Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
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 першим (бакалаврським) рівнем вищої освіти
на тему
“Програмне забезпечення системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P”

Виконав здобувач вищої освіти
IV курсу, групи КІ-22-1
ОПП «Комп’ютерна інженерія»
спеціальності 123 «Комп’ютерна інженерія»
_____ Мінжулін Є.О.
«____» _____ 2026 р.

Керівник проекту
доктор філософії (PhD)
_____ Дрєєва Г.М.
«____» _____ 2026 р.
Рецензент _____
_____

м. Кропивницький

Центральноукраїнський національний технічний університет

Факультет *Механіко-технологічний*

Кафедра *Кібербезпеки та програмного забезпечення*

Освітній ступінь *бакалавр*

Галузь знань ._12 _____*"Інформаційні технології"*_____

Спеціальність 123 *"Комп'ютерна інженерія"*

Освітньо-професійна (освітньо-наукова) програма _*"Комп'ютерна інженерія"*_

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

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ПЕРШИМ (БАКАЛАВРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Мінжуліну Євгенію Олексійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Програмне забезпечення системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P*

2. Керівник роботи *Дрєєва Ганна Миколаївна, доктор філософії (PhD)*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 133-02 від 27.02.2026 року

3. Строк подання студентом роботи до захисту *23.05.2026 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою роботи є розробка програмного забезпечення системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P*

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

1. Призначення та область використання.
2.Перегляд аналогічних існуючих систем.
3. Опис і обґрунтування проектних рішень.
4. Етапи програмування системи.
5. Впровадження системи в промислову експлуатацію.
6. Висновки

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

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

7. Дата видачі завдання « 15 » січня 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.03.2026 р.	
2.	Постановка задачі, оформлення ТЗ	15.03.2026 р.	
3.	Розробка моделі компонента	20.03.2026 р.	
4.	Розробка структур даних	25.03.2026 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.03.2026 р.	
6.	Програмування алгоритмів	10.04.2026 р.	
7.	Оформлення ПЗ	17.04.2026 р.	
8.	Попередній захист роботи	23.05.2026 р.	

Дата видачі завдання
« 15 » січня 2026 р.

Підпис керівника
_____ Дрєєва Г.М.
(прізвище та ініціали)

Завдання прийнято до виконання
« 15 » січня 2026 р.

Підпис здобувача
_____ Мінжулін Є.О.
(прізвище та ініціали)

АНОТАЦІЯ

Мінжулін Є.О. Програмне забезпечення системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2026.

В даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

Метою розробки є програмне забезпечення системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

Результат роботи – програмна реалізація системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

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'ютерна інженерія, SAN, Fibre Channel-PI-6P

ABSTRACT

Minzhulin E.O. Software for SAN client application support system using the Fibre Channel-PI-6P standard. 123 Computer Engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2026.

In this final qualification work for the first (bachelor's) level of higher education, software has been developed that is intended for a SAN client application support system using the Fibre Channel-PI-6P standard.

The purpose of the development is software for a SAN client application support system using the Fibre Channel-PI-6P standard.

The result of the work is a software implementation of a SAN client application support system using the Fibre Channel-PI-6P standard.

In the process of working on the software model, an analysis of existing hardware and software was performed. All components of the developed software are fully described.

A convenient user interface has been developed. Instructions for working with software are provided.

The program can be used on PCs with Windows 10/11.

The program is developed in Python.

Keywords: computer engineering, SAN, Fibre Channel-PI-6P

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

БД	–	база даних
КВ	–	коефіцієнт варіації
КЗ	–	канал зв'язку
ПС	–	програмна середа
СеМО	–	експонентна мережа масового обслуговування
СМО	–	система масового обслуговування
СПД	–	система передачі даних
ЦОД	–	центр обробки даних
FC	–	fibre channel – волоконний канал
LC	–	дуплексний коннектор
MPO	–	Multi-Fiber Push On – з'єднувачі багатоволоконних магістральних кабелів
SAN	–	Storage Area Network

ВСТУП

Актуальність теми. Протягом багатьох років технологія Fibre Channel була обділена увагою, оскільки інтерес галузі був прикутий до таких нових рішень, як Fibre Channel over Ethernet і мережі зберігання на основі протоколу IP. Але технологія усе ще жива, більше того, відповідний ринок продовжує розширюватися. Його основні каталізатори – поширення економічних флеш-масивів, доступність трансіверів шостого покоління із пропускною здатністю 32 Гбіт/с, 128 GFC і інших технологій, а також поява багатомодових волоконно-оптичних кабельних систем з поліпшеними характеристиками.

Для систем зберігання на базі флеш-накопичувачів дуже важлива наявність можливості підключення до каналів Fibre Channel Gen6, оскільки прискорення доступу й стійке виконання операцій читання-запису при застосуванні цієї технології дозволяє істотно поліпшити продуктивність інфраструктури зберігання в порівнянні з попередніми поколіннями Fibre Channel.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

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

– Огляд існуючих систем підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

– Дослідження системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

– Програмна реалізація системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Система призначена для використання стандарту Fibre Channel-PI-6P. Fibre Channel (FC) (англ. fibre channel – волоконний канал) – сімейство протоколів для високошвидкісної передачі даних. Стандартизацією протоколів займається Технічний комітет T11, що входить до складу Міжнародного комітету зі стандартам у сфері IT (INCITS), акредитованого Американським національним інститутом стандартів (ANSI). Споконвічне застосування FC в області суперкомп'ютерів згодом практично повністю перейшло в сферу мереж зберігання даних, де FC використовується як стандартний спосіб підключення до систем зберігання даних рівня підприємства.

Fibre Channel Protocol (FCP) – транспортний протокол (як TCP в IP-мережах), що інкапсулює протокол SCSI мережами Fibre Channel. Є основою побудови мереж зберігання даних.

Варто відзначити, що з моменту твердженні специфікації нового інтерфейсу й до появи на ринку підтримуючих його продуктів проходить звичайно кілька місяців. Широке поширення нового стандарту може затягтися на кілька років.

Також зараз ведеться робота з розробки версій уже існуючих інтерфейсів зі зниженим енергоспоживанням.

Fibre Channel

32Gbps FC (32GFC)

Робота над стандартом 32GFC, FC-PI-6, почалася на початку 2010 року. У грудні 2013 асоціація Fibre Channel Industry Association (FCIA) повідомила про завершення роботи над специфікацією. Продукти, що підтримують цей

інтерфейс, з'явилися на ринку в 2015 або 2016 роках. 32GFC використовує 25/28G SFP+ коннектор.

Мультиканальний інтерфейс FC 128Gb, відомий як 128FCp (паралельний чотирьохканальний), ґрунтується на технології FC 32Gb і доданий в офіційний план розвитку стандарту FC. Комітет T11 привласнив проекту назва FC-PI-6P. Деякі виробники представляють 32GFC і 128GFC як «Gen 6» Fibre Channel, тому що ця версія підтримує 2 різні швидкості передачі даних у двох різних конфігураціях (послідовної й паралельної).

64Gbps FC (64GFC), 256Gbps FC (256GFC)

Розробка стандартів 64GFC і 256GFC почалася в проекті FC-PI-7. Технічна стабільність очікується в 2027 році. Кожна ревізія FC назад сумісна як мінімум із двома попередніми поколіннями.

FC як інтерфейс SAN

Очевидно, Fibre Channel у доступному для огляду майбутньому буде залишатися основною технологією для побудови мереж SAN. За минулі роки в інфраструктуру FC були інвестовані значні засоби (мільярди доларів США), в основному, у центри обробки даних, які будуть функціонувати протягом ще багатьох років.

FC як дисковий інтерфейс

Fibre Channel як інтерфейс для підключення дисків іде в минуле, тому що виробники дисків корпоративного класу переходять на 6Gbps SAS і 12Gbps SAS. Через досить великий обсяг випущених 3.5-дюймових дисків з інтерфейсом FC, що використовуються в корпоративних дискових підсистемах, очікується, що FC буде використовуватися ще якийсь час для їхньої підтримки. Серед 2.5-дюймових дисків інтерфейс Fibre Channel, швидше за все, буде доступний на дуже невеликому числі пристроїв.

Fibre Channel over Ethernet

FCoE (FC-BB-6)

Робота над стандартом FC-BB-6 була завершена комітетом T11 у серпні 2014 року. FC-BB-6 стандартизує архітектуру VN2VB і поліпшує масштабованість Domain_ID.

VN2VN – це спосіб з'єднати прямо кінцеві вузли FCoE (Virtual N_Ports) без необхідності в FC або FCoE комутаторах (FC Forwarders), що дозволяє спростити конфігурацію в невеликих розміщеннях. Цю ідею іноді називають «Ethernet Only» FCoE. У таких мережах не потрібно зонування, що дає меншу складність і зменшує витрати.

Масштабованість Domain_ID (Domain_ID Scalability) дозволяє FCoE фабрикам масштабуватися до більших SAN.

40Gbps і 100Gbps

До появи 40Gbps FCoE залишився рік або два. Можливо, інтерфейс з'явиться одночасно з 32Gb FC. Стандарти IEEE 802.3ba 40Gbps і 100Gbps Ethernet були ратифіковані в червні 2010. Нові продукти повинні з'явитися через якийсь час.

Швидше за все, 40Gbps і 100Gbps FCoE, засновані на стандартах Ethernet 2010 року, будуть використовуватися спочатку для ISL-Ядер, тим самим залишаючи 10Gb FCoE в основному для кінцевих з'єднань. Очікується, що майбутні версії 100GFCoE кабелів і коннекторів будуть доступні в конфігураціях 10x10 і потім 4x25.

1.2 Область застосування

Областю застосування є SAN. SAN – Storage Area Network – призначена для консолідації дискового простору серверів на спеціально виділених дискових сховищах. Суть у тому, що так дискові ресурси ощадливіше використовуються, легше управляються й мають більшу продуктивність. А в питаннях віртуалізації й

кластеризації, коли декільком серверам потрібний доступ до одного дискового простору, подібні системи зберігання даних взагалі незамінна штука.

До речі, у термінологіях SAN, завдяки перекладу на українську, виникає деяка плутанина. SAN у перекладі означає «мережа зберігання даних» – СЗД. Однак класично в Україні під СЗД розуміється термін «система зберігання даних», тобто саме дисковий масив (Storage Array), що у свою чергу складається з Керуючого блоку (Storage Processor, Storage Controller) і дискових полиць (Disk Enclosure). Однак, в оригіналі Storage Array є лише частиною SAN, хоча часом і самої значимої. В Україні одержуємо, що СЗД (система зберігання даних) є частиною СЗД (мережі зберігання даних). Тому пристрою зберігання звичайно називають СЗД, а мережа зберігання – SAN (і плутають із «Sun», але це вже дріб'язки).

Компоненти й терміни

Технологічно SAN складається з наступних компонентів:

1. Вузли, ноди (nodes)

– Дискові масиви (системи зберігання даних) – сховища (таргети [targets]).

– Сервери – споживачі дискових ресурсів (ініціатори [initiators]).

2. Мережева інфраструктура

– Комутатори (і маршрутизатори в складних і розподілених системах).

– Кабелі.

Особливості

Якщо не вдаватися в деталі, протокол FC схожий на протокол Ethernet з WWN-адресами замість MAC-адрес. Тільки, замість двох рівнів Ethernet має п'ять (з яких четвертий поки не визначений, а п'ятий – це маппинг між транспортом FC і високорівневими протоколами, які по цьому FC передаються – SCSI-3, IP). Крім того, у комутаторах FC використовуються спеціалізовані сервіси, аналоги яких для IP мереж звичайно розміщаються на серверах. Наприклад: Domain Address Manager (відповідає за призначення Domain ID комутаторам), Name Server

(зберігає інформацію про підключені пристрої, такий аналог WINS у межах комутатора) і т.д.

Для SAN ключовими параметрами є не тільки продуктивність, але й надійність. Адже якщо в сервера БД пропаде мережа на парі секунд (або навіть мінут) – ну неприємно буде, але пережити можна. А якщо на цей же час відвалиться жорсткий диск із базою або з ОС, ефект буде куди більше серйозним. Тому все компоненти SAN звичайно дублюються – порти в пристроях зберігання й серверах, комутатори, лінки між комутаторами й, ключова особливість SAN, у порівнянні з LAN – дублювання на рівні всієї інфраструктури мережевих пристроїв – фабрики.

Фабрика (fabric – що взагалі-то в перекладі з англійської, тому що термін символізує переплетену схему підключення мережевих і кінцевих пристроїв, але термін уже встоявся) – сукупність комутаторів, з'єднаних між собою міжкомутаторними лінками (ISL – InterSwitch Link).

Високонадійні SAN обов'язково включають дві (а іноді й більше) фабрики, оскільки фабрика сама по собі – єдина точка відмови. Ті, хто хоч раз спостерігав наслідку кільця в мережі або спритному русі клавіатури, що вводить у кому комутатор рівня ядра або розподіли невдалим прошиванням або командою, розуміють про що мова.

Фабрики можуть мати ідентичну (дзеркальну) топологію або розрізнятися. Наприклад одна фабрика може складатися із чотирьох комутаторів, а інша – з одного, і до неї можуть бути підключені тільки висококритичні вузли.

Топологія

Розрізняють наступні види топологій фабрики:

Каскад – комутатори з'єднуються послідовно. Якщо їх більше двох, то ненадійно й непродуктивно.

Кільце – замкнутий каскад. Надійніше просто каскаду, хоча при великій кількості учасників (більше 4) продуктивність буде страждати. А одиничний збій ISL або одного з комутаторів перетворює схему в каскад з усіма що випливають.

Сітка (mesh). Буває **Full Mesh** – коли кожний комутатор з'єднується з кожним. Характерно високою надійністю, продуктивністю й ціною. Кількість портів, необхідна під міжкомутаторні зв'язку, з додаванням кожного нового комутатора в схему росте експоненційно. При певній конфігурації просто не залишиться портів під вузли – усе будуть зайняті під ISL. **Partial Mesh** – будь-яке хаотичне об'єднання комутаторів.

