послідовність функціонування системи представлено на рисунку 4.1. Після запуску програмного комплексу виконується перевірка доступності рівнів зберігання та завантаження їхніх характеристик. Далі система отримує метадані інформаційних об’єктів і статистику звернень до них. На основі цих відомостей визначаються інтенсивність використання, поточна вартість розміщення та вимоги до продуктивності.

Після формування допустимих варіантів виконується багатокритеріальне оцінювання. Якщо інший рівень забезпечує прийнятні характеристики та економічний ефект з урахуванням витрат на міграцію, формується відповідне рішення. Після перенесення перевіряються цілісність даних і коректність їхнього розташування.

Для побудови програмного комплексу доцільно використовувати багаторівневу архітектуру з відокремленням фізичного зберігання від логічного керування об’єктами. Нижній рівень складається з доступних сховищ, які можуть використовувати твердотільні накопичувачі, жорсткі диски, об’єктні хмарні сервіси або архівні системи.

Над ними розміщується компонент логічної адресації, який забезпечує відповідність ідентифікатора об’єкта його фактичному місцю зберігання. Аналітична підсистема накопичує статистику використання даних. Оптимізаційний компонент визначає рекомендований рівень розміщення, а служба міграції виконує дозволені операції перенесення. Окреме сховище метаданих підтримує відомості про стан об’єктів, їхні версії та історію прийнятих рішень.

Особливістю ієрархічної організації є відмінність між швидкістю доступу та вартістю використання окремих рівнів. Високопродуктивні ресурси зазвичай доцільно використовувати для даних із високою інтенсивністю звернень. Повільніші або архівні сховища можуть бути економічно вигідними для інформації, яка використовується рідко.

Однак зменшення вартості зберігання не завжди означає зменшення загальних витрат. Для деяких класів хмарного зберігання окремо оплачуються

операції читання, відновлення архівних об’єктів, передавання інформації та дострокове вилучення. Тому система повинна аналізувати повну вартість життєвого циклу даних, а не лише тариф за одиницю дискового простору.

Алгоритм роботи підпрограми багатокритеріального вибору рівня зберігання наведено на рисунку 4.2. Підпрограма отримує характеристики інформаційного об’єкта, прогнозовану кількість звернень, параметри доступних сховищ та обмеження політики. На початковому етапі виключаються рівні, які не забезпечують необхідну затримку, допустимий строк відновлення або вимоги до розміщення інформації. Для решти варіантів обчислюються прогнозовані витрати та показники швидкодії. Далі визначається інтегральна оцінка або множина Парето-оптимальних рішень. Якщо зміна розміщення не забезпечує достатньої переваги, поточний рівень зберігається. Отриманий результат передається до компонента планування міграції.

Під час розроблення необхідно враховувати відмінності між файловим, блоковим та об’єктним зберіганням. Для файлових систем одиницею переміщення може бути окремий файл. Для об’єктних сховищ використовується логічний об’єкт із власним ключем і метаданими. Блокові пристрої потребують іншого механізму перенесення, оскільки їхні дані можуть належати активній файловій системі або базі даних. Тому як основну одиницю оптимізації доцільно розглядати логічний інформаційний об’єкт, доступ до якого контролюється програмним комплексом. Підтримка блокового зберігання може реалізовуватися окремим спеціалізованим адаптером.

Важливо відокремлювати алгоритм визначення доцільного рівня від механізму фізичного переміщення. Оптимізаційний компонент повинен працювати з уніфікованими характеристиками ресурсів і не залежати від конкретного API сховища. Служба міграції, навпаки, повинна враховувати технологічні особливості відповідного постачальника. Наприклад, переміщення об’єкта між класами одного хмарного сервісу може виконуватися засобами самого провайдера. Перенесення між незалежними сховищами може потребувати

копіювання даних, перевірки цілісності та оновлення логічного маршруту доступу.

Для забезпечення коректності необхідно підтримувати єдине джерело відомостей про поточне розташування об’єктів. Якщо програма звертається безпосередньо до фізичних адрес, зміна рівня зберігання може порушити доступ до інформації. Тому пропонується використання логічного ідентифікатора, який не змінюється внаслідок міграції. Відповідний компонент адресації визначає актуальне фізичне розташування та стан об’єкта. Такий підхід потребує надійного зберігання метаданих, оскільки їхня втрата може ускладнити доступ навіть до фізично збереженої інформації.

Алгоритм ініціалізації ієрархічного середовища

Підготовка системи починається з визначення доступних рівнів зберігання. Вхідними даними є конфігурації сховищ, параметри підключення, ресурсні обмеження та правила доступу.

На першому етапі перевіряється наявність необхідних програмних адаптерів. Для кожного сховища встановлюється можливість виконання основних операцій, зокрема отримання метаданих, читання, записування та видалення об’єктів.

Далі визначаються характеристики ресурсу. До них належать загальна й доступна місткість, тип інтерфейсу, підтримувані режими зберігання та встановлені обмеження. Для хмарного сервісу додатково враховуються регіон розміщення та особливості відповідного класу доступу.

Після цього виконується перевірка з’єднання. Успішна автентифікація не повинна вважатися підтвердженням можливості виконання всіх операцій. Права читання й записування необхідно перевіряти окремо в межах дозволеної тестової області.

Наступним етапом є завантаження тарифної моделі. Для неї встановлюються одиниці вимірювання, період дії та джерело отримання значень. Некоректні або невідомі тарифи не повинні замінюватися довільними числовими значеннями.

Результатом алгоритму є реєстр доступних рівнів зберігання з підтвердженими характеристиками та станом готовності.

Алгоритм реєстрації інформаційних об’єктів

Для керування розміщенням необхідно створити структурований каталог інформації. Вхідними даними є ідентифікатор об’єкта, його фізичне розташування, розмір і службові характеристики.

На початковому етапі визначається логічний ідентифікатор. Він повинен забезпечувати однозначне звернення до об’єкта незалежно від обраного рівня зберігання.

Далі реєструється поточна версія даних. Для об’єктних сховищ доцільно використовувати підтримувані механізми версіонування. Якщо вони недоступні, система може підтримувати власний ідентифікатор версії в метаданих.

Окремо зберігається контрольне значення вмісту. Для підтвердження цілісності може використовуватися SHA-256. Необхідно враховувати, що службове значення ETag у хмарному сервісі не завжди є безпосереднім криптографічним хешем вмісту.

Після реєстрації визначається політика об’єкта. Вона може містити мінімальний строк зберігання, вимоги до доступності, конфіденційності та максимально допустимої затримки.

Якщо об’єкт має активні посилання з інших компонентів, система повинна забезпечити їхню працездатність після можливого перенесення.

Результатом алгоритму є узгоджений каталог об’єктів, необхідний для подальшого оптимізаційного аналізу.

Алгоритм збирання статистики звернень

Для визначення доцільного розташування інформації необхідно враховувати фактичну активність її використання. Вхідними даними є повідомлення про операції читання, записування та отримання метаданих.

На першому етапі для кожної операції визначається логічний ідентифікатор об’єкта. Далі фіксуються час звернення, тип операції, обсяг переданих даних і результат.

Окремо реєструється тривалість обслуговування запиту. Для її вимірювання доцільно використовувати монотонне джерело часу. Це дозволяє уникати помилок, пов’язаних зі зміною системного календарного часу.

Події можуть надходити з компонента логічного доступу або з доступних механізмів моніторингу сховища. При цьому необхідно враховувати, що агреговані метрики постачальника не завжди дозволяють визначити активність кожного окремого об’єкта.

Після накопичення подій система формує інтервальні характеристики. Для них обчислюється кількість звернень, обсяг читання та записування, а також статистика затримок.

Результатом алгоритму є історія використання об’єктів, яка може застосовуватися для класифікації та прогнозування.

Алгоритм визначення інтенсивності використання даних

Для розподілення об’єктів між рівнями необхідно визначити їхню поточну активність. Проста кількість звернень не завжди відображає фактичне навантаження, оскільки операції можуть суттєво відрізнятися за розміром.

Вхідними даними є статистика звернень за визначений часовий інтервал. На першому етапі обчислюється частота операцій читання й записування.

Далі визначається обсяг інформації, переданої під час відповідних операцій. Це дозволяє відокремити об’єкти з великою кількістю невеликих запитів від об’єктів, які рідко читаються, але потребують передавання значного обсягу даних.

Окремо враховується час останнього звернення. Об’єкт, який активно використовувався раніше, але тривалий час не отримував запитів, може бути кандидатом на перенесення до економнішого рівня.

Для зменшення впливу короткочасних коливань доцільно застосовувати ковзні часові вікна. За потреби використовуються характеристики за декілька періодів.

Результатом є вектор інтенсивності використання, який характеризує поточний режим роботи з об’єктом.

Алгоритм класифікації даних за температурою доступу

Одним зі способів попереднього групування інформації є класифікація за температурою доступу. Вона дозволяє визначати об’єкти, які доцільно розглядати для швидких або економніших рівнів зберігання.

Вхідними даними є інтенсивність читання, частота модифікацій, час останнього звернення та характеристики розміру.

На початковому етапі обчислюється інтегральний показник активності. Його складові нормалізуються, оскільки кількість звернень і обсяг даних мають різні одиниці вимірювання.

Далі об’єкти можуть поділятися на гарячі, теплі та холодні. Гарячими вважаються дані з високою інтенсивністю доступу. Теплі об’єкти характеризуються помірним або нерівномірним використанням. Холодні дані мають невелику частоту звернень протягом визначеного періоду.

Важливо, що класифікація не повинна автоматично визначати фізичний рівень розташування. Навіть холодний об’єкт може вимагати швидкого доступу через нормативні або прикладні обмеження.

Тому отримана категорія використовується як одна з ознак для багатокритеріального оцінювання.

Результатом алгоритму є попередня класифікація інформаційних об’єктів за характером доступу.

Алгоритм прогнозування майбутнього навантаження

Розміщення даних лише за поточною активністю може призводити до частих переміщень. Для зменшення таких ситуацій доцільно оцінювати очікувану кількість звернень протягом наступного періоду.

Вхідними даними є часові ряди операцій читання та записування. На першому етапі виконується перевірка їхньої повноти та часової послідовності.

Для початкового прогнозування може використовуватися експоненційне згладжування. Воно дозволяє надавати більшу вагу останнім спостереженням і поступово враховувати зміну інтенсивності доступу.

Якщо навантаження має добові або тижневі закономірності, доцільно застосовувати модель із відповідними сезонними характеристиками. Для складніших залежностей можуть використовуватися алгоритми машинного навчання.

Вхідними ознаками такої моделі можуть бути кількість звернень за попередні періоди, час доби, день тижня та категорія об’єкта. При цьому необхідно виключити використання майбутніх спостережень під час навчання.