Центр/периферія (Core/Edge) – близька до класичної топології LAN, але без рівня розподілу. Нерідко сховища підключаються до Core-комутаторів, а сервери – до Edge. Хоча для сховищ може бути виділений додатковий шар (tier) Edge-комутаторів. Також і сховища й сервери можуть бути підключені в один комутатор для підвищення продуктивності й зниження часу відгуку (це називається локалізацією). Така топологія характеризується гарною масштабованістю й керованістю.

Зонінг (зонування, zoning)

Ще одна характерна для SAN технологія. Це визначення пар ініціатор-таргет. Тобто яким серверам до яких дискових ресурсів можна мати доступ, щоб не вийшло, що всі сервери бачать всі можливі диски. Досягається це в такий спосіб:

– обрані пари додаються в попередньо створені на комутаторі зони (zones);

– зони містяться в набори зон (zone set, zone config), створені там же;

– набори зон активуються у фабриці.

Для первісного поста по темі SAN, думаю, досить. Прошу прощення за різномасті картинки – самому намалювати на роботі поки немає можливості, а будинку ніколи. Була думка намалювати на папері й сфотографувати, але вирішив, що краще так.

Перелічимо базові рекомендації із проектування фабрики SAN:

– Проектувати структуру так, щоб між двома кінцевими пристроями було не більше трьох комутаторів.

– Бажано щоб фабрика складалася не більш ніж з 31 комутатора.

– Варто задавати Domain ID вручну перед уведенням нового комутатора у фабрику – поліпшує керованість і допомагає уникнути проблем однакових Domain ID, у випадках, наприклад, перепідключення комутатора з однієї фабрики в іншу.

– Мати кілька рівноцінних маршрутів між кожним пристроєм зберігання й ініціатором.

– У випадках невизначених вимог до продуктивності виходити зі співвідношення кількості Nx-Портів (для кінцевих пристроїв) до кількості ISL-Портів як 6:1 (рекомендація EMC) або 7:1 (рекомендація Brocade). Дане співвідношення називається перепідпискою (oversubscription).

1. Рекомендації з зонінгу:

– використовувати інформативні імена зон і зон-сетів;

– використовувати WWPN-зонінг, а не Port-based (заснований на адресах пристроїв, а не фізичних портів конкретного комутатора);

– кожна зона – один ініціатор;

– чистити фабрику від «мертвих» зон.

2. Мати резерв вільних портів і кабелів.

3. Мати резерв устаткування (комутатори). На рівні сайту – обов'язково, можливо на рівні фабрики.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

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

У даному розділі проведемо огляд технологій SAN. Завдання системи зберігання даних – це зберігання й надання доступу до даних із заданою продуктивністю. З даних же ми й почнемо.

Дані

Тип даних

Що за дані ми плануємо зберігати? Дуже важливе питання, що може викреслити дуже багато які SAN навіть із розгляду. Наприклад, планується зберігання відеозаписів і фотографій. Відразу можна викреслювати системи, розраховані під випадковий доступ малим блоком, або системи з фірмовими фішками в компресії / дедупликації. Це можуть бути просто чудові системи, нічого поганого не хочемо сказати. Але в цьому випадку їхні сильні сторони або стануть навпроти слабкими (відео й фото не компресуються) або просто значно збільшать вартість системи.

І навпаки, якщо цільове використання навантажена транзакційна СУБД, те чудові потокові системи під мультимедіа, здатні видавати гігабайти в секунду, будуть поганим вибором.

Обсяг даних

Скільки даних ми плануємо зберігати? Кількість завжди переростає в якість, це не потрібно забувати ніколи, особливо в наш час експонентного росту обсягу даних. Системи петабайтного класу вже не рідкість, але чим більше петабайт обсягу, тим більше специфічної стає система, тим менш буде доступно звичної функціональності систем з випадковим доступом малого й середнього

обсягу. Банально тому що одні тільки таблиці статистики доступу по блоках стають більше доступного обсягу оперативної пам'яті на контролерах. Не говорячи вже про компресію / тиринг. Припустимо, ми хочемо перемкнути алгоритм компресії на могутніший і пережати 20 петабайт даних. Скільки це займе: півроку, рік?

З іншого боку, навіщо городити город, якщо зберігати й обробляти треба 500 ГБ даних? Усього 500. Побутові SSD (з низьким DWPD) подібного обсягу коштують усього нічого. Навіщо для цього будувати фабрику Fiber Channel і купувати зовнішню SAN високого класу вартістю в чавунний міст?

Який відсоток від загального обсягу гарячі дані? Наскільки нерівномірне навантаження по обсязі даних? Саме отут може дуже допомогти технологія багаторівневого зберігання або Flash Cache, якщо обсяг гарячих даних мізерний у порівнянні із загальним. Або навпаки, при рівномірному навантаженні по всьому обсязі, що часто зустрічається в потокових системах (відеоспостереження, деякі системи аналітики) подібні технології не дадуть нічого, і лише збільшать вартість / складність системи.

ІС

Зворотна сторона даних – це інформаційна система, що використовує ці дані. ІС має набір вимог, які успадковують дані.

Вимоги по відказостійкості / доступності

Вимоги по відказостійкості / доступності даних успадковуються від використовуючої їх ІС і виражаються в трьох числах – RPO, RTO, доступність.

Доступність – частка за заданий проміжок часу, протягом якого дані доступні для роботи з ними. Виражається звичайно в кількості 9. Наприклад, дві дев'ятки в рік означає, що доступність дорівнює 99%, або інакше допускається 95 годин неприступності в рік. Три дев'ятки – 9,5 годин у рік.

RPO / RTO – це показники не сумарні, а на кожний інцидент (аварію), на відміну від доступності.

RPO – обсяг загублених при аварії даних (у годинниках). Наприклад, якщо відбувається резервне копіювання раз доба, то RPO = 24 години. Тобто при аварії й повній втраті SAN можуть бути загублені дані обсягом до 24 годин (з моменту резервної копії). Виходячи із заданого для ІС RPO, наприклад, пишеться регламент резервного копіювання. Також, виходячи з RPO, можна зрозуміти наскільки потрібна синхронна / асинхронна реплікація даних.

RTO – час відновлення сервісу (доступу до даних) після аварії. Виходячи із заданого значення RTO ми можемо чи зрозуміти потрібний метрокластер, або досить односпрямованої реплікації. Чи потрібна багатоконтролерна SAN hi end класу – теж.

Вимоги по продуктивності

Незважаючи на те, що це цілком очевидне питання, з ним те саме й виникає більшість труднощів. Залежно від того, є у вас уже яка те чи інфраструктура ні й будуть будуватися шляхи збору необхідної статистики.

У вас уже є SAN і ви шукайте їй заміну або хочете придбати ще одну для розширення. Тут все просто. Ви розумієте, які сервіси у вас уже є і які ви плануєте впроваджувати в найближчій перспективі. Ґрунтуючись на поточних сервісах ви маєте можливість зібрати статистику по продуктивності. Визначитися з поточною кількістю IOPS і нинішніх затримках – які ці показники й чи вистачає їх для ваших завдань? Зробити це можна як на самій системі зберігання даних, так і з боку хостів, які до неї підключені.

Причому дивитися потрібно не просто поточне навантаження, а за який те період (краще місяць). Подивитися які максимальні піки в денний час, яку навантаження створює резервне копіювання й т.д. Якщо ваша SAN або ПЗ до неї не дає вам повний набір цих даних, можна скористатися безкоштовним RRDtool, що вміє працювати з більшістю найбільш популярних SAN і комутаторів і зможе надати вам докладну статистику по продуктивності. Також варто дивитися навантаження й на хостах, які працюють із даної SAN, по конкретних віртуальних машинах або що конкретно у вас працює на даному хості.

Варто окремо відзначити, що якщо затримки на томі й датасторі, що перебуває на даному томі відрізняються досить сильно – варто звернути увагу на вашу SAN-мережу, висока ймовірність, що з нею є проблеми й перш ніж здобувати нову систему, варто розібратися із цим питанням, адже дуже висока ймовірність збільшення продуктивності поточної системи.

Ви будуєте інфраструктуру з нуля, або здобуваєте систему під якийсь новий сервіс, про навантаження якого ви не в курсі. Отут є кілька варіантів: поспілкуватися з колегами на профільних ресурсах, щоб спробувати довідатися й спрогнозувати навантаження, звернутися до інтегратора, у якого є досвід впровадження подібних сервіс і який зможе розрахувати навантаження за вас. І третій варіант (звичайно самий складний, особливо якщо це стосується самописних або рідких застосунків) спробувати з'ясувати вимоги до продуктивності в розроблювачів системи.

І, увага, самий правильний варіант із погляду практичного застосування – це пілот на поточному устаткуванні, або устаткуванні наданому для тесту вендором / інтегратором.

Спецвимоги

Спецвимоги – все те, що не підпадає під вимоги про продуктивність, відказостійкості й функціональності по безпосередній обробці й наданню даних.

Одним з найпростіших спецвимог до системи зберігання даних можна назвати “відчужувані носії інформації”. І відразу стає зрозуміло, що дана система зберігання даних повинна містити в собі стрічкову бібліотеку або просто стример, на який скидається резервна копія. Після чого спеціально навчена людина підписує стрічку й гордо несе її в спеціальний сейф.

Інший приклад спецвимоги – захищене протиударне виконання.

Де

Другої головної складової у виборі тої або інший SAN є інформація про те, ДЕ буде стояти ця SAN. Починаючи від географії або кліматичних умов, і закінчуючи персоналом.

Замовник

Для кого планується дана SAN? Питання має під собою наступні підстави:

Держзамовник / комерційний.

Комерційний замовник не має ніяких обмежень, і не зобов'язаний навіть тендери проводити, крім як по власних внутрішніх регламентах.

Держзамовник – справа інше. 44 ФЗ та інші принадності з тендерами й ТЗ, які можуть бути оскаржені.

Замовник під санкціями

Ну отут питання дуже простій – вибір обмежується тільки доступними для даного замовника пропозиціями.

Внутрішні регламенти / дозволені до закупівлі вендори / моделі

Питання теж украй просте, але про нього треба пам'ятати.

Де фізично

У даній частині ми розглядаємо всі питання з географією, каналами зв'язку, і мікрокліматом у приміщенні розміщення.

Персонал

Хто буде працювати з даної SAN? Це не менш важливо, чим те, що SAN безпосередньо вміє.

Наскільки б не була перспективна, крута й чудова SAN від вендора А, змісту ставити її напевно небагато, якщо персонал уміє працювати тільки з вендором В, і подальших закупівель і постійного співробітництва з А не планується.

І зрозуміло, зворотна сторона питання – наскільки доступний у даній географічній локації підготовлений персонал безпосередньо в компанії й потенційно на ринку праці. Для регіонів може мати значний сенс вибір SAN із простими інтерфейсами або можливістю віддаленого централізованого керування. Інакше в який те момент може стати болісно боляче. Інтернет повний історій новий співробітник, що як прийшов, учорашній студент, наконфигурував такого, що вся контора полягла.

Оточення

Ну й розуміє немаловажне питання – у якому оточенні буде працювати дана SAN.

– Що з електроживленням / охолодженням?

– Яке підключення?

– Куди вона буде змонтована?

– І т.д.

Найчастіше дані питання вважаються самі собою що розуміють і особливо не розглядаються, але іноді саме вони можуть перевернути все з точністю до навпаки.

Вендор

На сьогодні (середина 2026) український ринок SAN можна поділити на умовні 5 категорій:

1. Вищий дивізіон – заслужені компанії із широкою лінійкою від найпростіших дискових полиць до hi-end (HPE, DellEMC, Hitachi, NetApp, IBM / Lenovo)

2. Другий дивізіон – компанії з обмеженою лінійкою, нишеві гравці, серйозні вендори SDS або новачки, що піднімаються (Fujitsu, Datacore, Infinidat, Huawei, Pure і тд)

3. Третій дивізіон – нишеві рішення в ранзі low end, дешевий SDS, наколінні поделия на ceph і інших відкритих проектах (Infortrend, Starwind і тд)

4. SOHO сегмент – малі й зверхмалі SAN рівня будинок/малий офіс (Synology, QNAP і тд)

5. Імпортозаміщенні SAN – сюди входять як залізо першого дивізіону з переклеєними лейблами, так і рідкі представники другого (RAIDIX, дамо їм авансом другий), але в основному це третій дивізіон (Aerodisk, Baum, Depo і т.д.)

Розподіл досить умовний, і зовсім не означає, що третій або SOHO сегмент погані і їх не можна використовувати. У специфічних проектах із чітко певним набором даних і профілем навантаження вони можуть відробити дуже

добре, далеко перевершуючи перший дивізіон по співвідношенню ціна/якість. Важливо спочатку визначитися із завданнями, перспективами росту, необхідною функціональністю – і тоді Synology буде служити вам вірою й правдою, а волосся стануть м'якими й шовковистими.

Один з немаловажних факторів при виборі вендора – поточне середовище. Скільки і яких SAN у вас уже є, з якими SAN уміють працювати інженери. Чи потрібний вам ще один вендор, ще одна точка контакту, чи будете ви мігрувати поступово все навантаження з вендора A на вендора B?

Не слід плодити сутності понад необхідні.

iSCSI / FC / File

З питання протоколів доступу немає єдиної думки серед інженерів, а суперечки нагадують більше теологічні дискусії, чим інженерні. Але в цілому можна відзначити наступні пункти:

FCoE скоріше мертвий, чим живий.

FC vs iSCSI. Одна із ключових переваг FC в 2020 перед IP SAN, виділена фабрика під доступ до даних, нівелюється виділеної IP мережею. Глобальних переваг в FC перед IP мережами немає й на IP можна будувати SAN будь-якого рівня навантаження, аж до систем для важких СУБД для АБС великого банку. З іншого боку, смерть FC пророкують уже не перший рік, але цьому постійно що те заважає. На сьогодні, наприклад, деякі гравці на ринку SAN активно розвивають стандарт NVMEo. Чи розділить він доля FCoE – покаже час.

Файловий доступ також не є чим те невартим уваги. NFS / CIFS прекрасно показують себе в продуктивних середовищах і при правильному проектуванні мають не більше дорікань, чим блокові протоколи.

Гібридні / All Flash Array

Класичні SAN бувають 2 видів:

1. AFA (All Flash Array) – системи, оптимізовані для використання SSD.

2. Гібридні – що дозволяють використовувати як HDD, так і SSD або їхнє сполучення.