Для перевірки прогнозування застосовується хронологічний поділ даних. Якість може оцінюватися за Mean Absolute Error або Root Mean Squared Error.

Якщо історії недостатньо, система повинна використовувати консервативну оцінку або зберігати поточне розміщення.

Результатом алгоритму є прогнозований профіль звернень на вибраному горизонті оптимізації.

Алгоритм оцінювання вартості зберігання

Економічна модель повинна враховувати всі суттєві складові витрат, пов’язані з розташуванням інформаційного об’єкта.

Вхідними даними є розмір об’єкта, прогнозована активність, тарифи сховищ і вибраний горизонт розрахунку.

На першому етапі визначається базова вартість зберігання. Для хмарного ресурсу вона може розраховуватися за тарифом на одиницю обсягу протягом відповідного періоду.

Далі обчислюються витрати на операції. Залежно від класу сховища вони можуть включати оплату запитів, читання певного обсягу даних і відновлення архівної інформації.

Окремо необхідно враховувати передавання між регіонами або незалежними сервісами. Для деяких конфігурацій такі операції можуть формувати додаткові витрати.

Далі оцінюється вартість можливого перенесення. Вона може складатися з операцій копіювання, передавання, тимчасового дублювання об’єкта та зміни класу зберігання.

Для архівних і низькочастотних класів можуть діяти мінімальні оплачувані строки. Перенесення раніше встановленого терміну здатне спричинити додаткові нарахування.

Загальна прогнозована вартість для об’єкта (i) та рівня (s) може бути представлена як

[

C_{i,s}=C^{store}{i,s}+C^{request}{i,s}+C^{retrieve}{i,s}+C^{transfer}{i,s}+C^{migration}{i,s}+C^{penalty}{i,s}.

]

У цій моделі складові відповідають витратам на зберігання, запити, отримання даних, передавання, міграцію та додаткові тарифні нарахування.

Результатом алгоритму є оцінка повної вартості розміщення об’єкта на кожному допустимому рівні.

Алгоритм оцінювання швидкодії доступу

Другим основним критерієм оптимізації є швидкодія. Її необхідно оцінювати за характеристиками, які відповідають фактичному способу використання інформації.

Вхідними даними є статистика часу обслуговування запитів, характеристика навантаження та параметри доступних сховищ.

На першому етапі визначається середня затримка читання. Для навантажень із вимогами до стабільності часу відповіді додатково використовуються перцентилі P95 або P99.

Далі аналізується пропускна здатність. Для об’єктів великого розміру може бути важливішою швидкість передавання, ніж час виконання окремого запиту.

Окремо враховуються архівні рівні. Вони можуть передбачати попереднє відновлення об’єкта, через що затримка отримання вимірюється значно більшими часовими інтервалами, ніж для звичайних оперативних сховищ.

Для оцінювання нового рівня можуть використовуватися результати контрольних вимірювань на порівнянних об’єктах. За відсутності таких даних система повинна позначати прогноз як недостатньо підтверджений.

Важливо не ототожнювати середню затримку з P95. Вони характеризують різні властивості розподілу часу обслуговування.

Результатом алгоритму є прогнозовані показники швидкодії для доступних варіантів розміщення.

Алгоритм формування множини допустимих рівнів

Перед багатокритеріальним порівнянням необхідно виключити сховища, які не відповідають обов'язковим вимогам.

Вхідними даними є характеристики об'єкта, політика розміщення та відомості про доступні ресурси.

На першому етапі перевіряється підтримка необхідних операцій. Якщо об'єкт повинен змінюватися безпосередньо під час роботи додатка, архівний рівень із попереднім відновленням може бути непридатним.

Далі перевіряється максимально допустима затримка. Рівень, для якого прогнозована затримка перевищує встановлену межу, виключається з подальшого вибору.

Окремо контролюються обмеження доступності, резервування, конфіденційності та географічного розміщення.

Після цього визначається достатність вільної місткості. Для міграції необхідно враховувати не лише розмір об'єкта, а й тимчасовий обсяг, потрібний для перевірки результату.

Якщо для певного класу зберігання діють мінімальні строки перебування, система перевіряє економічні наслідки подальшого переміщення.

Результатом алгоритму є множина рівнів, які відповідають обов'язковим технічним і експлуатаційним умовам.

Алгоритм багатокритеріальної оптимізації розміщення

Модель вибору рівня повинна враховувати суперечливість основних критеріїв. Мінімальна вартість зберігання може відповідати низькій швидкодії, а найменша затримка доступу може потребувати використання дорожчих ресурсів.

Вхідними даними є множина допустимих рівнів, прогнозовані витрати та характеристики продуктивності.

Для порівняння різнорідних показників використовується нормалізація. Вартість може вимірюватися в грошових одиницях, а затримка в мілісекундах. Без приведення до порівнянного масштабу ці характеристики некоректно використовувати в одній функції.

Після нормалізації може застосовуватися зважена модель

[

J_{i,s}=\lambda_C\widehat{C}{i,s}+\lambda_L\widehat{L}{i,s}+\lambda_M\widehat{M}_{i,s},

]

де (\widehat{C}{i,s}) є нормалізованою вартістю, (\widehat{L}{i,s}) характеризує затримку доступу, а (\widehat{M}{i,s}) відображає додаткові витрати або небажаність міграції. Коефіцієнти (\lambda_C), (\lambda_L) та (\lambda_M) визначають відносну значущість критеріїв.

Якщо вартість міграції вже включена до загальної вартості (C{i,s}), додатковий критерій (\widehat{M}_{i,s}) повинен відображати іншу характеристику, наприклад тривалість перенесення або операційний ризик. Інакше одна складова буде врахована двічі.

Вагові коефіцієнти можуть встановлюватися адміністратором відповідно до вимог системи. Для критичних даних більшу увагу доцільно приділяти затримці та доступності. Для довготривалого архівного зберігання може переважати економічний критерій.

Як альтернативний підхід використовується визначення множини Парето-оптимальних рішень. Варіант вважається домінованим, якщо інший допустимий рівень не гірший за всіма критеріями та кращий хоча б за одним.

Після формування множини недомінованих варіантів остаточний вибір виконується з урахуванням установлених пріоритетів.

Результатом алгоритму є рекомендований рівень, який відповідає обмеженням і прийнятій моделі багатокритеріальної оптимізації.

Алгоритм глобального розподілення об’єктів

Оптимізація кожного об’єкта незалежно не завжди забезпечує найкращий результат для всієї системи. Якщо велика кількість об’єктів одночасно потребує швидкого рівня, його місткість може бути недостатньою.

Тому необхідно враховувати спільні ресурсні обмеження.

Вхідними даними є перелік інформаційних об’єктів, їхні допустимі рівні, прогнозовані характеристики та доступна місткість сховищ.

Для кожного об’єкта формується набір можливих рішень. Після цього визначається економічний ефект від розміщення на певному рівні.

Задачу можна представити через бінарні змінні (x_{i,s}), які дорівнюють одиниці, якщо об’єкт (i) розміщується на рівні (s).

Для кожного об’єкта необхідно виконувати умову

[\sum_s x_{i,s}=1.]

Обмеження місткості рівня записується як

[\sum_i V_i x_{i,s}\leq K_s,]

де (V_i) визначає необхідний фізичний обсяг зберігання об’єкта, а (K_s) відповідає доступній місткості рівня.

За необхідності модель доповнюється обмеженнями на вартість, пропускну здатність і допустиму кількість міграцій.

Для невеликих наборів може використовуватися цілочисельна оптимізація. Для великих каталогів доцільно розглядати евристичні алгоритми, які ранжують об’єкти за очікуваним ефектом перенесення.

Результатом є план розміщення, який враховує обмежені ресурси всієї інфраструктури.

Алгоритм визначення доцільності міграції

Навіть якщо інший рівень має кращу оптимізаційну оцінку, переміщення об’єкта може бути недоцільним через разові витрати та ризик порушення доступності.

Вхідними даними є поточний рівень, рекомендований варіант, вартість перенесення та прогнозований строк використання даних.

На першому етапі визначається економічна різниця між поточним і новим розміщенням. Далі враховуються додаткові витрати на міграцію.

Якщо очікувана економія не компенсує необхідних витрат протягом обраного горизонту, система може залишити поточне розташування.

Окремо оцінюється зміна швидкодії. Перенесення до дорожчого рівня може бути виправданим, якщо поточне сховище не відповідає встановленим вимогам до затримки.

Для зменшення частоти повторних переміщень доцільно використовувати гістерезис. Він передбачає встановлення мінімального виграшу, необхідного для зміни рівня.

Додатково може застосовуватися мінімальний період перебування об’єкта на одному рівні. Винятки допускаються для випадків порушення критичних вимог доступу.

Результатом алгоритму є рішення про виконання міграції або збереження поточного розташування.

Алгоритм планування черги міграцій

Після формування кандидатів необхідно визначити послідовність їхнього перенесення. Одночасна міграція великої кількості об’єктів може перевантажувати мережу та дискову підсистему.

Вхідними даними є перелік рекомендованих переміщень, оцінка їхнього ефекту, розмір об’єктів і доступні ресурси.

На першому етапі визначаються пріоритети. Підвищену важливість можуть отримувати об’єкти, поточне розміщення яких порушує встановлені вимоги до доступу.

Для економічних міграцій може використовуватися співвідношення очікуваної економії до обсягу переданих даних.

Далі враховується стан мережевих каналів. Якщо інтенсивність користувацьких операцій зростає, фонові переміщення доцільно обмежувати.

Після цього встановлюється максимальна кількість паралельних операцій. Вона залежить від можливостей вихідного та цільового сховищ.

Окремо перевіряється доступний простір на цільовому рівні. Необхідно резервувати місткість з урахуванням тимчасового існування декількох копій даних.

Результатом алгоритму є впорядкована черга міграцій, узгоджена з ресурсними обмеженнями.

Алгоритм безпечного перенесення інформаційного об’єкта

Міграція повинна забезпечувати збереження вмісту та можливість відновлення доступу до інформації. Для цього доцільно використовувати послідовність керованих станів.

Вхідними даними є ідентифікатор об’єкта, його поточна версія, вихідне сховище та цільовий рівень.

На першому етапі перевіряється актуальність рішення. Якщо після його формування змінилася версія об’єкта, план перенесення необхідно переглянути.

Далі створюється запис операції міграції. Він містить унікальний ідентифікатор, вихідний і цільовий рівні, поточний стан та контрольні характеристики даних.

Після цього виконується підготовка цільового сховища. Перевіряються необхідні права доступу, доступна місткість та можливість записування.

Наступним етапом є копіювання. За наявності підтримки відповідного API може використовуватися серверне копіювання. Для незалежних сховищ може знадобитися передавання через окремий програмний компонент.