Головна їхня відмінність – підтримувані технології ефективності зберігання й максимальний рівень продуктивності (високі показники IOPS і низькі затримки). І ті й інші системи (у більшості своїх моделей, не вважаючи low-end сегмент) можуть працювати як блокові пристрої, так і файлові. Від рівня системи залежить і підтримуваний функціонал, і в молодших моделей, він найчастіше урізаний до мінімального рівня. На це варто звертати увагу, коли ви вивчаєте характеристики конкретної моделі, а не просто можливості всієї лінійки в цілому. Так само, звичайно, від рівня систему залежать і її технічні характеристики, такі як процесор, обсяг пам'яті, кешу, кількість і типи портів і т.д. З погляду ж керування, AFA від гібридних (дискових) систем відрізняються лише в питаннях реалізації механізмів роботи з SSD накопичувачами, і навіть якщо ви використовуєте SSD у гібридній системі, це зовсім не виходить, що ви зможете одержати рівень продуктивності на рівні AFA системи. Так само в більшості випадків inline механізми ефективного зберігання на гібридних системах відключені, а їхнє включення веде до втрати в продуктивності.

Спеціальні SAN

Крім SAN загального призначення, орієнтованих насамперед на оперативну обробку даних, існують спеціальні SAN із ключовими принципами, у корені відрізняющимися від звичних (низька затримка, багато IOPS).

Медіа

Дані системи призначені під зберігання й обробку медіа-файлів, що відрізняються більшим розміром. Відповідно затримка стає практично неважливою, а на перший план виходить здатність віддавати й приймати дані широкою смугою в багато паралельних потоків.

Дедуплікуючі SAN для резервних копій

Оскільки резервні копії відрізняються рідкої у звичайних умовах подібністю друга на друга (середня резервна копія відрізняється від учорашньої на 1-2%), даний клас систем украй ефективно впаковує записані на них дані в

рамках досить невеликої кількості фізичних носіїв. Наприклад, в окремих випадках коефіцієнти компресії даних можуть досягати 200 до 1.

Об'єктні SAN

У цих SAN немає звичних томів із блоковим доступом і файлових куля, а найбільше вони нагадують величезну базу даних. Доступ до об'єкта, що зберігається в подібній системі, здійснюється по унікальному ідентифікаторі, або по метаданим (наприклад всі об'єкти формату JPEG, з датою створення між XX-XX-XXXX і YY-YY-YYYY).

Compliance системи

Не так часто зустрічаються в Україні на сьогодні, але згадати про них коштує. Призначення таких SAN – гарантоване зберігання даних для відповідності політикам безпеки або вимогам регуляторів. У деяких системах (наприклад EMC Centera) була реалізована функція заборони на видалення даних – як тільки ключ повернуть і система перейшла в даний режим, ні адміністратор, ні хто або інший фізично не можуть видалити вже записані дані.

Фірмові технології

Flash cache

Flash Cache – загальна назва для всіх фірмових технологій використання флеш-пам'яті в якості кешу другого рівня. При використанні флеш кешу SAN як правило розраховується для забезпечення з магнітних дисків сталого навантаження, у той час як пікову обслуговує кеш.

При цьому необхідно розуміти профіль навантаження й ступінь локалізації звертань до блоків томів зберігання. Флеш кеш – технологія для навантажень із високою локалізацією запитів, і практично незастосовна для рівномірно навантажених томів (як наприклад для систем аналітики).

На ринку доступні дві реалізації флеш кешу:

– Read Only. У цьому випадку кешуються тільки дані на читання, а запис проходить відразу на диски. Деякі виробники, як наприклад, NetApp, уважають

що запис на їх SAN проходить і так оптимальним образом, і кеш ніяк не допоможе.

– Read/Write. Кешується не тільки читання, але й запис, що дозволяє буферізувати потік і знизити вплив RAID Penalty, а як наслідок підвищити загальну продуктивність для SAN з не таким оптимальним механізмом запису.

Tiering

Багаторівневе зберігання (тиринг) – технологія об'єднання в єдиний дисковий пул рівнів з різною продуктивністю, як наприклад SSD і HDD. У випадку яскраво вираженої нерівномірності звертань до блоків даних система зможе автоматично відбалансувати блоки даних, перемістивши навантажені на високопродуктивний рівень, а холодні, навпаки, на більше повільний.

Гібридні системи нижнього й середнього класів використовують багаторівневе зберігання з переміщенням даних між рівнями за розкладом. При цьому розмір блоку багаторівневого зберігання в кращих моделей становить 256 МБ. Дані особливості не дозволяють уважати технологію багаторівневого зберігання технологією підвищення продуктивності, як помилково вважається багатьма. Багаторівневе зберігання в системах нижнього й середнього класів – це технологія оптимізації вартості зберігання для систем з вираженою нерівномірністю навантаження.

Snapshot

Скільки ми б не говорили про надійність SAN, існує безліч можливостей втратити дані, що не залежать від апаратних проблем. Це можуть бути як віруси, хакери або будь-яке інше, ненавмисне видалення/псування даних. Із цієї причини, резервне копіювання продуктивних даних є невід'ємною частиною роботи інженера.

Снапшот – це знімок тому на який те момент часу. При роботі більшості систем, таких як віртуалізація, БД і тд. нам необхідно зняти такий знімок, з якого ми будемо копіювати дані в резервну копію, при цьому наші ІС зможуть спокійно продовжувати працювати із цим томом. Але варто пам'ятати – не всі снапшоти

однаково корисні. У різних вендорів, різні підходи до створення снапшотів, пов'язані з їхньою архітектурою.

CoW (Copy-On-Write). При спробі запису блоку даних його оригінальний уміст копіюється в спеціальну область, після чого запис проходить нормально. У такий спосіб запобігає ушкодження даних усередині снапшоту. Природно всі ці «паразитні» маніпуляції з даними викликають додаткове навантаження на SAN і із цієї причини вендори з подібною реалізацією не рекомендують використовувати більше десятка снапшотів, а на високонавантажених томах не використовувати їх взагалі.

RoW (Redirect-on-Write). У цьому випадку, оригінальний том натурально заморожується, а при спробі запису блоку даних SAN пише дані в спеціальну область у вільному просторі, змінюючи місце розташування даного блоку в таблиці метаданих. Це дозволяє зменшити кількість операцій перезапису, що в підсумку нівелює падіння продуктивності й знімає обмеження на снапшоти і їхня кількість.

Снапшоти бувають також двох типів стосовно застосунків:

Application consitent. У момент створення снапшоту SAN смикає агента в операційній системі споживача, що примусово скидає дискові кеші з пам'яті на диск і змушує зробити цей додаток. У цьому випадку при відновленні зі снапшоту дані будуть консистентні.

Crash consistent. У цьому випадку нічого подібного не відбувається й снапшот створюється як є. У випадку відновлення з такого снапшоту картина ідентична як якби раптово відключилося живлення й можливе деяка втрата даних, що зависли в кешах і так і не дійшли до диска. Такі снапшоти простіше в реалізації й не викликають осідання продуктивності в додатках, але менш надійні.

Навіщо потрібні снапшоти на системах зберігання даних?

– Безагентне резервне копіювання прямо з SAN.

– Створення тестових середовищ на основі реальних даних.

– У випадку з файловими SAN може використовуватися для створення середовищ VDI через використання снапшотів SAN замість гіпервізору.

– Забезпечення низьких RPO шляхом створення снапшотів за розкладом із частотою значно вище частоти резервного копіювання.

Cloning

Клонування тому – працює по аналогічному принципі, що й снапшоти, але служить не просто для читання даних, а для повноцінної роботи з ними. Ми маємо можливість одержати точну копію нашого тому, з усім даними на ньому, не роблячи фізичної копії, що дозволить заощадити місце. Звичайне клонування томів використовується або в Test&Dev або якщо ви хочете перевірити працездатність яких те відновлень на вашої ІС. Клонування дозволить зробити це максимально швидко й економічно з погляду дискових ресурсів, тому що записані будуть тільки змінені блоки даних.

Реплікація / журналювання

Реплікація – механізм створення копії даних на інший фізичної SAN. Звичайно існує фірмова технологія в кожного вендора, що працює тільки усередині власної лінійки. Але також є й сторонні рішення, у тому числі працюючі на рівні гіпервізора, як наприклад VMware vSphere Replication.

Функціональність фірмових технологій і зручність їхнього використання звичайно сильно перевершують універсальні, але виявляються незастосовні, коли наприклад необхідно робити репліку з NetApp на HP MSA.

Реплікація ділиться на два підвиди:

Синхронна. У випадку синхронної реплікації операція запису пересилається на другу SAN негайно й не підтверджується виконання доти, поки віддалена SAN не підтвердить. За рахунок цього росте затримка доступу, але зате ми маємо точну дзеркальну копію даних. Тобто RPO = 0 для випадку втрати основний SAN.

Асинхронна. Операції запису виконуються тільки на головної SAN і підтверджуються негайно, паралельно накопичуючись у буфері для пакетної

передачі на віддалену SAN. Подібний вид реплікації актуальний для менш коштовних даних, або для каналів низької пропускної здатності або маючих високу затримку (характерно для відстаней понад 100 км). Відповідно RPO = частоті пакетного відправлення.

Найчастіше разом з реплікацією існує механізм **журналювання** дискових операцій. У цьому випадку виділяється спеціальна область для журналювання й зберігаються операції запису певної глибини за часом, або обмежені обсягом журналу. Для окремих фірмових технологій, як наприклад EMC RecoverPoint, існує інтеграція із системним ПЗ, що дозволяє прив'язати певні закладки на конкретний запис у журналі. Завдяки цьому можливо відкотити стан тому (або створити клон) не просто на 23 квітня 11 годин 59 секунд 13 мілісекунд, а на момент, що передував “DROP ALL TABLES; COMMIT”.

Metro cluster

Метро кластер – технологія, що дозволяє створити двонаправлену синхронну реплікацію між двома SAN таким чином, що з боку ця пара виглядає як одна SAN. Застосовується для створення кластерів з географічно рознесеними плечима на метро-відстанях (менш 100 км).

На прикладі використання в середовищі віртуалізації метрокластер дозволяє створити датастор з віртуальними машинами, доступний на запис відразу із двох датацентрів. У цьому випадку створюється кластер на рівні гіпервізорів, що складається з хостів у різних фізичних датацентрах, підключений до даного датастору. Що дозволяє зробити наступне:

– Повна автоматизація процесу відновлення після смерті одного з датацентрів. Без яких або додаткових засобів всі ВМ, що працювали в померлому датацентре, будуть автоматично перезапущені в що остались. RTO = таймаут кластера високої доступності (15 секунд для VMware) + час завантаження операційної системи й старту сервісів.

– Disaster avoidance або, по-російському, запобігання катастроф. Якщо заплановані роботи з електроживлення в датацентрі 1, то ми заздалегідь, до

початку робіт, маємо можливість мігрувати все важливе навантаження в датацентр 2 нон стоп.

Віртуалізація

Віртуалізація SAN – це технічно використання томів з інший SAN як диски. SAN-віртуалізатор може просто прокинути чужий тім до споживача як свій, попутно дзеркалював його на ще одну SAN, або навіть створити RAID із зовнішніх томів.

Класичні представники в класі віртуалізації SAN – це EMC VPLEX і IBM SVC. Ну й розуміє SAN з функцією віртуалізації – NetApp, Hitachi, IBM / Lenovo Storwize.

Навіщо може знадобитися?

– Резервування на рівні SAN. Створюється дзеркало між томами, причому одна половина може бути на HP 3Par, а інша на NetApp. А віртуалізатор від EMC.

– Переїзд даних з мінімальним простоєм між SAN різних виробників. Припустимо, що дані треба мігрувати зі старого 3Par, що піде під списання, на новий Dell. У цьому випадку споживачі відключаються від 3Par, тому прокидуються під VPLEX і вже презентуються споживачам заново. Оскільки ні біта на томі не змінилося, робота триває. Тлом запускається процес дзеркалювання тому на новий Dell, а по завершенню дзеркало розбивається й 3Par відключається.

– Організація метрокластерів.

Компресія / дедупликація

Компресиия й дедупликація – це ті технології, які дозволяють вам заощаджувати дисковий простір на вашій SAN. Варто відразу згадати, що далеко не всі дані підлягають компресії й/або дедупликації в принципі, при цьому деякі типи даних тиснуться й дедуплікуються краще, а якісь – навпаки.

Компресія й дедупликація бувають 2 видів:

Inline – стиск і дедупликація блоків даних відбувається до запису цих даних на диск. У такий спосіб система тільки обчислює геш блоку й порівнює

його по таблиці із уже наявними. По-перше це виконується швидше, ніж просто запис на диск, по-друге ми не витрачаємо зайвий дисковий простір.

Post – коли ці операція проводять уже на записаних даних, які перебувають на дисках. Відповідно дані спочатку записуються на диск, а тільки потім, обчислюється геш і відбувається видалення зайвих блоків і вивільнення дискових ресурсів.

Варто сказати, що більшість вендорів використовують обидва види, що дозволяє оптимізувати ці процеси й тим самим підвищити їхня ефективність. У більшості вендорів SAN, є в наявності утиліти, які дозволяють проаналізувати ваші набори даних. Дані утиліти, працюють по тій же логіці, що реалізовано й в SAN, по цьому оцінний рівень ефективності буде збігатися. Також не варто забувати, що в багатьох вендорів є програми гарантії ефективності, які обіцяють рівень не нижче заявленого для певного (або всіх) типів даних. І не варто зневажати даною програмою, адже розраховуючи систему під свої завдання, з урахуванням коефіцієнта ефективності конкретної системи, ви можете заощадити на обсязі. Так само варто враховувати, що ці програми розраховані на AFA системи, але завдяки закупівлі меншого обсягу SSD, ніж HDD у класичних системах, це дозволить знизити їхня вартість, і якщо не зрівнятися з вартістю дискової системи, те досить сильно до неї наблизитися.

Модель

І отут ми приходимо до правильно заданого питання.

“От мені пропонують два варіанти SAN – ABC SuperStorage S600 і XYZ HyperOcean 666v4, що порадите”

Перетворюється в “От мені пропонують два варіанти SAN – ABC SuperStorage S600 і XYZ HyperOcean 666v4, що порадите?

Цільове навантаження змішані віртуальні машини VMware з контурів продуктив / тест / розробка. Тест = продуктиву. 150 ТБ на кожний з піковою продуктивністю 80 000 IOPS 8kb блоком 50% випадкового доступу 80/20

читання-запис. 300 ТБ на розробку, там 50 000 IOPS вистачить, 80 випадковий, 80 запис.

Продуктив приблизно в метрокластер RPO = 15 мінут RTO = 1 годину, розробку в асинхронну реплікацію RPO = 3 години, тест на одній площадці.

Буде 50ТБ СУБД, було б непогано для них журналювання.

У нас скрізь сервери Dell, SAN старі Hitachi, ледве справляються, плануємо ріст 50% навантаження по обсязі й продуктивності”

Як говориться, у правильно сформульованому питанні втримується 80% відповіді.

Ціни

Тепер що ж стосується цін – взагалі на SAN ціни якщо й попадаються, то звичайно це List price, від якої кожний замовник одержує індивідуальну знижку. Розмір знижки складається з великої кількості параметрів, так що пророчити, яку кінцеву ціну одержить саме ваша компанія, без запиту до дистриб'ютора просто неможливо. Але при цьому, останнім часом low-end моделі стали з'являтися у звичайних комп'ютерних магазинах, таких як, наприклад rozetka.ua або comfy.ua. У них можна відразу придбати систему, що цікавить вас, за фіксованою ціною, як будь-які комп'ютерні комплектуючі.

Але хочеться відзначити відразу, що пряме порівняння по ТВ/$ не є вірним. Якщо підходити із цієї точки зору, то найбільш дешевим рішенням буде простий JBOD + сервер, що не дасть ні тої гнучкості, ні тої надійності, які забезпечує повноцінна, двоконтролерна SAN. Це зовсім не виходить, що JBOD поганий, просто потрібно знов-таки дуже чітко розуміти – як і для яких цілей ви будете використовувати це рішення. Часто можна почути, що в JBOD нема чому ламатися, там же один бекплейн. Однак і бекплейни буває виходять із ладу. Усе ламається рано або пізно.

Порівнювати системи між собою потрібно не тільки за ціною, або не тільки по продуктивності, а по сукупності всіх показників.

Купуйте HDD тільки якщо ви впевнені, що вам потрібні HDD. Для низьких навантажень і нестисливих типів даних, в іншому випадку, варто звернути на програми гарантії ефективності зберігання на SSD, які зараз є в більшості вендорів (і вони дійсно працюють, навіть в Україні), але отут все залежить від застосунків і даних, які будуть розташовуватися на даної SAN.

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Python – високорівнева мова програмування, яку називають другою за популярністю в світі. Її використовують для розробки вебзастосунків, програмного забезпечення, машинного навчання. Python застосовують для вирішення робочих завдань у компаніях Google, Instagram, Facebook, IBM, NASA, Dropbox, Netflix та інших. Розробники цінують цю мову програмування за простоту у вивченні, ефективність та мультиплатформність.

Python – скриптова мова програмування з досить простим синтаксисом. Для розуміння достатньо порівняти принципи написання найпростішої програми, яка виводить на екран текстове повідомлення. Саме тому мова програмування Python більш доступна для новачків, а професіонали встигли адаптувати її для вирішення великої кількості завдань. Це мультиплатформне рішення, тому знання Python дає можливість працювати у різних сферах: від розробки мобільних застосунків до ігрової індустрії та штучного інтелекту.

У мови програмування динамічна типізація: є можливість передавати до функцій будь-який тип даних без попереднього вказання. Інтерпретованість дозволяє знаходити помилки у коді ще до повної збірки у робочий застосунок. При цьому Python дуже чітко дає зрозуміти, де та через що виникла помилка.

Це мова об'єктно-орієнтованого програмування (ООП). Програмне забезпечення на Python оформлене у вигляді моделей, які можуть бути зібраними у пакети. Тип та структуру кожного об’єкта можна запитати під час виконання

програми. Для кожного з об’єктів можна отримати всю інформацію щодо його внутрішньої структури. Окрім того:

– у мови логічний синтаксис, завдяки чому вихідний код легко читати та розуміти;

– гнучкість та масштабованість Python дозволяє адаптувати високорівневу логіку та розширяти складні застосунки, як тільки виникне така необхідність;

– розробка на Python у більшості випадків проходить швидше, ніж на інших мовах програмування;

– Python – інтерпретована мова програмування. Це значить, що код можна написати у будь-якому текстовому файлі на будь-якій платформі, і потім успішно запустити;

– у Python – колосальна спільнота однодумців. Тож будь-які складнощі конкретних розробників вирішуються колективно.

Проте є декілька особливостей, які можна віднести до недоліків. Це повільність (ця мова програмування хоч і універсальна, проте повільніша за інші), велика кількість ресурсів, необхідних для роботи та «прив’язаність» до системних бібліотек.

Мова програмування Python використовується у наступних сферах:

1. Розробка програмних застосунків будь-якого напрямку.

2. Розробка серверної частини мобільних застосунків (найпопулярніший напрямок).

3. Ігри. Багато сучасних ігор для комп’ютерів (наприклад, World of Tanks) частково чи повністю написані на Python.

4. Вбудовані системи для різних пристроїв. Дуже часто Python використовують для написання внутрішніх платформ управління банкоматами.

5. Скрипти та плагіни до уже реалізованих програм для автоматизації процесів чи створення інших рішень.

6. Тестування (автоматизація цього процесу).

7. Машинне навчання. – основна мова для написання алгоритмів і аналітичних застосунків у сфері Machine Learning.

Бібліотеки Python

Різні бібліотеки Python використовують для виконання конкретних завдань. Наприклад, Matplotlib підходить для відображення даних у двовимірній та тривимірній графіці. Pandas підходить для зручної роботи з даними. NumPy дозволяє створювати масиви та керувати ними. Requests використовується для веброзробки. OpenCV-Python відкриває можливості для обробки зображень з метою оптимізації систем «машинного зору».

Найвідоміші фреймворки для мови програмування Python

Фреймворки Python допомагають створити зручне та функціональне середовище для розробки. У них міститься набір інструментів, модулів та бібліотек, корисних для виконання конкретних завдань. Це значно полегшує роботу: наприклад, дає змогу не витрачати час на розписування дій, які повторюються, а використати релевантний інструмент. Тож є можливість позбутися рутинних процесів та сконцентруватися на логіці проєкту.

Серед найпопулярніших фреймворків для Python:

– Django – найстаріший та найвідоміший. Створений для реалізації великих інтерактивних проєктів;

– Pyramid – зручний у налаштуваннях, і дає можливість реалізувати складні нестандартні ідеї;

– Web2py – підходить в першу чергу для вебзастосунків і може використовуватись на будь-яких архітектурах.

Популярні Python IDE

IDE або інтегровані середовища розробки – це програмне забезпечення, яке надає розробникам необхідні інструменти для написання, редагування, тестування та налаштування коду. Для розробки на Python найчастіше використовують IDE PyCharm, IDLE, Spyder та Atom.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

В процесі розробки випускної кваліфікаційної роботи за першим (бакалаврським) 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

Стандарт Fibre Channel із пропускною здатністю 1 Гбіт/с був затверджений в 1997 році. Життєвий цикл технології 1 GFC з моменту її появи на ринку в 1998 році й до старіння в 2004-м протікав одночасно з розробкою й виводом на ринок стандарту 2 GFC, а також з першим етапом розробки 4 GFC. В 2005 році трансівери, що підтримують 2 GFC, поставлялися вже в промислових обсягах, а технологія 4 GFC була затверджена як стандарт і почала пропонуватися на ринку. Цей цикл визнання, старіння й відновлення повторюється протягом останніх 20 років.

На практиці досить часто спостерігається мирне співіснування різних технологій з різними швидкостями передачі даних. При цьому клієнтський пристрій і комутатор можуть підтримувати різні швидкості (наприклад, коли в комутаторі встановлені плати 16 GFC, а старий масив зберігання забезпечує підтримку тільки 4 GFC).

З'єднання Fibre Channel між серверами й системами зберігання організуються з використанням високошвидкісного волоконно-оптичного кабелю. Пропускна здатність каналів росте в геометричній прогресії зі знаменником 2: щораз швидкість передачі Fibre Channel подвоюється в порівнянні з попередньої.

Порти Fibre Channel підтримують трансівери з різною дальністю й швидкістю передачі даних. Відповідно до нормативних вимог поточний стандарт трансіверів повинен забезпечувати зворотну сумісність із двома попередніми поколіннями (наприклад, трансівери 8 GFC сумісні з 4 GFC і 2 GFC).

У цей час на ринку домінують два типи трансіверов: малий формат SFP (з дуплексним коннектором LC) і Quad SFP (QSFP з 8/ 12-волоконним коннектором

MPO). У тих конфігураціях, де комутатори розташовуються поруч або сервери встановлюються в безпосередній близькості від прикордонного комутатора SAN, можна задіяти мідні кабелі прямого підключення (Direct Attach Cable, DAC), що підтримують Fibre Channel (як правило, їхня довжина не перевищує 10 м).

Модулі SFP, представлені зараз у дуже різноманітному асортименті, використовуються в більшості трансіверів від 1 до 32 GFC. Споконвічно ці модулі підтримували пропускну здатність до 5 Гбіт/с або нижче, тоді як нові варіанти розроблялися для підтримки застосунків 4 GFC і більше пізніх. Наприклад:

– SFP+ для 8 GFC, 10 GFC, 10 Gigabit Ethernet (10Gb) і 16 GFC;

– SFP28 для 32 GFC (і 25 Gb).

За попередньою інформацією, модулі SFP28 будуть підтримувати й більше високі швидкості (50 і 64 GFC), але це ще вимагає підтвердження. Специфікації нових поколінь модулів SFP передбачають більше високу цілісність сигналу в порівнянні з попередніми, щоб забезпечити відповідність зростаючим вимогам до пропускної здатності.

Модулі QSFP уперше почали застосовуватися в InfiniBand і високошвидкісних конфігураціях Ethernet (40 Gb). У них є чотири паралельних оптичних або електричних канали, що забезпечують високошвидкісну передачу 100 Gb (4 x 25 Гбіт/с) і 128 GFC (4 x 32 Гбіт/с).

Споконвічно модулі QSFP підтримували швидкість передачі до 4 ? 5 Гбіт/с, а нові варіанти розроблялися для застосунків Fibre Channel, що перевищують 16 GFC. Модулі QSFP використовуються в багатьох трансіверах у конфігураціях від 40 Gb до 128 GFC. Подібно дуплексним модулям SFP з послідовною передачею даних на швидкості до 32 Гбіт/с, нові покоління QSFP, QSFP+ і QSFP28 відповідають підвищеним вимогам до швидкості передачі даних:

– QSFP+ для 128 GFC (у тому числі для високоплотних конфігурацій 4x32 GFC);

– QSFP28 для перспективних рішень Ethernet / Fibre Channel (із цільовими швидкостями 200 Gb і 256 GFC).

Для скорочення витрат там, де це можливо, при побудові каналів SAN Fibre Channel у ЦОД використовують короткохвильові трансівери в сполученні із багатомодовим волоконно-оптичним кабелем (Multimode Fiber, MMF). Одномодовий кабель (Singlemode Fiber, SMF) і трансівери (як правило, така комбінація обходиться дорожче) застосовуються для з'єднання комутаторів (Interswitch Link, ISL) у різних приміщеннях/будинках ЦОД.

Для організації з'єднання 32 GFC довжиною 100 м потрібний кабель MMF другого покоління, оптимізований для лазерної передачі (OM4).

На більше довгих дистанціях використовуються довгохвильові трансівери й одномодовий кабель. Трансівер SMF, що відповідає специфікаціям Fibre Channel Physical Interface, забезпечує передачу даних на відстань до 10 км, а більше дешеві варіанти – до 2 км.

Для підтримки 128 GFC у трансіверах 128 GFC-PSM4 буде застосовуватися та ж технологія, що й у недорогих паралельних 4-канальних одномодових трансіверах (PSM4) з дальністю дії 500 м для Ethernet зі швидкістю 100 Гбіт/с. Аналогічним образом, технологія CWDM4, використовувана при побудові каналів Ethernet довжиною 2 км із пропускною здатністю 100 Гбіт/с, буде задіяна в рішеннях 128 GFC. Очікується, що для забезпечення ISL-з'єднання між модулями 128 G QSFP28 і 32 G SFP28 виробники комутаторів стануть застосовувати технологію PSM4.

Кабельна інфраструктура Fibre Channel

У відповідності зі стандартами IEC і TIA під лінією (link) розуміється постійна волоконно-оптична кабельна інфраструктура, до якої підключається активне устаткування. До лінії не ставляться сполучні шнури, використовувані для підключення активних мережевих пристроїв у розподільних зонах, і кроссировочні шнури.

Тестування ліній у стандартах ISO/IEC і TIA являє собою перевірку характеристик фіксованих (постійних) сегментів прокладеного кабелю. Успішне проходження тестування гарантує відповідність ліній вимогам стандартів і їхнє стійке функціонування при використанні якісних комутаційних шнурів.

Стандарт Fibre Channel ANSI T11 визначає вимоги до каналів, що утворять SAN. Кожний з рівнів Fibre Channel Physical Media Dependent (PMD) має різні характеристики залежно від виду кабельної проводки. Для проектування гнучкої й надійної кабельної системи передбачається набір «регулювань», маніпулюючи якими можна домогтися стабільного функціонування каналів SAN.

У бюджет потужності ліній для застосунків Ethernet і Fibre Channel не включаються втрати при проходженні сигналу через з'єднувачі, які підключають до устаткування на обох кінцях тракту. Вони враховуються в бюджеті потужності лінії як різниця між мінімальною потужністю передавача й мінімальною чутливістю приймача. Число з'єднувачів у каналі дорівнює загальній кількості сполучених пар. З'єднувачі, що підключаються до оптичних трансіверам, сполученими парами не вважаються.

Оптичний бюджет для застосунків

Загальний бюджет потужності оптичної лінії визначається стандартом застосунків (наприклад, Ethernet) і залежить від втрат потужності й загальної довжини каналу. Як правило, більша частина оптичних втрат невелика й не перевищує 0,3 дб. При цьому до двох найбільш істотних факторів, що обмежують довжина каналу, ставляться міжсимвольна інтерференція (Inter-Symbol Interference, ISI), що залежить від ширини смуги пропущення оптичного волокна, і втрати, внесені з'єднувачами (Connector Insertion Loss, IL). Їхній вплив багато в чому залежить від якості проектування й методів, використовуваних при побудові й тестуванні каналу.