Після завершення передавання виконується перевірка цілісності. Для цього порівнюються контрольні значення або застосовується інша підтверджена процедура перевірки.

Якщо дані відповідають очікуваній версії, система переходить до оновлення логічного розташування. Відповідний запис у сховищі метаданих змінюється лише за умови, що початкова версія об’єкта залишається актуальною.

Після успішного перемикання нові запити спрямовуються до цільового рівня. Попередня копія тимчасово зберігається відповідно до встановленої політики відновлення.

Видалення вихідних даних допускається лише після підтвердження успішної міграції та перевірки відсутності обмежень на їхнє збереження.

Результатом алгоритму є перенесений об’єкт з актуалізованим логічним розташуванням і підтвердженою цілісністю.

Алгоритм забезпечення узгодженості під час міграції

Особливу складність становить перенесення об’єктів, які можуть змінюватися під час роботи прикладних програм.

Вхідними даними є поточний стан міграції, версія об’єкта та повідомлення про операції записування.

На першому етапі визначається, чи допускається модифікація під час перенесення. Для незмінних об’єктів достатньо контролю вихідної версії.

Для змінюваних даних необхідно передбачити додатковий механізм. Одним із варіантів є тимчасове обмеження записування на період завершального перемикання.

Іншим способом може бути використання версій. У цьому випадку копіюється визначена версія, а нові зміни створюють наступний логічний стан об’єкта.

Після завершення копіювання система повторно перевіряє актуальність вихідної версії. Якщо вона змінилася, підготовлену копію не можна безумовно оголошувати основною.

Для атомарного оновлення метаданих може використовуватися транзакція або умовна операція порівняння й заміни.

Необхідно враховувати, що копіювання між незалежними хмарними сервісами саме по собі не забезпечує єдиної транзакції. Тому узгодженість повинна підтримуватися прикладним компонентом.

Результатом алгоритму є контрольований перехід між рівнями без втрати актуальної версії даних.

Алгоритм доступу до об’єкта через логічну адресацію

Для користувацьких програм переміщення даних повинно бути максимально відокремлене від способу звернення до об’єкта.

Вхідними даними є логічний ідентифікатор, тип запиту та повноваження користувача.

На першому етапі перевіряються права доступу. Далі виконується пошук актуального запису в каталозі метаданих.

Якщо об’єкт перебуває в стабільному стані, запит передається до відповідного сховища.

Під час міграції система повинна враховувати встановлену політику узгодженості. Для операцій читання може використовуватися підтверджена поточна копія. Операції записування можуть бути тимчасово обмежені або спрямовані до визначеного власника актуальної версії.

Для архівного рівня пряме читання може бути недоступним. У такому випадку формується окрема процедура відновлення.

Якщо метадані відсутні або пошкоджені, система не повинна довільно обирати одну з фізичних копій. Необхідно виконати процедуру звіряння стану.

Результатом алгоритму є правильне спрямування запиту до актуальної версії інформації.

Алгоритм відновлення даних з архівного рівня

Архівні сховища можуть передбачати тривалу процедуру підготовки даних до читання. Тому для них необхідно використовувати окремий алгоритм доступу.

Вхідними даними є ідентифікатор архівного об’єкта, запит на його отримання та вимоги користувача до строку доступності.

На першому етапі визначається можливість безпосереднього читання. Якщо архівний клас не підтримує відповідну операцію, формується завдання відновлення.

Далі оцінюються прогнозована тривалість та додаткові витрати. За наявності декількох режимів відновлення вибір виконується з урахуванням дозволеного бюджету й терміновості запиту.

Після запуску операції система зберігає її ідентифікатор і поточний стан. У разі асинхронного відновлення користувацькому компоненту повертається повідомлення про необхідність очікування.

Після отримання підтвердження готовності перевіряється доступність відновлених даних. Далі запит обробляється відповідно до встановлених правил.

Слід враховувати, що архівний рівень не повинен обиратися для об’єктів, які мають вимогу негайного отримання, якщо така можливість не забезпечується.

Результатом алгоритму є контрольоване відновлення архівної інформації з урахуванням часу та вартості операції.

Алгоритм керування кешуванням часто використовуваних даних

Для скорочення затримки звернень може застосовуватися додатковий рівень кешування. Його використання доцільне за наявності повторних операцій читання.

Вхідними даними є статистика звернень, розмір об’єкта, місткість кешу та вимоги до узгодженості.

На першому етапі визначається очікувана повторюваність доступу. Далі оцінюється економічна та продуктивнісна користь розміщення копії в швидкому сховищі.

Якщо користь перевищує витрати, формується рішення про кешування. При цьому оригінальний об’єкт залишається на основному рівні відповідно до політики зберігання.

Під час читання система спочатку перевіряє наявність актуальної кешованої версії. Якщо її немає, запит передається до основного сховища.

Після модифікації об’єкта необхідно оновити або визнати недійсною відповідну кешовану копію.

Для звільнення місткості можуть використовуватися алгоритми LRU або LFU. Вони забезпечують різні стратегії витіснення за часом останнього звернення або частотою використання.

Окремо необхідно враховувати, що вбудований механізм cache tiering у Ceph вважається застарілим і не рекомендований для нових розгортань. Тому за використання Ceph кешування доцільно проєктувати як незалежний прикладний механізм або застосовувати актуальні підтримувані засоби.

Результатом алгоритму є контрольоване розміщення часто використовуваних копій у швидкому кешувальному просторі.

Приклад програмної реалізації вибору рівня зберігання

Для пояснення принципу багатокритеріального оцінювання можна використати програмний алгоритм мовою Python. Наведений фрагмент демонструє вибір економічно доцільного рівня за умови дотримання обмеження на прогнозовану затримку читання. У моделі враховуються вартість місткості, операцій читання, отримання даних і разова вартість міграції.

Програмний приклад є ілюстративним. Він не підтверджує використання саме такого коду у фактичній реалізації системи та не містить тарифів конкретних постачальників хмарних послуг.

```
from dataclasses import dataclass
from math import isfinite


@dataclass(frozen=True)
class DataObject:
    # Характеристики інформаційного об'єкта
    object_id: str
    size_gb: float
    expected_reads: int
    average_read_gb: float
    max_p95_ms: float
    current_tier: str


@dataclass(frozen=True)
class StorageTier:
    # Параметри рівня зберігання
    name: str
    storage_price_gb_month: float
    request_price_per_1000: float
    retrieval_price_gb: float
    predicted_p95_ms: float
    online_access: bool
    permitted: bool


@dataclass(frozen=True)
class OptimizationPolicy:
    # Горизонт оцінювання та мінімальна економія
    horizon_months: float
    min_savings: float
    migration_price_gb: float
    migration_allowed: bool


def estimate_cost(
    obj: DataObject,
    tier: StorageTier,
    policy: OptimizationPolicy,
```

```
    migrating: bool
) -> float:
    """Оцінювання витрат за спрощеною тарифною моделлю."""

    # Оплата зберігання протягом горизонту
    storage_cost = (
        obj.size_gb
        * tier.storage_price_gb_month
        * policy.horizon_months
    )

    # Кількість запитів задається для всього горизонту
    request_cost = (
        obj.expected_reads / 1000.0
        * tier.request_price_per_1000
    )

    # Прогнозований обсяг читання
    retrieval_cost = (
        obj.expected_reads
        * obj.average_read_gb
        * tier.retrieval_price_gb
    )

    # Разові витрати на перенесення об'єкта
    migration_cost = (
        obj.size_gb * policy.migration_price_gb
        if migrating
        else 0.0
    )

    return (
        storage_cost
        + request_cost
        + retrieval_cost
        + migration_cost
    )


def select_storage_tier(
    obj: DataObject,
    tiers: list[StorageTier],
    policy: OptimizationPolicy
) -> dict:
    """Вибір рівня з урахуванням вартості та затримки."""

    # Перевіряємо коректність вхідних даних
    values = (
        obj.size_gb,
        obj.average_read_gb,
        obj.max_p95_ms,
        policy.horizon_months,
        policy.min_savings,
        policy.migration_price_gb
    )

    if (
        not all(isfinite(value) for value in values)
        or obj.size_gb < 0
        or obj.expected_reads < 0
        or obj.average_read_gb < 0
        or obj.max_p95_ms <= 0
        or policy.horizon_months <= 0
        or policy.min_savings < 0
```

```
    or policy.migration_price_gb < 0
):
    return {"status": "INVALID_INPUT"}

current = next(
    (tier for tier in tiers if tier.name == obj.current_tier),
    None
)

if current is None:
    return {"status": "CURRENT_TIER_UNKNOWN"}

candidates = []

for tier in tiers:
    # Рівень повинен бути дозволеним і підтримувати
    # необхідний оперативний доступ до даних
    if not tier.permitted or not tier.online_access:
        continue

    # Перевіряємо тарифні параметри та прогноз
    tier_values = (
        tier.storage_price_gb_month,
        tier.request_price_per_1000,
        tier.retrieval_price_gb,
        tier.predicted_p95_ms
    )

    if not all(isfinite(value) and value >= 0
               for value in tier_values):
        continue

    # Виключаємо рівні з надмірною затримкою
    if tier.predicted_p95_ms > obj.max_p95_ms:
        continue

    migrating = tier.name != obj.current_tier

    if migrating and not policy.migration_allowed:
        continue

    cost = estimate_cost(obj, tier, policy, migrating)

    candidates.append((cost, tier.name))

if not candidates:
    return {
        "status": "NO_FEASIBLE_TIER",
        "selected_tier": obj.current_tier
    }

# Обираємо найдешевший допустимий варіант
candidates.sort()
best_cost, best_tier = candidates[0]

# Оцінюємо чинне розміщення за тим самим горизонтом
current_cost = estimate_cost(obj, current, policy, False)

current_feasible = (
    current.permitted
    and current.online_access
    and current.predicted_p95_ms <= obj.max_p95_ms
)
```

```
savings = current_cost - best_cost

# Уникаємо міграції за незначного виграшу
if (
    current_feasible
    and best_tier != obj.current_tier
    and savings < policy.min_savings
):
    return {
        "status": "NO_CHANGE",
        "selected_tier": obj.current_tier,
        "estimated_cost": current_cost
    }

return {
    "status": "PROPOSED",
    "selected_tier": best_tier,
    "estimated_cost": best_cost,
    "estimated_savings": savings
}
```

У наведеному прикладі спочатку перевіряється коректність характеристик об’єкта та оптимізаційної політики. Далі формується множина допустимих онлайн-рівнів зберігання. Для кожного варіанта обчислюється прогнозована вартість з урахуванням визначеного горизонту та можливих витрат на міграцію.

Рівні, які не відповідають вимогам до затримки, виключаються. Серед решти обирається варіант із найменшою розрахунковою вартістю. Якщо очікувана економія не перевищує встановленого порога, поточне розміщення зберігається.

Якщо чинний рівень порушує встановлене обмеження затримки, система може запропонувати дорожчий, але технічно придатний варіант. Таким чином швидкодія використовується як обов’язкове обмеження, а вартість визначає вибір серед допустимих рішень.

Наведена функція не виконує фізичного перенесення. Вона формує лише рекомендацію. Перед практичним застосуванням необхідно доповнити модель перевіркою місткості, вартості передавання між регіонами, мінімальних оплачуваних строків, фактичних умов API та дозволів на зміну розташування.

Для архівних класів потрібний окремий механізм оцінювання, оскільки їхня затримка та порядок відновлення можуть не відповідати моделі звичайного онлайн-доступу.

Алгоритм контролю повторних міграцій

За змінного навантаження об’єкт може періодично переходити між категоріями гарячих і холодних даних. Якщо реагувати на кожну короткочасну зміну, система виконуватиме надмірну кількість переміщень.

Вхідними даними є історія міграцій, поточні характеристики об’єкта та нова рекомендація оптимізаційної моделі.

На першому етапі визначається час останньої зміни рівня. Якщо відтоді минуло недостатньо часу, повторне перенесення відкладається.

Далі аналізується стійкість прогнозованого навантаження. Для цього можуть використовуватися декілька послідовних часових вікон.

Якщо зміна активності має короткочасний характер, система може зберігати попереднє розташування.

Окремо встановлюється мінімальна економія, необхідна для нового переміщення. Вона повинна враховувати фактичні витрати попередніх операцій.

Для випадків порушення вимог до доступності або швидкодії можуть застосовуватися винятки. Однак такі рішення також повинні проходити перевірку технічної допустимості.

Результатом алгоритму є обмеження зайвих міграцій і зменшення нестабільності розміщення даних.

Алгоритм актуалізації тарифів і характеристик сховищ

Результати оптимізації залежать від правильності вхідних характеристик. Тарифи хмарних сервісів, обмеження ресурсів і фактична продуктивність можуть змінюватися.

Вхідними даними є зареєстровані параметри сховищ, доступні тарифні відомості та результати моніторингу.

На першому етапі визначається актуальність чинної конфігурації. Якщо її строк дії завершився, система формує запит на оновлення.

Далі перевіряються одиниці вимірювання та правила розрахунку. Наприклад, тариф за одиницю обсягу протягом місяця не можна безпосередньо порівнювати з вартістю операцій за тисячу запитів.

Після отримання нових значень виконується їхня перевірка. Якщо частина тарифів відсутня, відповідний ресурс може бути тимчасово виключений із фінансової оптимізації.

Окремо оновлюються показники продуктивності. Значні відхилення від попередніх вимірювань можуть потребувати повторного профілювання.

Після перевірки створюється нова версія параметрів. Завдяки цьому попередні рішення можна аналізувати з урахуванням тарифів, які діяли на момент їхнього формування.

Результатом алгоритму є актуальна модель ресурсів, придатна для подальших оптимізаційних розрахунків.

Алгоритм оброблення помилок і відмов компонентів

Під час функціонування можливі недоступність сховища, пошкодження метаданих, помилки передавання та порушення доступності аналітичного компонента.

Для кожної ситуації необхідно визначити допустиму реакцію.

Якщо оптимізаційний модуль тимчасово недоступний, система повинна продовжувати обслуговування запитів за поточними метаданими без виконання нових міграцій.

За відмови цільового сховища під час перенесення операція позначається як незавершена. Вихідна копія повинна залишатися доступною, якщо її стан підтверджений.

Якщо копіювання завершилося, але оновлення метаданих не відбулося, система повинна перевірити фактичний стан обох копій перед повторною спробою.

Для запобігання конфліктам кожна міграція повинна мати унікальний ідентифікатор. Повторне надходження того самого запиту не повинно створювати неконтрольовану кількість копій.

У разі втрати доступу до каталогу метаданих нові операції зміни розташування необхідно призупинити. Відновлення може виконуватися з резервної копії та журналу завершених операцій.

Результатом алгоритму є контрольоване відновлення після відмов без безумовного видалення даних із невизначеним станом.

Особливості програмної реалізації системи

Для реалізації керувального компонента може використовуватися Python. Він забезпечує засоби роботи з програмними API, оброблення статистичних даних і реалізації оптимізаційних алгоритмів.

Для взаємодії зі сховищами доцільно передбачити окремі адаптери. Вони повинні реалізовувати узгоджений набір операцій читання, записування, отримання метаданих і копіювання.

За використання об’єктного зберігання можуть застосовуватися відповідні клієнтські бібліотеки хмарних провайдерів. Конкретний набір залежить від фактично обраного середовища.

Для збереження каталогу об’єктів і станів міграцій може використовуватися PostgreSQL. Особливе значення має транзакційне оновлення логічного розташування.

Для накопичення телеметрії можуть застосовуватися окремі служби збору подій і сховище часових рядів.

Багатокритеріальна модель може реалізовуватися власним алгоритмом перебору допустимих варіантів. Для глобального розподілення об’єктів можуть використовуватися відповідні бібліотеки математичної оптимізації.

Архітектура повинна забезпечувати незалежність логіки вибору рівня від способу фізичного перенесення. Це дозволяє змінювати набір сховищ без повного перероблення алгоритмів оцінювання.

Забезпечення інформаційної безпеки та надійності

Ієрархічне зберігання повинно враховувати вимоги до конфіденційності, цілісності та доступності інформації.

Для кожного рівня необхідно визначати дозволені категорії даних. Якщо політика забороняє передавання певної інформації за межі визначеного середовища, відповідний хмарний ресурс повинен виключатися з множини допустимих варіантів.

Під час передавання доцільно використовувати захищені мережеві з’єднання. За необхідності застосовується шифрування збережених даних.

Повноваження оптимізаційного компонента та служби міграції необхідно розмежовувати. Аналітична підсистема може працювати з телеметрією без права видалення фізичних об’єктів.

Окремо контролюється цілісність інформації після перенесення. Успішна відповідь API не повинна автоматично замінювати перевірку фактичної доступності нової копії.

Для службового каталогу необхідно передбачити резервне копіювання. Його втрата може призвести до порушення логічної адресації.

Результатом відповідних механізмів є контрольоване керування розміщенням із дотриманням установлених правил захисту інформації.

Перевірка коректності алгоритмів та оцінювання ефективності

Для експериментального дослідження запропонованої системи необхідно перевірити правильність класифікації даних, точність прогнозування, доцільність оптимізаційних рішень та надійність міграції.

На початковому етапі формується контрольований набір інформаційних об’єктів із різними характеристиками використання. Для цього можуть застосовуватися синтетичні дані, які відрізняються розміром, інтенсивністю читання та кількістю операцій записування.

Далі перевіряється реєстрація об’єктів і правильність формування метаданих. Для кожного файлу визначається очікуване контрольне значення, яке порівнюється з результатом програмного комплексу.

Наступним етапом є перевірка збирання статистики. Для цього генерується визначена кількість звернень до об’єктів. Необхідно підтвердити правильність обчислення частоти запитів і переданого обсягу.

Для оцінювання класифікації використовуються контрольовані сценарії з високою, середньою та низькою інтенсивністю доступу. Важливо встановити, чи правильно система відносить відповідні об’єкти до визначених категорій.

Окремо перевіряється алгоритм прогнозування. Для цього часові ряди розділяються на навчальний і тестовий періоди. Результати можуть оцінюватися за допомогою Mean Absolute Error та Root Mean Squared Error.

Для перевірки економічної моделі формуються умовні тарифи різних рівнів. На основі відомих значень вручну обчислюється очікувана вартість розміщення, після чого вона порівнюється з результатом програмного алгоритму.

Додатково необхідно перевірити врахування вартості міграції та мінімальних оплачуваних строків. За недостатнього економічного ефекту система повинна зберігати поточне розташування.

Для дослідження багатокритеріальної оптимізації доцільно використовувати сценарії з різними пріоритетами. В одному випадку основним критерієм може бути зменшення витрат, в іншому перевага надається швидкодії.

Під час перевірки необхідно підтвердити, що алгоритм не обирає рівні, які порушують обов’язкові обмеження. Вибір найдешевшого, але технічно непридатного сховища повинен вважатися помилкою.

Окремо досліджується глобальний розподіл об’єктів. Для цього встановлюється обмежена місткість швидкого рівня. Оптимізаційний компонент повинен враховувати конкуренцію між об’єктами за доступний простір.

Для оцінювання міграції використовується контрольований набір файлів. Після копіювання перевіряється збіг контрольних значень і правильність оновлення логічного розташування.

Важливим є тестування операцій записування під час перенесення. Для цього створюється сценарій зміни об’єкта після початку копіювання. Система повинна запобігати перемиканню на застарілу версію.

Окремо перевіряється відновлення після відмов. Для цього можуть моделюватися переривання передавання, тимчасова недоступність сховища та помилка оновлення метаданих.

Очікуваним результатом є можливість повторної обробки міграції без втрати актуальних даних та без необґрунтованого видалення початкової копії.

Для оцінювання продуктивності можуть використовуватися середня затримка читання, P95, пропускна здатність і кількість операцій за секунду.

Експериментальне порівняння доцільно виконувати між фіксованим розміщенням, простими правилами перенесення за температурою доступу та запропонованою багатокритеріальною моделлю.

Під час аналізу необхідно враховувати як витрати на зберігання, так і вартість операцій читання, передавання та міграції. Порівняння лише тарифів за місткість не дозволяє достовірно оцінити економічний результат.

Додатково можуть визначатися кількість міграцій, загальний обсяг перенесених даних і час, необхідний для завершення оптимізаційного циклу.

Важливим показником є частка запитів, які відповідають установленим вимогам до затримки. Для архівних об’єктів необхідно окремо оцінювати час відновлення.

Для перевірки стабільності використовуються сценарії з періодичною зміною активності. Необхідно встановити, чи запобігають гістерезис і мінімальні строки перебування надмірним переміщенням.

Розглянуте алгоритмічне забезпечення формує послідовний процес інтелектуального керування ієрархічним зберіганням даних. Основними його складовими є збирання статистики використання, прогнозування навантаження, оцінювання повної вартості, багатокритеріальний вибір рівня та контрольована міграція інформаційних об’єктів. Поєднання економічних і продуктивнісних характеристик дозволяє визначати допустимі варіанти розміщення відповідно до встановлених вимог. Запропоновані алгоритми створюють основу для програмної реалізації системи, ефективність якої повинна підтверджуватися результатами експериментального дослідження.