Два джерела втрат – IL (втрати в сполучених розніманнях) і загасання (ослаблення лазерного сигналу усередині оптичного волокна). IL – це критично важливий параметр, що визначає характеристики каналу. Втрати в лінії залежать від кількості з'єднань оптичного волокна й індивідуальних втрат у сполучених

парах рознімань. Загальні втрати в лінії обчислюються як сума втрат у з'єднаннях і загасання, що залежить від довжини волокна.

Найчастіше загасання сигналу у волокні й втрати потужності, що виникають внаслідок міжсимвольної інтерференції, можна варіювати для компенсації втрат у з'єднаннях, але при розрахунках однаково варто проявляти обережевість і знаходити компромісний варіант.

Як приклад розглянемо з'єднання 16 GFC OM4 (M5F в Fibre Channel) довжиною 50 м. Це третина від максимально припустимої специфікаціями дальності в 150 метрів. У такої лінії параметр ISI виявляється значно менше, ніж у лінії довжиною 150 м. Таким чином, загальні втрати в з'єднаннях можуть бути більше – до 2,4 дБ. Або ж втрати ISI можна скоротити за рахунок використання волокна OM4 з більше широкою смугою пропущення.

3.2 Розробка структурної схеми

Структурована кабельна модель для Fibre Channel

Структурована кабельна система дозволяє з мінімальними ризиками здійснювати переміщення, додавання й зміни (Moves, Adds And Changes, MAC), а також ефективно управляти з'єднаннями між ISL, серверами й системами зберігання. Її застосування створює умови для керування змінами в ЦОД із забезпеченням подальшого розвитку й переходу на нові технології. Вибираючи таку кабельну модель, можна розгортати комутатори високої щільності, не жертвуючи керованістю.

З появою кожного чергового покоління устаткування Fibre Channel і збільшенням швидкості передачі дані резерви потужності неухильно знижуються. Важливо також, щоб у середовищах високої щільності забезпечувалася можливість підключення «кожного до кожному». Це може потенційно привести до перевищення доступного бюджету потужності, оскільки в таких каналах з'являється усе більше з'єднань.

Ціль структурованої кабельної моделі полягає в тому, щоб всі порти в схемі перехресних з'єднань «кожного з кожним» служили логічним поданням портів комутаторів, серверів і засобів зберігання в ЦОД. Такий крос називається головною розподільною областю (Main Distribution Area, MDA) у стандарті TIA 942 і центральною комутаційною зоною в специфікаціях Fiber Transport Systems (FTS) компанії IBM. Перехресні з'єднання між кінцевими точками виконуються за допомогою сполучних кабелів між комутаційними панелями в MDA. При використанні такого підходу робота активного устаткування може бути порушена тільки в результаті аварії або технологічних змін.

Структурована кабельна система базується на стандартах, що гарантує керованість і масштабованість переміщень, додавань і змін (з використанням TIA 606) для підтримки майбутніх технологій Fibre Channel.

На рис. 3.1 показане підключення з'єднувачів багатоволоконних магістральних кабелів (Multi-Fiber Push On, MPO) до касетних модулів MPO/LC. Магістральні кабелі з розніманнями MPO містять кратне 12 число волокон і можуть прокладатися між комутаційними панелями/шафами в ЦОД на великі відстані. Такі магістральні кабелі підключаються до зазначеного вище модулям або до інших кабелів з розніманнями MPO через панелі для сполучення MPO. У статичному середовищі, де не потрібна комутація, як варіант можуть використовуватися кабельні розгалужувачі з коннекторами LC для підключення до магістрального кабелю з розніманнями MPO через панель для сполучення MPO.

Касетні модулі MPO/LC високої щільності в комутаційних панелях/корпусах мають до 72 портів LC або MPO на 1 RU (rack unit).

Розгалужувачі MPO-LC дозволяють підключити трансівери SFP+ у комутаторі високої щільності SAN Director (з розніманнями LC) до волоконно-оптичної магістралі, оснащеної коннекторами MPO.

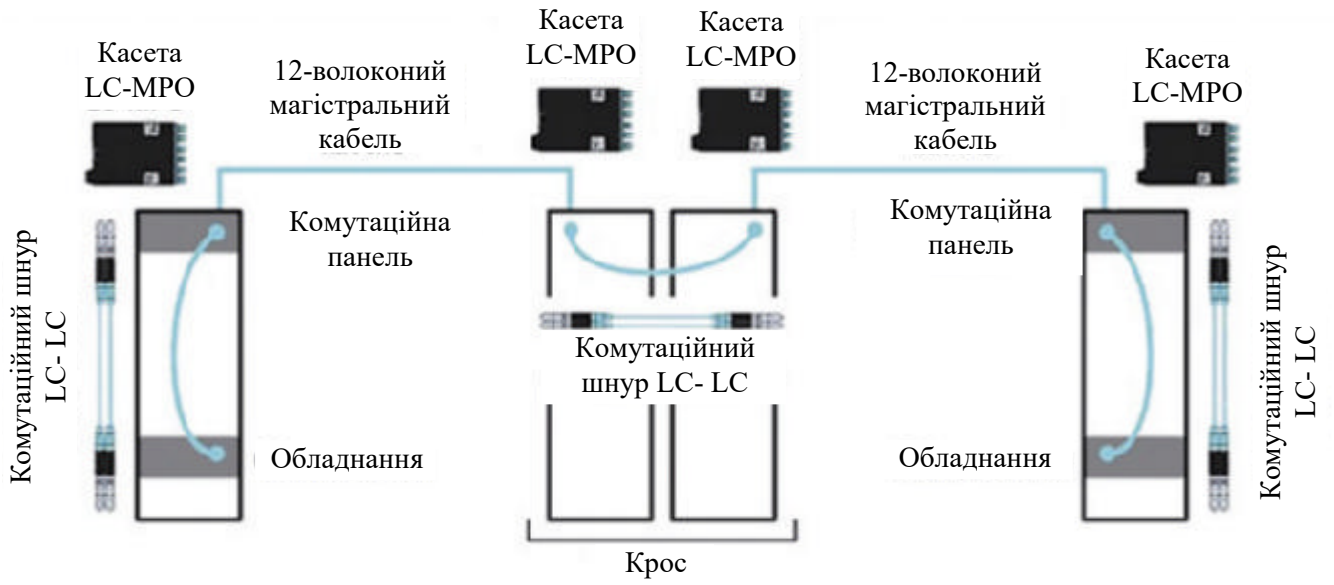

Рисунок 3.1 – Структурна схема системи

Модульні касети й панелі оптичних адаптерів MPO підтримують магістральні кабелі з коннекторами MPO. Магістральний кабель заводиться в задню частину такого компонента, а до його фронтальної частини підключаються комутаційні шнури LC або MPO. Модульна архітектура plug-and-play мінімізує загальну вартість інсталяції й забезпечує максимальну масштабованість.

Модульні касетні розподільні системи підтримують 72 порту LC (із двоволоконним дуплексом). Стійка/шафа з такою системою може вміщати до 3024 портів LC. При побудові рішень настільки високої щільності необхідно забезпечити керованість і мінімізувати ризик ушкодження суміжних (потенційно активних) ланцюгів в областях активної комутації шляхом вибору спеціально призначених для цього розподільних систем.

Типові кабельні системи SAN у ЦОД

Канали можна згрупувати на верхньому рівні по кількості пар сполучених рознімань. З'єднання звичайно здійснюються в комутаційних панелях або шафах, які можуть мати різне призначення (консолідація, відображення портів, кроссировка), а панелі бувають або статичними (у точці консолідації кабелю під підлогою з невеликим числом MAC), або динамічними (у кросах, що

забезпечують виконання операцій MAC «кожний з кожним»). Найчастіше з'єднання Fibre Channel розгортаються одним із трьох способів:

– точка-точка: пряме з'єднання між портами Fibre Channel за допомогою волоконно-оптичного кабелю без рознімних з'єднань у каналі;

– міжз’єднання: один постійний кабельний сегмент, термінований на кожному з кінців на комутаційній панелі, до якої за допомогою сполучного кабелю підключаються трансівери устаткування;

– кроссировка: два сегменти постійного кабелю (комутаційні панелі на кінцях кожного), між якими здійснюється кроссировка на центральному вузлі, і сполучні кабелі для підключення трансіверів устаткування (див. рис. 3.1).

У структурованих кабельних системах використовуються оптичні розподільні системи (комутаційні поля високої щільності), за допомогою яких з'єднується постійна інфраструктура (магістралі MPO), що зв'язує різні області розміщення устаткування (Equipment Distribution Area, EDA) у ЦОД. Застосування претермінованих систем MPO (на відміну від монтуємих на місці) при використанні кросів дозволяє швидко створити таку інфраструктуру, де порти пристрою можна підключати до будь-якого іншого порту.

В умовах збільшення щільності мережевого устаткування порти SAN починають обчислюватися тисячами, і керування підключеними кабелями перетворюється в складне завдання. Історично підключення кабелів безпосередньо до окремих портів устаткування вважалося прийнятним лише в системах невеликого масштабу (при з'єднанні точка-точка або міжз’єднанні). Застосування того ж підходу до устаткування з більшою кількістю портів різко підвищує ймовірність помилок при виконанні операцій MAC.

Fibre Channel може підтримувати різні кабельні конфігурації, але для встановлюваних трансіверів потрібно знати бюджет потужності, особливо якщо канали відрізняються від базових каналів стандарту Fibre Channel-PI (дві пари сполучених оптичних рознімань із загасанням 0,75 дб на кожну).

У великих ЦОД використовують канали із чотирма касетними модулями й магістральними кабелями MPO, які зв'язують різні області розміщення устаткування з MDA.

На рис. 3.1 показаний канал із чотирма касетними модулями, у якому використовуються магістральні кабелі, прокладені з кожної сторони кросу/MDA. Кожний магістральний кабель підключається до модульної оптичної касети, створюючи канал, що поєднує чотири такі касети (загальна величина внесених втрат складається із втрат, що виникають у цих з'єднаннях).

Комутаційний шнур LC-LC в MDA з'єднує відображувані порти із внутрішньої сторони суміжних магістральних кабелів. Шнури LC-LC з кожної сторони лінії підключаються до трансіверів в устаткуванні. У такому сценарії використовуються два магістральних кабелі, два сполучних шнури LC-LC в EDA, один комутаційний шнур MDA LC-LC і чотири касетних модулі LC. В альтернативній конфігурації один касетний модуль LC заміняється (звичайно це робиться з боку SAN Director високої щільності) розгалужувачем MPO-LC і комутаційною панеллю MPO. У такій конфігурації мінімізовані втрати IL у з'єднаннях, оскільки одна пара сполучених рознімань LC-LC виключається з каналу (відсутня комутація LC в EDA SAN Director).

Канал із чотирма касетними модулями починається на порту SFP+ пристрою зберігання (у нижньому лівому куті рис. 3.1). Сполучний шнур LC-LC зв'язує пристрій зберігання з першою касетою, що, у свою чергу, підключається до магістралі MPO. Цю магістраль підключають до лівої касети в MDA, комутаційний шнур LC-LC зв'язує її із протилежною касетою, а в остаточному підсумку й з магістраллю в правій частині. Нарешті, остання приєднується до касети в області комутації, а правий сполучний шнур LC-LC – до порту SFP+ на комутаторі.

Це приклад типового розгортання структурованої кабельної інфраструктури із кросом MDA.

Вибір з'єднань Fibre Channel

У попередньому прикладі був показаний спосіб побудови каналу за допомогою касет LC-MPO і магістральних кабелів MPO, але в загальному випадку для реалізації структурованої кабельної моделі можна використовувати багато різних компонентів. Магістральні кабелі з розніманнями LC, які підключаються до задньої частини панелі, оснащеної адаптерами LC і розташовуваної в оптичній розподільній шафі, дозволяють обійтися без касетних модулів і тим самим зменшити загальні втрати IL у з'єднаннях (чотири сполучені пари рознімань MPO у цьому випадку виключаються). Проектні рішення Fibre Channel допускають використання різних кабельних компонентів і архітектур у залежності від конкретних потреб замовників.

Число з'єднань у каналі обмежено. У продемонстрованої нами моделі із чотирма касетними модулями втрати IL будуть високими, і для дотримання вимог стандарту Fibre Channel-PI їх необхідно зменшити. У базовій моделі стандарту Fibre Channel із двома з'єднаннями загальне загасання IL не повинне перевищувати 1,5 дб (дві сполучені пари рознімань по 0,75 дб у кожній), що досягається декількома шляхами.

Перший і найпростіший передбачає використання модулів з низькими втратами, завдяки чому сумарне загасання IL у з'єднаннях для конфігурації із чотирма касетами виявляється менше 1,5 дб (максимум по 0,375 дб на кожну). Другий спосіб вимагає знання канальної моделі й вибору компромісних варіантів. Втрати в з'єднаннях можуть і перевищувати 1,5 дб за рахунок зменшення загасання у волокні й зниження впливу ISI.

Дуже важливо вже на ранній стадії залучити до проектування фахівців зі структурованих кабельних систем, що дозволить краще зрозуміти, які обмеження властиві розроблювальної моделі з'єднань Fibre Channel.

Рішення Fibre Channel передбачають використання доступних за ціною трансіверів MMF і SFM для реалізації високошвидкісних портів SAN як у невеликих серверних, так і в гіпермасштабованих центрах обробки даних. У

більшості каналів Fibre Channel використовуються MMF із кросами й центральними комутаційними вузлами, розташованими в безпосередній близькості від більших комутаторів Fibre Channel директорського класу. У великих системах з комутаторами високої щільності кращим рішенням уважається відображення портів комутатора в кросах.

Для реалізації в кросах гнучкого зв'язку «кожного з кожним» у каналі використовуються кілька сполучних переходів. У представленому тут сценарії із чотирма касетами втрати IL у з'єднаннях нівелюються за рахунок вибору волокна необхідної довжини, що дозволяє одержати необхідний запас потужності.