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Програмне забезпечення системи ієрархічного зберігання даних призначене для автоматизованого керування розміщенням інформаційних об’єктів між сховищами з різними характеристиками вартості та швидкодії. Система забезпечує облік доступних ресурсів, аналіз інтенсивності звернень, визначення економічно доцільного рівня зберігання та контроль перенесення даних. Використання моделей багатокритеріальної оптимізації дозволяє враховувати витрати на зберігання, затримку доступу й обмеження доступності інформації. Головне вікно програми показане на рисунку 5.1.

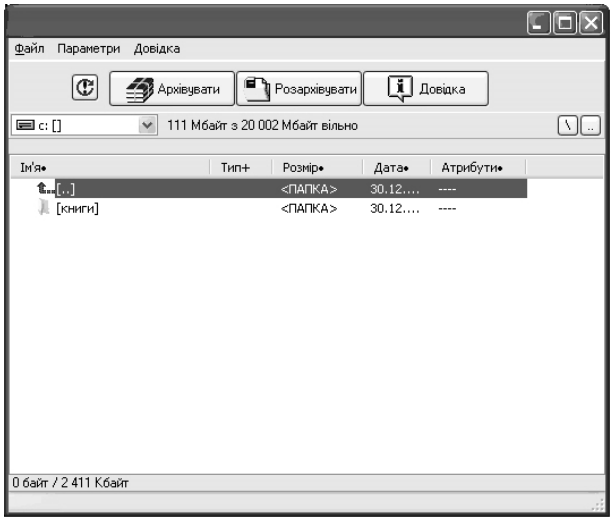


Рисунок 5.1 – Головне вікно розробленого ПЗ

Для впровадження програмного комплексу необхідно підготувати середовище виконання, забезпечити підключення до сховищ і налаштувати компоненти керування даними.

Запропонована архітектура передбачає модуль реєстрації інформаційних об’єктів, підсистему збору статистики, компонент оптимізації, службу міграції та сховище метаданих.

Окремий компонент логічної адресації забезпечує доступ до об’єктів незалежно від їхнього фізичного розташування.

Вимоги до апаратного забезпечення залежать від кількості інформаційних об’єктів, інтенсивності звернень і способу їхнього передавання. Керувальний сервер повинен мати достатній обсяг оперативної пам’яті для оброблення метаданих та виконання оптимізаційних розрахунків.

Для зберігання журналів і службової інформації необхідно передбачити окремий дисковий простір. Продуктивність мережевих інтерфейсів повинна відповідати очікуваній інтенсивності міграцій між сховищами.

Підготовка середовища починається зі встановлення інтерпретатора Python та необхідних програмних залежностей. Для ізоляції бібліотек доцільно використовувати окреме віртуальне середовище.

Після встановлення перевіряється можливість запуску аналітичного компонента, виконання розрахунків і взаємодії з програмними інтерфейсами сховищ. Точні версії компонентів та команди встановлення повинні відповідати обраному технологічному стеку.

Наступним етапом є підготовка рівнів ієрархічного зберігання. Для експериментального середовища можуть використовуватися локальні накопичувачі SSD і HDD, а також доступні об’єктні сховища.

Для кожного ресурсу необхідно визначити адресу підключення, підтримувані операції, доступну місткість і параметри автентифікації. Якщо використовуються хмарні сервіси, додатково враховуються регіон розміщення та можливості відповідного класу зберігання.

Для взаємодії з фізичними ресурсами передбачається налаштування програмних адаптерів. Вони повинні забезпечувати читання, записування, отримання метаданих і копіювання об’єктів.

Під час підключення необхідно перевірити доступність сховища та наявність дозволів на виконання відповідних операцій. Повноваження читання й записування доцільно перевіряти окремо.

Операції видалення не повинні дозволятися без необхідності, визначеної

механізмом контрольованої міграції.

Для збереження каталогу інформаційних об’єктів як можливий варіант може використовуватися PostgreSQL. У відповідному сховищі необхідно передбачити реєстрацію логічних ідентифікаторів, фізичних адрес, версій об’єктів та станів перенесення.

Додатково зберігаються результати оптимізаційних розрахунків і статистика звернень. Перед запуском програмного комплексу перевіряється доступність бази даних та правильність виконання операцій створення й оновлення службових записів.

Після підключення ресурсів формується їхній реєстр. Для кожного рівня вказуються тарифні характеристики, допустимі затримки доступу та встановлені експлуатаційні обмеження. Якщо система працює з хмарними ресурсами, необхідно враховувати оплату операцій читання, відновлення архівних об’єктів і передавання між сервісами. Значення тарифів повинні відповідати фактичним умовам використання. За відсутності достовірних відомостей відповідний ресурс доцільно виключати з автоматизованого економічного оцінювання.

Окремо налаштовується компонент збору статистики. Для нього визначаються інтервали накопичення показників, правила реєстрації операцій читання й записування та порядок обчислення затримки доступу. Джерелом телеметрії може бути компонент логічної адресації або підтримуваний інтерфейс моніторингу сховища. Під час перевірки необхідно переконатися, що кожне звернення правильно зіставляється з інформаційним об’єктом.

Підготовка оптимізаційного компонента передбачає встановлення параметрів багатокритеріальної моделі. До них належать вагові коефіцієнти вартості та швидкодії, допустима затримка, горизонт прогнозування і мінімальний економічний ефект міграції. Окремо задаються обмеження місткості та правила розміщення конфіденційних даних. Початкові значення параметрів доцільно уточнювати за результатами контрольованих випробувань.

Для зменшення кількості зайвих переміщень встановлюються правила стабілізації рішень. Вони можуть передбачати мінімальний строк перебування

об’єкта на одному рівні та поріг доцільності повторної міграції. Якщо прогнозований економічний ефект не компенсує витрати на перенесення, поточне розташування повинно зберігатися. Обмеження доступності та швидкодії мають перевірятися незалежно від фінансових переваг.

Наступним етапом є налаштування служби міграції. Для неї визначаються допустима кількість одночасних переміщень, граничне використання мережевого каналу та порядок перевірки цілісності. Після копіювання об’єкта необхідно підтвердити правильність його вмісту. Для цього може використовуватися SHA-256. Оновлення логічного розташування допускається лише після успішного завершення копіювання та перевірки актуальності версії.

Рекомендована послідовність запуску починається з перевірки доступності сховища метаданих і фізичних ресурсів. Далі активуються адаптери та компонент логічної адресації. Після цього запускається служба збору статистики. Наступним етапом є завантаження оптимізаційної моделі та активація планувальника міграцій. Завершується підготовка запуском користувацького інтерфейсу й перевіркою стану основних компонентів.

Первинне тестування доцільно виконувати на окремому наборі некритичних інформаційних об’єктів. Спочатку перевіряється їхня реєстрація та можливість читання за логічним ідентифікатором. Далі створюється контрольоване навантаження з різною інтенсивністю звернень. Система повинна правильно формувати статистику використання й визначати допустимі рівні зберігання відповідно до встановлених параметрів.

Для перевірки багатокритеріальної оптимізації можуть використовуватися тестові конфігурації з відомими тарифами та характеристиками продуктивності. Необхідно підтвердити правильність розрахунку вартості, урахування витрат на міграцію та дотримання обмежень затримки. Окремо перевіряється ситуація, коли найдешевший рівень не відповідає вимогам швидкодії. У такому випадку він не повинен обиратися для розміщення.

Перевірка механізму міграції передбачає копіювання контрольного об’єкта між двома рівнями. Після перенесення порівнюються контрольні

значення та перевіряється доступність інформації за початковим логічним ідентифікатором. Додатково необхідно випробувати переривання передавання та помилку оновлення метаданих. За таких умов вихідна копія повинна зберігатися до підтвердження успішного завершення операції.

У складі програмного забезпечення передбачається представлення відомостей про його розробника. Ця інформація використовується для ідентифікації авторства та організації подальшого технічного супроводження. Авторські дані системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії представлені на рисунку 5.2.

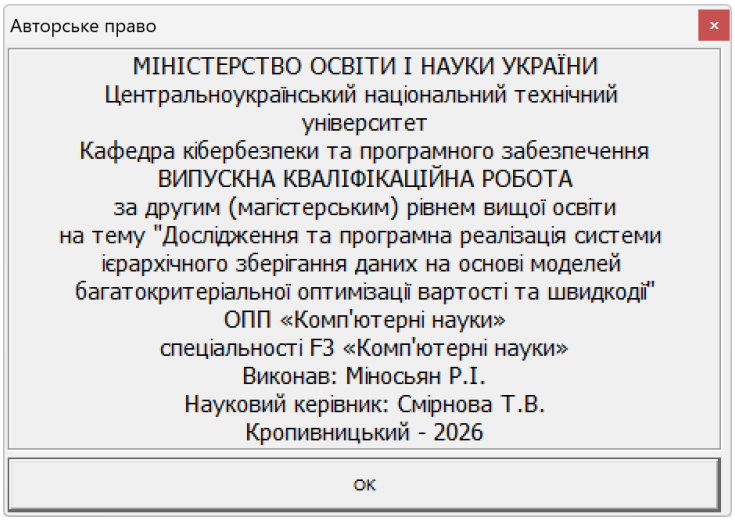


Рисунок 5.2 – Шаблонізована форма авторського права

6 НАУКОВА НОВИЗНА

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Метою розробки є дослідження та програмна реалізація системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Об'єктом дослідження є процес ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Предметом дослідження є методи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Методи дослідження базуються на методах обробки та зберігання інформації, методах оптимізації, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

– Розроблено вітчизняний продукт ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії, який має більш широкі можливості, на відміну від існуючих аналогів.

7 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

7.1 Вступ

В кожній ІТ-компанії є трудові відносини з працівниками. Згідно закону України “Про охорону праці” [3] кожна компанія впроваджує заходи з охорони праці. Реалізується трудові відносини з вживанням необхідних засобів з охорони праці та розробки відповідних документів:

– Інструкцій з охорони праці по кожній професії і загальні.

– Положення про охорону праці.

– Накази з охорони праці.

– Журнали реєстрації та інструктажу.

Роботодавець створює відділ який працює відповідно до типового положення, яке затверджується центральним органом виконавчої влади і забезпечує виконання вимог державної політики у сфері охорони праці.

За недотримання вимог, керівники ІТ-компаній можуть бути притягнуті до відповідальності, яка тягне накладання штрафу. Якщо в результаті порушення умов охорони праці є постраждалі працівники, то керівні особи ІТ-компаній притягуються до кримінальної відповідальності.

Законом України “Про охорону праці” [3] регламентуються загальні положення державної політики в галузі охорони праці, а конкретизуються ці положення нормативно-правовими актами про охорону праці, зокрема Наказом Міністерства соціальної політики України 14.02.2018 № 207, який зареєстровано в Міністерстві юстиції України 25 квітня 2018 р. за №508/31960 «Про затвердження Вимог щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями» [5], яким затверджено нормативно-правовий акт з охорони праці НПАОП 0.00-7.15-18, «Правила охорони праці під час експлуатації електронно-обчислювальних машин», та «Державні санітарні

правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» ДСанПіН 3.3.2-007-97.

Програмісти у процесі роботи мають негативний вплив на органи зору, а також мають значну розумову напругою і нервово-емоційне навантаження. Руки (суглоби пальців та м'язи рук) при роботі з клавіатурою мають теж істотне навантаженням. До шкідливих факторів, які впливають на робітників галузі інформаційних технологій (ІТ) спеціалісти відносять високочастотні електромагнітні коливання (випромінювання) роботи апаратної частини ЕОМ та виділення шкідливих газів.

Ці шкідливі фактори можуть привести до професійних захворювань.

Розглянемо шкідливі чинники роботи програмістів керуючись наступними нормативно-правовими актами: «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» ДСанПіН 3.3.2-007-98 [5], та «Вимоги щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями» НПАОП 0.00-7.15-17.

Умови праці програміста включають наступні фактори:

– параметри повітряного середовища в приміщенні;

– вентиляція приміщення;

– освітлення приміщення;

– параметри повітряного середовища в приміщенні, тощо.

Щоб запропонувати заходи щодо зменшення негативного впливу комп’ютера на організм людини визначимо фактори, які можуть викликати професійне захворювання і впливають на працездатність програміста.

7.2 Шкідливі і небезпечні фактори при роботі з комп’ютером

Електронно-обчислювальна машин (ЕОМ) та інше обладнання є джерелами небезпеки ураження електричним струмом. Оскільки робота програміста характеризується істотним зоровим навантаженням, то вимагає

належного освітлення. У приміщенні, в якому працюють програмісти, необхідно створити належний мікроклімат, параметри якого регламентуються Державними санітарними правилами і нормами, зокрема ДСанПіН 3.3.2.007-97.

При роботі з використанням ЕОМ відзначають наступні небезпечні та шкідливі фактори:

– ризик виникнення надзвичайних ситуацій природного або штучного характеру на об’єкті або території.

– ризик виникнення пожежі;

– негативний вплив на органи зору людини;

– ризики ураження електричним струмом;

– недостатня, або надмірна освітленість робочого місця;

– електромагнітні (у тому числі високочастотні) випромінювання (коливання);

– несприятливі мікрокліматичні умови;

– нервово-емоційна напруженість праці;

– інтелектуальні навантаження;

– монотонність праці;

– невідповідність ергономічних показників робочого місця діючим вимогам;

– шум;

– статичні навантаження на кістково-м’язовий апарат.

Працю користувачів ЕОМ відносять до психічних форм праці з високим ступенем навантаження. Ця діяльність пов'язана зі сприйняттям зображення на екрані, постійним стеженням за його динамікою, розрізненням картин, схем, читанням тексту рукописних та друкованих матеріалів, введенням інформації з клавіатури, необхідністю підтримувати активну увагу високою ціною помилки. Будь-яка діяльність із застосуванням ЕОМ супроводжується необхідністю активації уваги та інших вищих психічних функцій, а організм людини, крім того, піддається впливу кількох десятків різноманітних факторів.

7.3 Аналіз санітарно-гігієнічних умов праці на робочому місці програміста

Розглянемо умови праці у приміщенні, в якому працюють програмісти. Геометричні розміри приміщення наведено у таблиці 7.1.

Таблиця 7.1 – Розміри приміщення

Найменування	Значення, м
Ширина	5
Довжина	6,2
Висота	3,4

Таблиця 7.2 – Площа та обсяг приміщення, на одного працюючого*

Геометрична характеристика	Одиниця виміру	Нормативне значення*	Фактичне значення
Площа, S	$м^2$	не менше 6.0	6,2
Об'єм, V	$м^3$	не менше 20.0	21,8

* Згідно ДСанПіН 3.3.2.007-98 (Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин).

У зазначеному приміщенні працюють двоє людей. За даними, які наведено у таблицях 7.1-7.2, можна зробити висновок, що площа та об’єм приміщення у розрахунку на одне робоче місце програміста не відповідають нормативним вимогам ДСанПіН 3.3.2-007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» [5], але відповідають нормативним вимогам Наказу Міністерства соціальної політики України № 207, від 14.02.2018 «Про затвердження Вимог щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями» [5] та НПАОП 0.00-1.28-10 «Правила охорони

праці під час експлуатації електронно-обчислювальних машин»). Таним чином, можна зробити висновок, що санітарно-гігієнічні умови праці на робочому місці програміста відповідають вимогам.

Температура повітря в приміщенні визначається впливом температури зовнішнього повітря і тепловою енергією, яка виділяється всередині приміщення. Джерелами виділення теплоти в даному приміщенні є електроустаткування, освітлювальні прилади, а також люди. У світлий час доби джерелом надлишкового тепла є сонячна радіація. Згідно Постанови №42 від 01.12.1999 Головного державного санітарного лікаря України, робота, виконувана в даному приміщенні, відноситься до категорії Іа. В цьому випадку людина витрачає енергії до 120 ккал у годину. Вологість повітря в приміщенні визначається впливом багатьох факторів, серед яких: вологість атмосферного повітря, виділення вологи людьми (при диханні та випарами з поверхні шкіри).

Мікроклімат повітряного середовища в приміщенні характеризується запиленістю та загазованістю повітря. Мікроклімат приміщення визначається діючим на організм людини поєднанням, вологості, температури, швидкості руху повітря та інтенсивності теплового випромінювання. Аналіз мікроклімату складається з визначення зазначених вище факторів і порівняння результатів із встановленими нормами.

У таблиці 7.3 наведено оптимальні та фактичні значення параметрів мікроклімату як для категорії ваги робіт Іа, так і розглянутого приміщення. У приміщеннях, де встановлено ЕОМ, рекомендується дотримування тільки оптимальних значень показників мікроклімату.

Проведений аналіз показує, що показники мікроклімату в приміщенні відповідають установленим нормам. Штучне опалення застосовується у холодний період року.

В літню пору застосовується кондиціонер.

Для боротьби з пилом робляться регулярні провітрювання та вологі прибирання приміщенні.

Таблиця 7.3 – Оптимальні і фактичні значення параметрів мікроклімату

Пора року	Оптимальні для Іа			Фактичні		
	Температура, °С	Волого-гість,%	Швидкість повітря, м/с	Температура,°С	Воло-гість, %	Швидкість повітря, м/с
Холодна	22-24	40-60	0,1	22-23	40-58	0,11
Тепла	23-25	50-70	0,1	24-25	50-65	0,2

У приміщенні знаходяться наступні джерела шуму: принтер Epson L132 (C11CE58403), електродвигуни вентиляторів ЕОМ.

Одним з найважливіших факторів, які впливають на ефективність трудової діяльності людини та попереджають травматизм і професійні захворювання програмістів, є освітлення на робочому місці.

З 2019 року діють Державні будівельні норми України “Природне і штучне освітлення” – ДБН В.2.5-28:2018 [1], у яких прописані вимоги до використання всіх освітлювальних приладів, у тому числі світлодіодних.

Роботу працівника, який постійно працює за комп’ютером, згідно ДБН В.2.5-28:2018 [1], можна віднести до роботи з малою точністю (найменший розмір об'єкта розрізнення від 1 до 5 мм) V-го розряду зорової роботи, з великою контрастністю об'єкта розрізнення (символів на екрані дисплея), з темним тлом (під розряд зорової роботи В).

Приміщення можна віднести до 1-ої групи приміщень, у яких проводиться розрізнення об'єктів зорової роботи при фіксованому напрямку лінії зору того, хто працює, на робочу поверхню. Для такого типу приміщень і розряду зорової роботи нормоване значення коефіцієнта природної освітленості (КПО) робочої поверхні (при поєднаному, спільному освітленні), повинен становити не більше 1,5%, освітленість при штучному освітленні повинна

становити 300 Лк. [1], Крім того, все поле зору повинне бути освітлено достатньо рівномірно – це основна гігієнічна вимога. Оскільки яскраве світло на ділянці периферийного зору значно збільшує напруженість очей і, як наслідок, призводить до їх швидкої стомлюваності, ступінь освітлення приміщення і яскравість екрану комп'ютера повинні бути приблизно однаковими.

7.4 Розробка заходів з умов поліпшення охорони праці

Згідно аналізу умов праці в розглянутому приміщенні, ми одержали наступні результати:

– розмірі приміщення, у розрахунку на одному працюючого, відповідають нормативам;

– мікроклімат відповідає нормативному значенню;

– акустичні умови роботи не перевищують нормативних значень;

Таким чином можна припустити, що основною причиною можливого зниження працездатності програміста є психофізіологічний фактор, тому основна пропозиція буде така: дотримання позитивної психологічної атмосфери в колективі та регламентованого режиму праці та відпочинку, організація робочого місця з урахуванням ергономічних вимог.

Рекомендовані заходи: регулярні періодичні наочні огляди персоналом шляхів для евакуації людей із приміщення, відповідно до плану евакуації (який повинен розташовуватись на видному місці у приміщенні), включення до колективного договору мінімально можливого вмісту аптечок з обов’язковою наявністю масок-клапанів, або іншого спорядження для штучного дихання. Регулярна періодична перевірка параметрів заземлення та занулення (вимірювання опору ланцюга).

Регулярна наочне знайомство персоналу із шляхами для евакуації людей із приміщення відповідно до плану евакуації, забезпечення розподільних щитів спеціальними розетками з заземлюючими контактами; організація заземлення всіх приладів і пристроїв, які працюють при напрузі вище 36 В.

Так як при ураженні електричним струмом у людини може статися фібриляція шлуночків серця, в організації бажано мати дефібрилятор і підготовлений персонал для роботи з ним.

7.5 Розрахункова частина

Проведемо розрахунок штучного освітлення за методом коефіцієнту використання світлового потоку для приміщення ширина якого складає 5 м, довжина – 6,2 м, висота – 3,4 м.

У зазначеному приміщенні працює 5 людей.

Для того, щоб визначити потрібну кількість світильників, які повинні забезпечити нормований рівень освітленості, визначимо світловий потік, що падає на робочу поверхню за формулою:

$$F = E \cdot S \cdot K \cdot Z/n,$$

де

F – світловий потік, що розраховується, Лм;

E – нормована мінімальна освітленість, Лк; E = 300 Лк;

S – площа освітлюваного приміщення (у нашому випадку S = 5х6,2 = 31 $м^2$);

K – коефіцієнт запасу, що враховує зменшення світлового потоку лампи в результаті забруднення світильників в процесі експлуатації (його значення залежить від типу приміщення і характеру робіт, що проводяться в ньому, в нашому випадку K = 1,5);

Z – відношення середньої освітленості до мінімальної (зазвичай приймається рівним 1.1... 1.2, в нашому випадку Z = 1,1);

n – коефіцієнт використання світлового потоку, (відношення світлового потоку, що падає на розрахункову поверхню, до сумарного потоку всіх ламп, обчислюється в долях одиниці; залежить від характеристик світильника, розмірів приміщення, забарвлення стін і стелі, що характеризуються коефіцієнтами відбиття від стін ($\rho_{стін.}$) і стелі ($\rho_{стелі}$), значення коефіцієнтів дорівнюють $\rho_{стін}$ =

50% і $\rho_{стелі}$ = 50%.