При побудові протяжних каналів усередині будинків або між ними використовуються довгохвильові трансівери й одномодові волокна. У загальному випадку застосування рішень SMF (для варіантів довжиною 10 км) допомагає збільшити доступний бюджет потужності, що дає можливість установлювати кабельні системи з більше високими загальними втратами.

Перспективні плани розвитку Fibre Channel передбачають підтримку клієнтських застосунків SAN з подальшим ростом пропускної здатності в геометричній прогресії зі знаменником 2. Шосте покоління Fibre Channel забезпечує передачу даних на швидкості 128 Гбіт/с по чотирьох передавальним і чотирьох приймальнях лініям із пропускною здатністю 32 Гбіт/с кожна (з використанням паралельних оптичних кабелів MPO).

Незважаючи на відсутність явної вказівки в стандарті Fibre Channel-PI-6P, розгалужувальні рішення із пропускною здатністю 32 Гбіт/із уже підтримуються постачальниками комутаційного устаткування Fibre Channel. Недорогі модулі на базі SWDM (OM5 MMF), а також PSM4 і CWDM4 (SMF) будуть розроблені також для 128 GFC і перспективних технологій Fibre Channel із пропускною здатністю 256 Гбіт/с і вище.

3.3 Розробка функціональної схеми

Функціональна схема розробленої системи зображена на рисунку 3.2. Представимо опис розробленого програмного комплексу для дослідження системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P із неоднорідним потоком повідомлень на основі аналітичного й імітаційного моделювання.

Спочатку сформулюємо методику дослідження розподілених системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P із неоднорідним трафіком, що містить наступні етапи.

1. Підготовка вихідних даних для програмного комплексу:

– кількість вузлів мережі і їхнє взаємне розташування;

– навантаження системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P, створювана користувачами при роботі із прикладними програмами, наборами даних і в процесі обміну повідомлень;

– обмеження на середній час доставки пакетів або на вартість системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P;

– максимальна довжина пакета;

– вибір типу каналів і завдання вартісних коефіцієнтів КЗ.

2. Розрахунок оптимальних пропускних здатностей типових топологій системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P при заданих наступних параметрах:

– типова топологія: зірка, кільце, дерево, повнозв’язна;

– модель взаємодії користувачів мережі: RDA (Remote Data Access), DBS (DataBase Server) і AS (Application Server);

– розподіл прикладних програм і наборів даних по вузлах системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P;

– метод маршрутизації.

Після завдання цих параметрів розраховуються оптимальні значення пропускних здатностей КЗ, час передачі пакетів і завантаження каналів.

3. Розрахунок пропускних здатностей КЗ при розподілених топологіях.

4. Вибір найкращої топології системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P, для якої обрана залежно від постановки завдання як критерій ефективності характеристика системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P (середній час доставки пакетів або вартість системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P) приймає найменше значення.

5. Варіювання параметрів і характеристик КЗ і системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P і вибір дискретних значень пропускних здатностей.

6. Оцінка погрішностей аналітичних методів розрахунку характеристик КЗ.

7. Оцінка впливу характеру потоків пакетів і тривалості передачі даних на характеристики системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P:

– вплив третього моменту розподілу вхідного потоку пакетів на характеристики КЗ;

– вплив довжини пакетів на характеристики каналів зв'язку й системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P;

– вплив законів розподілів трафіку в мережах і часі передачі пакетів у КЗ на характеристики функціонування системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

8. Дослідження системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P при різних розміщеннях прикладних програм і наборів даних.

9. Детальний аналіз спроектованої системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P, що припускає варіювання параметрів системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P, у тому числі: довжини запитів і відповідей прикладних програм і наборів даних, імовірність передачі по основному шляху, способи маршрутизації й т.д.

Завдання проектування мережі складається у визначенні пропускних здатностей КЗ із урахуванням неоднорідності трафіку, різноманіття топологій, алгоритмів маршрутизації, варіантів розміщення прикладних програм і наборів даних по вузлах мережі, способів взаємодії користувачів мережі.

Аналітична модель системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P будується у вигляді розімкнутої СеМО для будь-якої топології мережі, що задається аналітично або графічно. На основі аналітичної моделі вирішується завдання визначення пропускних здатностей КЗ у розподілених системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P при обмеженнях на час доставки пакетів або на вартість мережі.

Програма дозволяє розрахувати характеристики системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P, такі як:

– пропускні здатності КЗ;

– час затримки пакетів при передачі по кожному каналу й у мережі в цілому;

– завантаження кожного КЗ;

– інтенсивності потоків пакетів у каналах зв'язку;

– імовірності передачі пакетів від користувачів у мережу, між каналами й з мережі до користувачів.

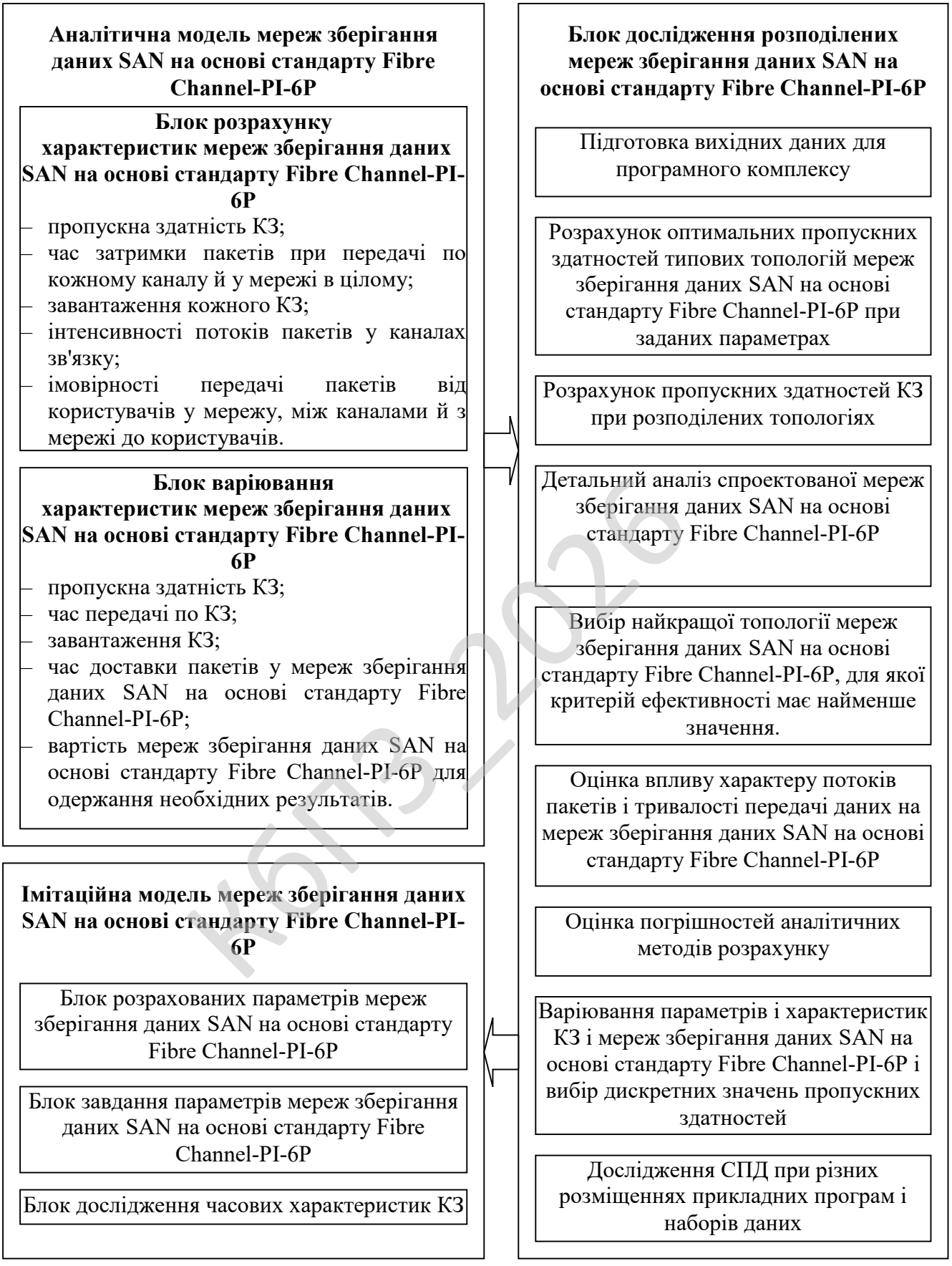

Рисунок 3.2 – Функціональна схема системи

Програма дозволяє варіювати отримані характеристики, такі як:

– пропускні здатності;

– час передачі по КЗ;

– завантаження КЗ;

– час доставки пакетів у мережі зберігання даних SAN на основі стандарту Fibre Channel-PI-6P;

– вартість системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P для одержання необхідних результатів.

Крім того, вона дозволяє вибрати значення пропускних здатностей з дискретного ряду значень, знайдених із традиційної оптимізації безперервних значень пропускних здатностей.

Імітаційна модель системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P генерується автоматично на основі розрахунку характеристик аналітичної моделі у вигляді розімкнутої експонентної СеМО. Отримані характеристики аналітичної моделі використовуються як параметри імітаційної моделі системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P, що представляє собою розімкнуту СеМО, вузли якої відповідають каналам зв'язку, а заявки – пакетам у системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P. Число джерел заявок в імітаційній моделі дорівнює числу вузлів у системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P, при цьому інтенсивності надходження заявок у моделі визначаються як зовнішні інтенсивності пакетів від користувачів до вузлів системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P відповідно. Аналітична модель, реалізована в програмі, дозволяє розрахувати наступні параметри для імітаційної моделі.

1. Імовірність передачі пакетів від користувача *j* до каналу *k*.

2. Імовірність передачі пакетів від каналу *k* до каналу *h*.

3. Імовірність передачі пакетів від каналу *k* до користувача *j*.

4. Середній інтервал часу між вступниками пакетами від користувача *j* у системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

5. Середній час передач пакетів у каналах зв'язку *k*.

Крім **розрахованих параметрів** для імітаційної моделі системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P додатково необхідно **задати наступні параметри**:

а) закони розподілів інтервалів часу між вступниками пакетами від користувачів у системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P із коефіцієнтами варіації цих розподілів;

б) закони розподілів часу передачі пакетів у КЗ із коефіцієнтами варіації цих розподілів.

Відзначимо, що імітаційна модель системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P призначена для детального аналізу характеристик функціонування мережі, спроектованої в процесі аналітичного моделювання. При цьому, у випадку відмінності реального характеру процесів надходження пакетів у мережу або передачі пакетів по каналах зв'язку від експонентного, в імітаційній моделі передбачена можливість варіювання законів розподілу часу передачі (довжин обслуговування) пакетів у кожному із КЗ, а також законів розподілу інтервалів часу між вступними в мережу пакетами. Як такі закони в роботі використовувалися наступні розподіли: експонентний, детермінований; гіпоекспоненційний різного порядку й, відповідно, з різними коефіцієнтами варіації; рівномірний; експонентний з ненульовими зсувами; гіперекспонентний; Гамма-розподіл.

Аналогічно **розроблений засіб дослідження часових характеристик каналу зв'язку** шляхом генерування імітаційної моделі каналу зв'язку у вигляді СМО типу G/G/1 з можливістю зміни завантаження каналу й варіювання законів розподілу інтервалів часу між пакетами й часу передачі пакетів по каналі. Даний

засіб дозволяє одержати значення часових характеристик каналу зв'язку й оцінити погрішність аналітичних методів розрахунку характеристик каналу зв'язку. Розглянувши усі блоки функціональної схеми перейдемо до розгляду діаграми взаємодії процесів, які відбуваються у системі.

3.4 Розробка діаграми процесів

Розглянемо розроблену діаграму процесів яка зображена на рисунку 3.3. Основна будова діаграми процесів полягає у графічному представленні складу сукупностей даних, що характеризуються як співвідношення різних частин кожної з сукупностей. Склад статистичної сукупності графічно може бути представлений як за допомогою абсолютних, так і відносних показників. Графічне зображення складу сукупності по абсолютними і відносними показниками сприяє проведенню більш глибокого аналізу і дозволяє проводити аналіз системи.

Діаграма взаємодії процесів використовується для візуалізації процесів обробки даних (структурне проектування).

Для розробника вважається звичним спочатку креслити діаграму взаємодії процесів даних рівня контексту, завдяки чому буде показано взаємодію системи.

Ця діаграма в подальшому підлягає уточненню шляхом деталізації процесів та потоків даних з метою показати систему що розробляється.

При детальному її розгляді можна побачити як саме проходе взаємодія у розробленій системі. Використовується модель проектування, графічне представлення «потоків» даних в інформаційній системі.

Діаграми потоків даних містять чотири типи елементів:

– Зовнішні по відношенню до системи сутності.

– Потоки даних між елементами трьох попередніх типів.

– Процеси які являють собою трансформацію даних в рамках описуваної системи.

– Сховища даних (репозиторії).