Обчислимо індекс приміщення за формулою:

$$i=S/(h\cdot(A+B)),$$

де:

S – площа приміщення, S = 31 м2;

h – розрахункова висота підвісу, h = 3 м (співпадає з висотою стелі, оскільки лампи освітлення закріплюються на стелі);

A – ширина приміщення, A = 5 м;

B – довжина приміщення, B = 6,2 м.

Підставимо всі значення у формулу та визначимо індекс приміщення:

$$i=0,43.$$

Знаючи індекс приміщення, за знаходимо n = 0,23 (з табличних даних коефіцієнтів використання світлового потоку (n) світильників з відповідним типом лампам). Підставимо всі значення у формулу, визначимо світловий потік: F=66717 Лм.

Для розрахунку будемо використовувати світлодіодні панелі LED панель PL PFM 600 30W/3000K, світловий потік яких $F_л$ = 3000 Лм.

Число ламп визначається за формулою:

$$N=F/F_л$$

де

F – світловий потік,

$F_л$ – світловий потік однієї лампи.

Підставимо всі значення у формулу та визначимо потрібну кількість ламп:

$$N= 66717/3000 = 22,18 \text{ шт.}$$

Приймаємо необхідну кількість світлодіодних світильників 23 шт.

Висновки до розділу

Дотримання всіх необхідних умов праці не лише сприяє збереженню здоров’я працівників, а також підвищує ефективність виробництва в цілому.

З цих міркувань було здійснено аналіз умов праці, призначеного для праці програмістів, проведено розгляд небезпечних та шкідливих факторів, що негативно впливають на програмістів під час роботи.

Виконано розрахунок штучного освітлення, як одного з ключових факторів впливу на працездатність та здоров’я програміста. Розроблено заходи з охорони праці.

8 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти, призначено для системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

В межах України в недостатній мірі представлені вітчизняні розробки в цій області.

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти наведені теоретичне узагальнення й рішення наукового завдання дослідження методів ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Рішення даного завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

– Досліджена система ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

– На основі отриманих результатів досліджень створена програмна реалізація системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Розроблені під час виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії.

Проведено аналіз предметної галузі в ході якого були виявлені об'єкти, взаємодія яких носить істотний характер для функціональної діяльності предметної галузі, і їхні основні характеристики; побудована алгоритм і вибраний середовище розробки.

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість у освоєнні роботи програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створені програмного забезпечення було використано об'єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Програма реалізована на мові високого рівня Python. Дана мова програмування дозволяє найбільш ефективно обробляти дані. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи Windows 10/11.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

В цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і застосування у різних галузях.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Міносьян Р.І. Дослідження та програмна реалізація системи ієрархічного зберігання даних на основі моделей багатокритеріальної оптимізації вартості та швидкодії // Збірник праць молодих науковців ЦНТУ. – Вип. 17. – Кропивницький: ЦНТУ, 2027.
2. JJ Geewax. API Design Patterns. Manning Publications Co. 2021. 481 p.
3. Prateek Prasad. App Design Apprentice. Razeware LLC. 2020. 272 p.
4. Dawn Griffiths, David Griffiths. Head First Android Development. O'Reilly Media, Inc. 2021. 1414 p.
5. Nathan Metzler. Kotlin Programming for Beginners. Independently published. 2021. 158 p.
6. Aaron Torres. Go Programming Cookbook Second Edition. Packt Publishing Ltd. 2019. 427 p.
7. Мелешко Є.В., Якименко М.С., Поліщук Л.І. Алгоритми та структури даних: Навчальний розділ для студентів технічних спеціальностей денної та заочної форми навчання. – Кропивницький: Видавець – Лисенко В.Ф., 2019. – 156 с.
8. Knuth D. The Art of Computer Programming, Vol. 1: Fundamental Algorithms, 3rd Edition 3rd Edition. – Addison-Wesley Professional, 2019. – 672 p.
9. Knuth D. Art of Computer Programming, Vol. 2: Seminumerical Algorithms 3rd Edition, Kindle Edition. – Addison-Wesley Professional, 2019. – 672 p.
10. Knuth D. The Art of Computer Programming: Vol. 3: Sorting and Searching 2nd Edition, Kindle Edition. – Addison-Wesley Professional, 2019. – 800 p.
11. Cormen T.H., Leiserson C.E., Rivest R.L., Stein C. Introduction to Algorithms, 3rd Edition (The MIT Press) 3rd Edition – The MIT Press, 2019. – 1292 p.
12. Богаченко, С., Смірнов, О., Буравченко, К., Смірнова, Т.,

Конопліцька-Слободенюк, О., Якименко, Н., Смірнов, С. «Обмеження ресурсів мобільних та edge-платформ при розробці програмного забезпечення для реалізації та оптимізації моделей машинного навчання». *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»*, 2026, т. 2 № 34, С. 677-686.

13. Смірнов, О., Ткачук, Р., Козірова, Н., Константинова, Л., Конопліцька-Слободенюк, О., Якименко, Н., Смірнов, С.. «Дослідження застосування векторних баз даних у генеративному штучному інтелекті». *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»*, 2026, Том 1 № 33, 667-682.

14. Вінтенко Б.Ю., Смірнова Т.В., Миронець І.В., Смірнов О.А., Буравченко К.О. «Метод оцінки функціональної стійкості комп'ютерно-орієнтованих процедур системи підтримки оперативного персоналу АЕС». *Системи управління, навігації та зв'язку*, 2026, вип. 2(84), С. 62-69.

15. Смірнова Т.В., Усік П.С., Лисенко І.А., Буравченко К.О., Смірнов О.А. «Методологія підтримки технологічних процесів у критичній інфраструктурі з забезпеченням безпеки інформації на основі хмарних технологій». *Безпека інформації.* 2025. Том 31 № 1. С. 49-60.

16. Смірнова Т.В., Буравченко К.О., Добринчук О.А., Смірнов С.А., Якименко Н.М., Смірнов О.А. «Хмарна технологія моніторингу ключових індикаторів ефективності технологічних процесів у критичній інфраструктурі». *Центральноукраїнський науковий вісник. Технічні науки.* 2025. № 12(43), ч. II. С. 36-54.

17. Лисенко, І., Минайленко, Р., Смірнов, С., Буравченко, К., Якименко, Н., Смірнов, О. «Дослідження інструментів штучного інтелекту для інтелектуального аналізу даних». *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»*, 2025, № 3(31), С. 227–241.

18. Vintenko, B., Myronets, I., Smirnov, O., Smirnova, T. «Enhancing the Reliability of Input Data for Nuclear Power Plant Operator Support Systems». *International Conference on Sensors, Devices and Systems. SDaS 2025. Lecture Notes*

in Electrical Engineering, 2026. vol 1617. pp 3-12 Springer, Cham.

19. Смірнов О., Заріцкий В., Буравченко К., Конопліцька-Слободенюк О., Константинова Л., Якименко Н., Смірнов С. «Оптимізація розпізнавання облич за допомогою CUDA прискореної бібліотеки dlib». *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»*, 2026, Том 4, № 32, С. 573-582.

20. Kuznetsov O., Frontoni E., Kuznetsova Y., Chevardin V., Smirnov O. «Architectural foundations for adaptive security in edge computing systems». *Cybersecurity Defensive Walls in Edge Computing*, 2025. pp. 21-61.

21. Вінтенко, Б.Ю., Миронець, І.В., Смірнов, О.А., Коваленко, О.В., Усік, П.С., Буравченко, К.О., Лисенко, І.А. «Логіко-структурна модель комп'ютерно-орієнтованої процедури системи підтримки оперативного персоналу АЕС». *Кібербезпека: освіта, наука, техніка*. 2025. Том 2 № 30. С. 413-427, 2025.

22. Смірнова, Т.В. «Дослідження методів, моделей та сучасних ІТ-рішень для підтримки технологічних процесів у критичній інфраструктурі держави». *Кібербезпека: освіта, наука, техніка*. 2025. Том 2 № 30. С.195-208, 2025.

23. Al-Azzeh, J., Ayyoub, B., Mesleh, A., Smirnova, T., Gnatyuk, S., Drieiev, O., Smirnov, O., Dorenskyi, O. «Cloud-Based Information System for Evaluating Caverns in the Process of Blasting Metal Surfaces of Details». *International Review on Modelling and Simulations* 18 (1), 2025. pp. 32-42.

24. Вінтенко Б.Ю., Смірнов О.А., Миронець І.В., Смірнова Т.В., Коваленко О.В., Мацуй А.М. «Модель шляхів отримання вхідних даних комп’ютерної інтелектуальної системи підтримки оперативного персоналу АЕС». *Центральноукраїнський науковий вісник. Технічні науки.* 2025. Вип. 11(42), ч. II. С.52-62.

25. Смірнов, О.А., Константинова, Л.В., Конопліцька-Слободенюк, О.К., Козірова, Н.В, Якименко, Н.М., Доренський, О.П., Буравченко, К.О. «Дослідження інструментів штучного інтелекту для роботи з базами даних та

аналізу даних». *Кібербезпека: освіта, наука, техніка.* 2025. №3(27), С. 429–448.**)**

26. Smirnov O., Fedorov E., Neskorodieva A., Neskorodieva T. «Intellectual Classification method of Gymnastic Elements Based on Combinations of Descriptive and Generative Approache». *CEUR Workshop Proceedings* Volume 3664, 2024, Pages 11-23.

27. Вінтенко, Б., Миронець, І., Смірнов, О., Коваленко, А., Конопліцька-Слободенюк, О., Смірнова, Т., Константинова, Л. «Дослідження застосування систем підтримки оперативного персоналу об'єкту критичної інфраструктури при керуванні енергоблоком АЕС з реактором типу ВВЕР-1000». *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»,* 2024. № 2(26), С. 6-26.

28. Вінтенко, Б., Миронець, І., Смірнов, О., Кравчук, О., Козірова, Н., Савеленко, Г., Коваленко, А. «Дослідження вимог та аналіз кібербезпеки програмного забезпечення інформаційно-керуючих систем АЕС, важливих для безпеки». *Кібербезпека: освіта, наука, техніка.* 2024. №3(23), С. 111-131.