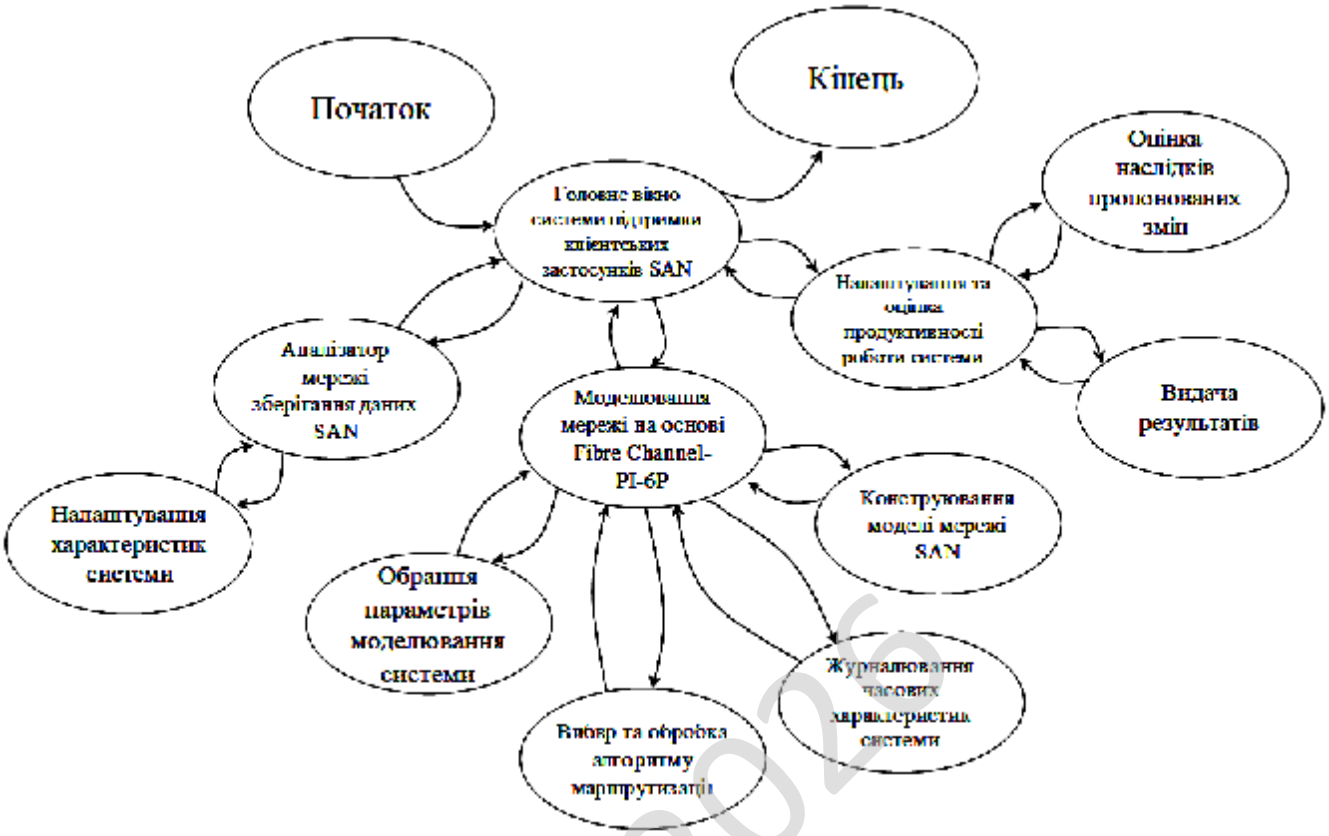

Рисунок 3.3 – Діаграма взаємодії процесів

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 ПРОЕКТУ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ПРАВИЛЬНІСТЬ ПРОЕКТНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Розглянемо реалізацію бакалаврської дипломної роботи. Були проведені розрахунки і підібрані набори тестових даних для перевірки правильності реалізації проектних рішень.

Було створено блок-схеми роботи системи. Перед їх розглядом необхідно провести роз’яснення який саме тип блок-схем використовується. Блок-схема це представлення задачі для її аналізу або розв'язування за допомогою спеціальних символів (геометричних образів), які позначають такі елементи, як операції, потік, дані тощо.

Блок вхідних та вихідних даних прийнято позначати паралелограмом, блок обчислень (обробки) даних – прямокутником, блок прийняття рішень – ромбом, еліпсом – початок та кінець алгоритму.

У інформаційних технологіях функціональна схема складається з функціональних блоків, які являють собою конструктивно відособлені частини (елементи або пристрої) автоматичних систем, які виконують певні функції. Функціональні блоки на схемі позначають прямокутниками, всередині яких надписують їх найменування відповідно до функцій, що виконуються. Зв'язки між функціональними блоками (внутрішні впливи) позначаються лініями зі стрілками, які вказують напрям впливів.

Функціональні схеми можуть виконуватися в укрупненому і розгорненому вигляді. У першому випадку на схемі зображають найважливіші блоки системи і зв'язки між ними.

У другому варіанті схема відображається більш детально, що полегшує її читання та ілюструє принцип роботи.

Основні елементи схем алгоритму це термінатор, процес, рішення, зумовлений процес (підпрограма), дані та з'єднувач. Термінатор це елемент відображає вхід із зовнішнього середовища або вихід з неї (найчастіше застосування – початок і кінець програми). Всередині фігури записується відповідна дія.

Процес це виконання однієї або кількох операцій, обробка даних будь-якого виду (зміна значення даних, форми подання, розташування). Всередині фігури записують безпосередньо самі операції. Рішення це показує рішення або функцію перемикального типу з одним входом і двома або більше альтернативними виходами, з яких тільки один може бути обраний після обчислення умов, визначених всередині цього елементу.

Вхід в елемент позначається лінією, що входить зазвичай у верхню вершину елементу. Якщо виходів два чи три то зазвичай кожен вихід позначається лінією, що виходить з решти вершин (бічних і нижній). Якщо виходів більше трьох, то їх слід показувати однією лінією, що виходить з вершини (частіше нижній) елемента, яка потім розгалужується.

Відповідні результати обчислень можуть записуватися поруч з лініями, що відображають ці шляхи. Зумовлений процес (підпрограма) це символ відображає виконання процесу, що складається з однієї або кількох операцій, що визначені в іншому місці програми (у підпрограмі, модулі). Всередині символу записується назва процесу і передані в нього дані. Дані це перетворення у форму, придатну для обробки (введення) або відображення результатів обробки (виведення). Цей символ не визначає носія даних (для вказівки типу носія даних використовуються специфічні символи). З'єднувач це символ відображає вихід в частину схеми і вхід з іншої частини цієї схеми.

Використовується для обриву лінії та продовження її в іншому місці (приклад: поділ блок-схеми, що не поміщається на листі). Відповідні сполучні символи повинні мати одне (при тому унікальне) позначення.

Блок-схеми показують весь процес роботи системи з підсистемами та частково доказують правильність вибраних проектних рішень. Тому від точності і детальної блок-схеми залежить результат всієї програми.

При виборі початкової точки відліку при побудові схем було враховано, що виходячи з вибору мови програмування і інших технічних засобів, програма буде об’єктно-орієнтована що вимагає високого рівня декомпозиції задач на класи.

На рисунку 4.1 зображена основна блок-схема програми, на рисунку 4.2 зображено роботу підпрограми.

З яких видно що робота основної програми складається з початкових етапів ініціалізації ПЗ, перевірки наявності ресурсів системи, блоку початку основного циклу з чеканням запиту від користувача в якому відбувається виклик підсистеми та останньої стадії – перевірка поточного стану з завершенням роботи розробленого ПЗ. При роботі підпрограми виконується основний функціонал системи з циклічними послідовностями, перевіркою поточного стану та поверненням в основну програму прапорів стану виконання.

NetBIOS (Network Basic Input/Output System) – протокол для роботи в локальних мережах на персональних ЕОМ типу IBM/PC, розроблений у вигляді інтерфейсу, який не залежить від фірми–виробника. Був розроблений фірмою Sytek Corporation за замовленням IBM в 1983 році. Він включає в себе інтерфейс сеансового рівня (англ. NetBIOS interface), в якості транспортних протоколів використовує TCP і UDP.

Особливістю NetBIOS є можливість його роботи поверх різних протоколів, найпоширенішими/відомими з яких є NetBEUI, IPX і стек протоколів TCP/IP; причому якщо старі версії Windows орієнтувалися на більш легкі в реалізації і менш ресурсомісткі NetBEUI і IPX, то сучасні Windows орієнтуються на TCP/IP.

При використанні NetBEUI і IPX NetBIOS сам забезпечує надійність доставки даних (функціональність SPX не використовувати), а при використанні TCP/IP надійність доставки забезпечує TCP, за що удостоївся окремого імені «NBT».

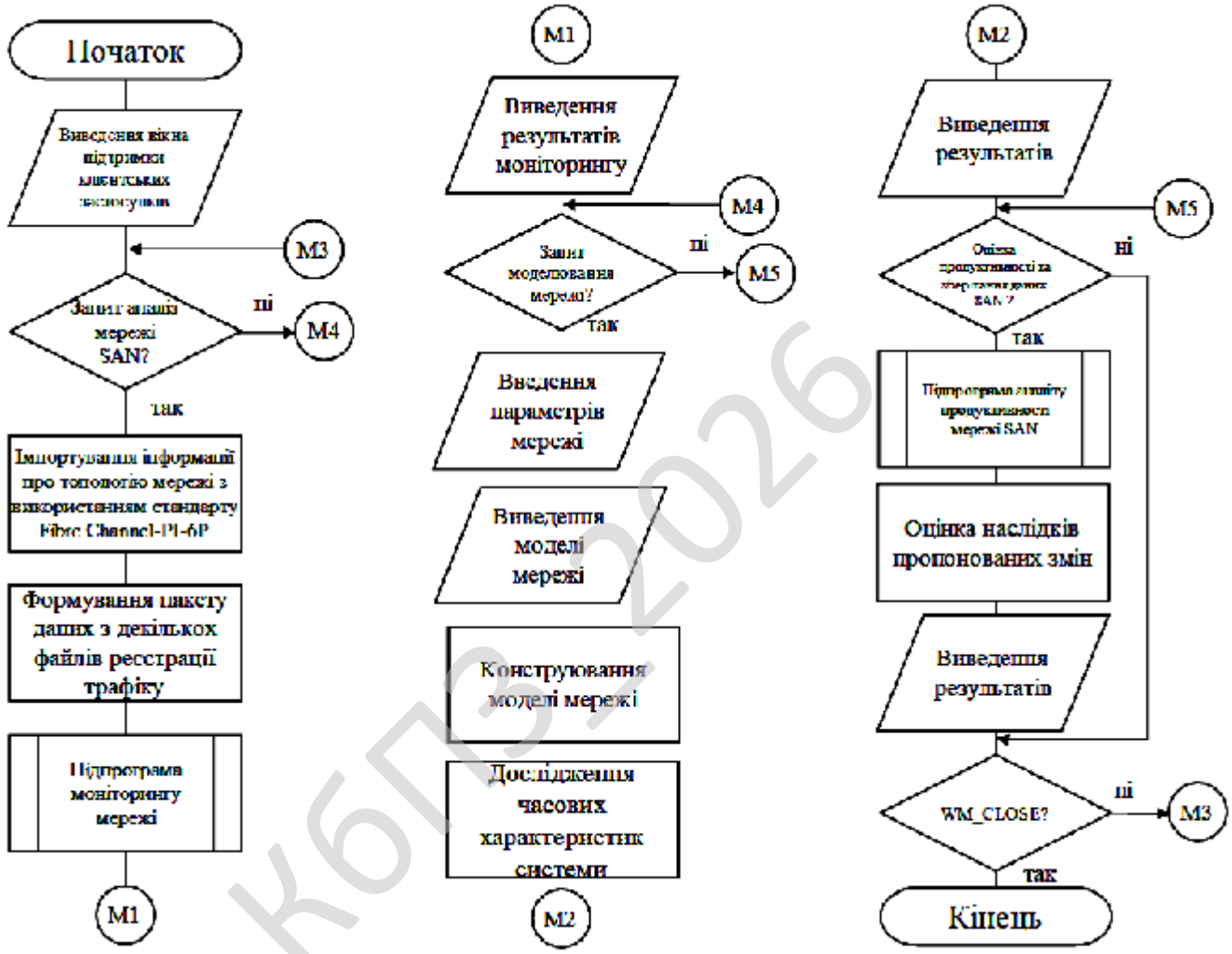

Рисунок 4.1 – Блок-схема основної програми

Інтерфейс NetBIOS являє собою типовий інтерфейс взаємодії програм (API) для забезпечення мережевих операцій введення–виведення і управління транспортним протоколом.

Програми, що використовують NetBIOS API інтерфейс, можуть працювати тільки при наявності протоколу, що допускає використання такого інтерфейсу.

NetBIOS також визначає протокол, що функціонує на сеансовому/транспортному рівнях моделі OSI. Цей протокол використовується протоколами нижчих рівнів, такими як NBFP (NetBEUI) і NetBT для виконання мережевих запитів вводу–виводу і операцій, описаних в стандартному интерфейсном наборі команд NetBIOS.

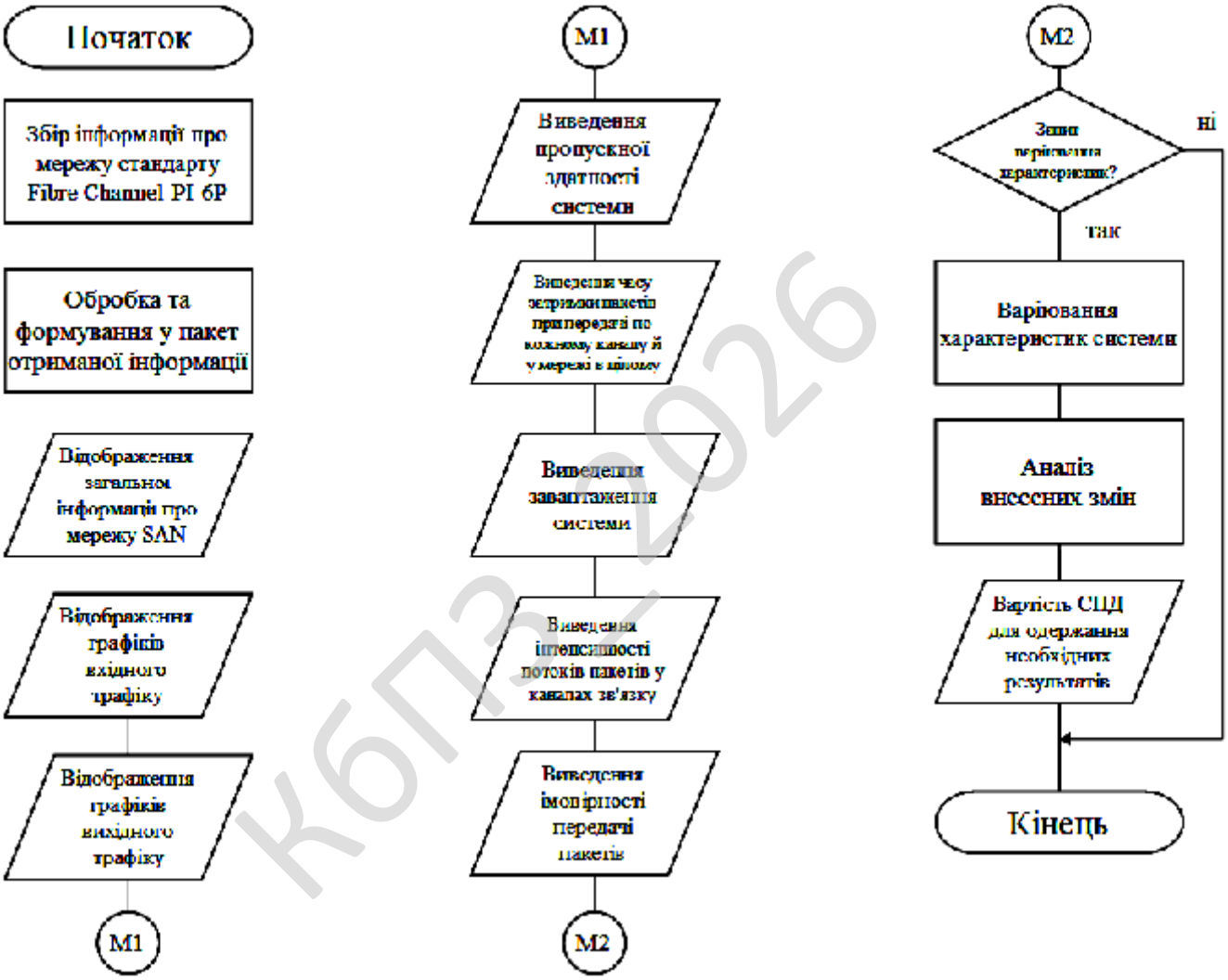

Рисунок 4.2 – Блок-схема роботи підпрограми

Тобто, NetBIOS сам не підтримує виконання файлових операцій. Ця функція покладається на протоколи нижчих рівнів, а сам NetBIOS забезпечує тільки зв'язок з цими протоколами і NetBIOS API–інтерфейс.