29. Kuznetsov O., Ilchenko O., Kryvinska N., Buravchenko K., Smirnov O., Savchenko Iu. «An Empirical Assessment of Leading Blockchain Financial Services». *2023 IEEE 1st Ukrainian Distributed Ledger Technology Forum (UADLTF)*, Kyiv, Ukraine, 2023, pp. 1-6,

30. Kuznetsov, O., Kryvinska, N., Ilchenko, O., Smirnova, T., Ulianovska, Y. «Comparative Analysis of Cryptocurrency Trading Platforms Using the Analytic Hierarchy Process». *CEUR Workshop Proceedings*, 2023, 3628, pp. 106-115.

31. Malyukov V., Bebeshko B., Lakhno V., Smirnov O., Malyukova I., Mohylnyi H. «Managing the Purchase-Sale Process of Digital Currencies Under Fuzzy Conditions». *Lecture Notes in Networks and Systems,* 2023, 729 LNNS, pp. 104–112.

32. Al-Mudhafar Aqeel, A.M., Smirnova, T., Buravchenko, K., Smirnov, O. «The method of assessing and improving the user experience of subscribers in software-configured networks based on the use of machine learning». *Advanced Information Systems*, 2023, 7(2), pp. 49-56.

33. Smirnov, O., Sydorenko, V., Aleksander, M., Zhyharevych, O., Yenchev,

S. «Simulation of the cloud IoT-based monitoring system for critical infrastructures». *CEUR Workshop Proceedings,* Volume 3530, 2023, pp. 256-265.

34. Smirnov, O., Odarchenko, R., Smirnova, T., Bondar, S., Volosheniuk, D. «Optimal Structure Construction of Private 5G Network for the Needs of Enterprises». Lecture Notes on Data Engineering and Communications Technologies, 2023, 178, pp. 208–223.

35. Вінтенко Б.Ю., Смірнов О.А., Коваленко А.С., Смірнов С.А., Буравченко К.О. «Дослідження вимог міжнародних стандартів IEC60880 та IEC62138 з розробки програмного забезпечення інформаційно-керуючих систем АЕС, важливих для безпеки». *Системи управління, навігації та зв'язку*, 2023, вип. 3(73), С. 155-166.

36. Вінтенко, Б., Миронець, І., Смірнов, О., Кравчук, О., Козірова, Н., Савеленко, Г., Коваленко, А. «Дослідження вимог та аналіз кібербезпеки програмного забезпечення інформаційно-керуючих систем АЕС, важливих для безпеки». *Кібербезпека: освіта, наука, техніка.* 2024. №3(23), С. 111-131.

37. Вінтенко Б.Ю., Смірнов О.А., Коваленко О.В., Смірнов С.А., Коваленко А.С. «Дослідження нормативних документів та галузевих стандартів розробки програмного забезпечення комп'ютерних систем управління АЕС, важливих для безпеки». Системи управління, навігації та зв'язку, 2023, вип. 2(72), С. 170-178.

38. Аль-Мудхафар Акіл Абдулхуссейн М., Смірнова Т.В., Буравченко К.О., Смірнов О.А. «Метод оцінки та підвищення користувальницького досвіду абонентів в програмно-конфігурованих мережах на основі використання машинного навчання». Сучасні інформаційні системи, 2023, том 7, № 2, С. 49-56.

39. Вінтенко Б.Ю., Смірнов О.А., Коваленко О.В., Смірнов С.А. «Дослідження нормативної документації та стандартів розробки програмного забезпечення комп'ютерних систем управління АЕС, важливих для безпеки». VI міжнародна науково-практична конференція "Інформаційна безпека та комп'ютерні технології", м. Кропивницький. 20-21 квітня 2023 р. –

Кропивницький: ЦНТУ. – 2023. – С. 35-36.

40. Smirnov, O., Karapetyan, A., Fedorov, E., «Creating Neural Network and Single Solution Human-Based Metaheuristic Methods of Solving the Traveling Salesman Problem». CEUR Workshop Proceedings, Volume 3312, 2022, pp. 47-58.

41. Smirnov O., Kuznetsov A., Kryvinska N., Kiian A., Kuznetsova K. «Full Non-Binary Constant-Weight Codes». SN Computer Science, Vol 2, 337, 2021. https://doi.org/10.1007/s42979-021-00739-w.

42. Smirnov O., Kovalenko O., Kovalenko A., Kavun S. «Quantitative Risk Assessment Method Development in the Context of the SDLC-model». 2021 IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (PIC S&T), 2021, pp. 203-208, doi: 10.1109/PICST54195.2021.9772143

43. Smirnova T., Gnatyuk S., Berdibayev R., Avkurova Zh., Iavich M. «Cloud-Based Cyber Incidents Response System and Software Tools». Communications in Computer and Information Science, 2021, vol 1486. Springer, Cham. pp 169-184.

44. Smirnov, O., Kuznetsov, A., Potii, O., Poluyanenko, N., Stelnyk, I., Mialkovsky, D. «Combining and filtering functions in the framework of nonlinear-feedback shift register». International Journal of Computing; 2020, Volume 19, Issue 2 – Research Institute for Intelligent Computer Systems – 2020. – P. 247-256.

45. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». CEUR Workshop Proceedings. Volume 2740, 2020, Pages 102-114.

46. Smirnov O., Kuznetsov A., Kiian A., Cherep A., Kanabekova M., Chepurko I. «Testing of code-based pseudorandom number generators for post-quantum application». 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT), Ukraine, Kyiv, May 14-18. 2020. P. 172-177.

47. Smirnov, O., Shekhanin, K., Kuznetsov, A., Krasnobayev, V. «Detecting Hidden Information in FAT». International Journal of Computer Network and Information Security (IJCNIS). Vol. 12, No. 3, 2020. PP.33-43.

48. Smirnov, O., Drieieva, H., Drieiev, O., Simakhin, V., Bondar, S., Odarchenko, R. «Managing multifractal properties of the binary sequence generated with the Markov chains», CEUR Workshop Proceedings Volume 2608, 2020, Pages 633-645.

49. Smirnov O. Kuznetsov A., Zaicenko Yu., Pastukhov M., Oleshko O., Kuznetsova K., «Formation of Discrete Signals with Special Correlation Properties». International Conference on Information and Telecommunication Technologies and Radio Electronics, UkrMiCo 2019; Odessa; Ukraine; 9-13 September 2019. P.22-28.

50. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». International Journal of Computing; 2019, Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407.

51. Smirnov, O., Kuznetsov, A., Reshetniak, O., Ivko, N., Katkova, T., Kuznetsova, T., «Generators of Pseudorandom Sequence with Multilevel Function of Correlation». 2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T), Kyiv, Ukraine, 8 – 11 October 2019. P.517-522.

52. Smirnov, O., Krasnobayev, V., Yanko, A., Kuznetsova, T. «Methods of nulling numbers in the system of residual classes». CEUR Workshop Proceedings, Vol 2588, P. 90-106, 2019.

53. Kuznetsova, T., «Code-Based Schemes for Post-Quantum Digital Signatures», 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019; Metz; France; 18-21 September 2019. P. 707-712.

54. Smirnov, O., Kuznetsov, A., Stefanovych, O., Gorbenko, Y., Krasnobaev, V., Kuznetsova K. «Information Hiding Using 3D-Printing Technology», 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019; Metz; France; 18-21 September 2019. P.701-706.

55. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Averchev, A., Pastukhov,

M., Kuznetsova, K., «Formation of Pseudorandom Sequences with Special Correlation Properties», 2019 3rd International Conference on Advanced Information and Communications Technologies, AICT -2019/ Lviv, Ukraine, 2-6 July, 2019, P. 395-399.

56. Смірнов, О.А., Усік П.С., Полігенько О.О., Одарченко Р.С., Терещенко Л.Ю. «Інформаційна технологія та програмне забезпечення для підвищення ефективності планування підсистеми базових станцій стільникового зв'язку». Проблеми телекомунікацій. № 1(26). С. 83-96. 2020.

57. Смірнов О.А., Усік П.С., Миронець І.В., Буравченко К.О., Якименко Н.М. «Метод підвищення ефективності розподіленої обробки даних у комп'ютерних системах операторів стільникового зв'язку» Вісник Черкаського державного технологічного університету. Технічні науки. №4. С. 103-110. 2020.

58. О.А.Смірнов, Т.В.Смірнова, Л.І. Поліщук, К.О. Буравченко, А.О.Макевнін, «Дослідження хмарних технологій як сервісів», Кібербезпека: освіта, наука, техніка. № 3(7). С. 43-62. 2020.

59. Смірнов О.А., Дрєєва Г.М., Дрєєв О.М., Смірнова Т.В. «Фрактальний аналіз генератора самоподібного трафіку на основі ланцюга Маркова». Центральноукраїнський науковий вісник. Технічні науки. № 2(33). с. 161-172, 2019.

60. Smirnov, O., Kuznetsov, A., Kuznetsova., K. Synthesis of Discrete Signals with Improved Correlation Properties. Монографія: In.: ISCI'2019: Information Security in Critical Infrastructures. Collective monograph. Edited by Ivan D. Gorbenko and Alexandr A. Kuznetsov, ASC Academic Publishing, USA, 2019, pp. 281-299. – ISBN: 978-0-9989826-8-7 (Hardback), ISBN: 978-0-9989826-9-4 (Ebook).

61. Смірнова Т.В., Солових Є.К., Смірнов О.А., Дрєєв О.М. Побудова хмарних інформаційних технологій оптимізації технологічного процесу відновлення та зміцнення поверхонь деталей. Центральноукраїнський науковий вісник. Технічні науки. № 1(32). с. 184-194, 2019.

62. Смірнов О.А., Котелянець В.В. Стійкі до колізій стохастичні моделі

функціонування безпроводових сенсорних мереж. Вісник інженерної академії України, №3, с. 145-152, 2018

63. O. Smirnov, O. Kovalenko, A. Kovalenko, S. Smirnov, V. Vialkova. The mathematical model of the testing technology for Dom Xss vulnerabilities. Scientific & practical cyber security journal (SPCSJ) Vol 2 Issue 1, 22-28 pp. [Электронный Журнал]. Georgia. Tbilisi: SCSA – 2018.

64. Oleksii Smirnov, Oleksandr Kovalenko, Jamil Al-Azzeh, Anna Kovalenko, Serhii Smirnov. Qualitative risk analysis of software development. Asian Journal of Information Technology. – Volume 17(3). – Medwell Journals. – 2018. – P. 218-230.

65. Смірнов О.А., Коваленко О.В., Коваленко А.С., Смірнов С.А. Розробка методу передтестової компіляції й розподілу доступу. Збірник наукових праць III міжнародної науково-практичної конференції "Інформаційна безпека та комп'ютерні технології", м. Кропивницький. 19-20 квітня 2018р. – Кропивницький: ЦНТУ. – 2018. – С. 214-215