Пакет NETBIOS створений для використання групою ЕОМ. Підтримує як режим сесій (робота через з'єднання), так і режим дейтаграм (без встановлення з'єднання). 16–символьні імена об'єктів в NETBIOS розподіляються динамічно.

NETBIOS має власну DNS, яка може взаємодіяти з інтернетівський. Ім'я об'єкта при роботі з NETBIOS не може починатися з символу *.

Додатки можуть знайти через NETBIOS потрібні їм ресурси, встановити зв'язок і послати або отримати інформацію. NETBIOS використовує для служби імен порт 137, для служби дейтаграм – порт 138, а для сесій – порт 139. Будь–яка сесія починається з NETBIOS–запиту, завдання IP–адреси і визначення TCP–порту віддаленого об'єкта, далі йде обмін NETBIOS–повідомленнями, після чого сесія закривається.

Сесія здійснює обмін інформацією між двома NETBIOS – додатками. Довжина повідомлення лежить в межах від 0 до 131 071 байт. Припустимо одночасне встановлення декількох сесій між двома об'єктами.

При організації IP–транспорту через NETBIOS IP–дейтаграмма вкладається в NETBIOS–пакет. Інформаційний обмін відбувається в цьому випадку без встановлення зв'язку між об'єктами. Імена NETBIOS повинні містити в собі IP–адреси. Так, частина NETBIOS–адреси може мати вигляд IP.XX.XX. XX. XX, де IP вказує на тип операції (IP через Netbios), а XX. XX. XX.XX – IP– адреса.

Система NETBIOS має власну систему команд (call, listen, hang up, send, receive, session status, reset, cancel, adapter status, unlink, remote program load) і примітивів для роботи з дейтаграммами (send datagram, send broadcast datagram, receive datagram , receive broadcast datagram).

Всі кінцеві вузли NETBIOS діляться на три типи:

– широкомовні («b») вузли;

– вузли точка–точка («p»);

– вузли змішаного типу («m»).

IP–адреса може асоціюватися з одним із зазначених типів. B–вузли встановлюють зв'язок зі своїм партнером за допомогою широкомовних запитів. P–

і M–вузли для цієї мети використовують netbios сервер імен (NBNS) і сервер розподілу дейтаграм (NBDD).

NetBIOS забезпечує:

– реєстрацію і перевірку мережевих імен;

– встановлення і розрив з'єднань;

– зв'язок з підтвердженням доставки інформації;

– зв'язок без підтвердження доставки інформації;

– підтримку управління і моніторингу драйвера і мережевої карти. Розглянемо визначення API. Це прикладни́й програ́мний інтерфе́йс (Application Programming Interface, API) – набір визначень підпрограм, протоколів взаємодії та засобів для створення програмного забезпечення.

Спрощено – це набір чітко визначених методів для взаємодії різних компонентів. API надає розробнику засоби для швидкої розробки програмного забезпечення. API може бути для веб-базованих систем, операційних систем, баз даних, апаратного забезпечення, програмних бібліотек.

Одним з найпоширеніших призначень API є надання набору широко використовуваних функцій, наприклад для малювання вікна чи іконок на екрані. API є абстрактним поняттям – програмне забезпечення, що пропонує деякий API, часто називають реалізацією (implementation) даного API. У багатьох випадках API є частиною набору розробки програмного забезпечення, водночас, набір розробки може включати як API, так і інші інструменти/апаратне забезпечення, отже ці два терміни не є взаємозамінювані.

Високорівневі API часто програють у гнучкості. Виконання деяких функцій нижчого рівня стає набагато складнішим, або навіть неможливим.

В об'єктно-орієнтованих мовах, прикладний програмний інтерфейс зазвичай включає в себе опис набору визначень класу, з набором форм поведінки, пов'язаних з цими класами. Це абстрактне поняття пов'язане з реальними функціями, які надані або надаватимуться, класами, які реалізуються в методах класу.

Прикладний програмний інтерфейс в даному випадку можна розглядати як сукупність всіх методів, які публічно доступні в класах (зазвичай званий інтерфейс класу). Це означає, що прикладний програмний інтерфейс вказує методи, за допомогою яких взаємодіє з об'єктами, отриманими з визначень класів і обробляє їх.

У більш загальному плані можна визначити Прикладний Програмний Інтерфейс як сукупність усіх видів об'єктів, які можна вивести з визначення класу, і пов'язаних з ними можливих варіантів поведінки.

Наприклад: клас, що представляє Stack, може просто виставити публічно два методи Push() (для додавання нового елемента в стек) і Pop() (для віддалення останнього пункту, ідеально розташований на вершині стека).

У цьому випадку Прикладний Програмний Інтерфейс може бути інтерпретованим як два методи pop() і push(), або, більш широко, використовується варіант, коли можна використовувати елемент типу Stack, який реалізує поведінку стека, надаючи йому можливість для додавання / видалення елементів з вершини. Друга інтерпретація видається більш доречною в дусі об'єктно-орієнтованого підходу.

Якість документації, пов'язаної з Прикладним Програмним Інтерфейсом, є часто ключовим фактором, що визначає його успішність з точки зору простоти використання.

ППІ, як правило, пов'язаний із бібліотеками програмного забезпечення: ППІ описує і вказує очікувану поведінку в той час, як бібліотека є фактичною реалізацією даного набору правил. Один ППІ може мати декілька реалізацій (або жодної, будучи абстрактним) у вигляді різних бібліотек, які мають такий же інтерфейс.

Прикладний програмний інтерфейс також може бути пов'язаним з платформами програмування: платформа може бути заснована на кількох бібліотеках реалізує декілька інтерфейсів ППІ, але на відміну від звичайного використання ППІ, доступ до поведінки вбудований в платформу

опосередкований шляхом розширення його змісту новими класами і вставлений в саму платформу. Крім того, загальний потік управління програми може бути під контролем абонента.

Прикладний програмний інтерфейс може бути також реалізацією протоколу. Коли ППІ реалізує протокол, він може бути заснованим на проксі-методах віддалених викликів, що засновані на протоколі зв'язку. Роль ППІ може заключатися саме в тому, щоб приховати деталі транспортного протоколу. Наприклад: RMI є ППІ, який реалізує протокол або JRMP IIOP як RMI-IIOP.

Протоколи, як правило, розподіляються між різними технологіями і зазвичай дозволяють різним технологіям обмінюватися інформацією, діючи як абстракція між двома світами. ППІ, як правило, є специфічним для конкретної технології: звідси, інтерфейси даної мови не можуть бути використані на інших мовах, якщо виклики функції не будуть перетворені з конкретної адаптації бібліотеки.

Деякі мови, серед яких такі, що працюють на віртуальних машинах (наприклад: мови, сумісні з NET CLI середовища CLR і JVM сумісних мов у віртуальній машині Java) можуть ділитися програмними інтерфейсами.

У цьому випадку віртуальна машина дозволяє мові взаємодії завдяки спільному знаменнику віртуальної машини, що абстрагується від конкретної мови, використовувати проміжний байт-код і його мову.

При використанні прикладного програмного інтерфейсу в контексті веб-розробки, як правило, ППІ визначається набором повідомлень запиту HTTP, також визначається структура повідомлень-відповідей, зазвичай у розширенні мови розмітки XML або в форматі об'єктного запису JavaScript (JSON). У той час як прикладний програмний інтерфейс у Web історично був практично синонімом для веб-служби, останнім часом тенденція змінилась (так званий Web 2.0) на відхід від Simple Object Access Protocol (SOAP) на основі веб-сервісів і сервіс-орієнтованої архітектури (SOA) на більш прямі передачі репрезентативного стану (REST) стилів веб-ресурсів та ресурсів-орієнтованої архітектури (ROA).

Частина цієї тенденції пов'язана з рухом Семантичного веб-ресурсу до Опису Платформ (RDF), Концепції розвитку веб-технологій інженерних онтологій. Прикладні програмні інтерфейси у Web, що дозволяють комбінувати декількома прикладними програмними інтерфейсами в нові додатки називають гібридними.

Розглянемо протокол SNMP (Simple Network Management Protocol, простий протокол керування мережею) – це протокол керування мережами зв'язку на основі архітектури TCP/IP.

На основі концепції TMN в 1980–1990 р. різними органами стандартизації був вироблений ряд протоколів керування мережами передачі даних з різним спектром реалізації функцій TMN. До одного з типів таких протоколів керування належить Simple Network Management Protocol (SNMP).

SNMP – це технологія, покликана забезпечити керування й контроль за пристроями й програмами в мережі зв'язку шляхом обміну керуючою інформацією між агентами, що розташовуються на мережних пристроях, і менеджерами, розташованими на станціях керування. SNMP визначає мережу як сукупність мережних керуючих станцій й елементів мережі (головні машини, шлюзи й маршрутизатори, термінальні сервери), які спільно забезпечують адміністративні зв'язки між мережними керуючими станціями й мережними агентами. SNMP різних версій присвячений цілий ряд рекомендацій IETF (RFC).

Зазвичай при використанні SNMP присутні керовані та керуючі системи. До складу керованої системи входить компонент, який називається агентом, який відправляє звіти керуючій системі. По суті SNMP агенти передають управлінську інформацію на керуючі системи як змінні (такі як «вільна пам'ять», «ім'я системи», «кількість працюючих процесів» тощо).

Керуюча система може отримати достовірну інформацію через операції протоколу GET, GETNEXT і GETBULK. Агент може самостійно без запиту надсилати дані, використовуючи операцію протоколу TRAP або INFORM. Управляючі системи можуть також відправляти конфігураційні оновлення або

контролюючі запити, використовуючи операцію SET для безпосереднього управління системою. Операції конфігурування та управління використовуються тільки тоді, коли потрібні зміни у мережній інфраструктурі. Операції моніторингу зазвичай виконуються на регулярній основі.

Змінні, доступні через SNMP, організовані в ієрархії. Ці ієрархії та інші метадані (такі як тип і опис змінної) описуються Базами Керуючої Інформації' (Management Information Bases (MIBs)).

SNMP не визначає, яку інформацію (які змінні) керована система повинна надавати. Навпаки, SNMP використовує розширювану модель, в якій доступна інформація визначається Базами Керуючої Інформації (MIB – Management Information Base).

Бази Керуючої Інформації описують структуру керуючої інформації пристроїв. Вони використовують ієрархічний адресний простір імен, що містить унікальний ідентифікатор об'єкта (object identifier (OID)).

Грубо кажучи, кожен унікальний ідентифікатор об'єкта ідентифікує змінну, яка може бути прочитана чи встановлена через SNMP. MIB'и використовують нотацію, визначену в ASN.1.

Ієрархія MIB може бути зображена як дерево з безіменним коренем, рівні якого приписані різними організаціями. На найвищому рівні MIB OID'и належать різним організаціям, що займаються стандартизацією, в той час як на нижчих рівнях OID'и виділяються асоційованими організаціями. Ця модель забезпечує управління на всіх шарах мережної моделі OSI, адже MIB'и можуть бути визначені для будь-яких типів даних і операцій.

Керований об'єкт – це одна з будь-якого числа характеристик, специфічних для керованого пристрою. Керований об'єкт включає в себе один або більше екземплярів об'єкта (що ідентіфікуються за OID), які насправді є змінними.

Існує два типи керованих об'єктів: Скалярні об'єкти визначають єдиний екземпляр об'єкта; Табличні об'єкти визначають множинні, пов'язані екземпляри об'єктів які групуються в таблицях MIB.

Прикладом керованого об'єкта може бути atInput, який є скалярним об'єктом що містить єдиний екземпляр об'єкта, ціле число, яке показує загальну кількість вхідних пакетів AppleTalk на мережний інтерфейс маршрутизатора.

Ідентифікатор об'єкта (OID) унікально ідентифікує керований об'єкт в ієрархії MIB.

В телекомунікаціях і комп'ютерних мережах, ASN.1 є стандартною гнучкою нотацією для опису структур даних, що служать для кодування, передачі і декодування даних. ASN.1 являє собою набір правил для опису структури об'єктів, незалежних від специфічних для обладнання методик кодування, і формальну нотацію, яка дозволяє уникнути неоднозначностей.

ASN.1 це єдиний ISO та ITU-T стандарт, спочатку визначений у 1984 році як частина стандарту CCITT X.409: 1984.

Пізніше, в 1988 році, завдяки його широкому застосуванню, він був виділений в окремий стандарт X.208. Значно переглянута версія 1995року описана в X.680.

Адаптована підмножина ASN.1 Структура Керуючої Інформації (SMI) описана в протоколі SNMP для визначення наборів пов'язаних MIB об'єктів, також званих MIB модулями.

SNMP працює на прикладному рівні TCP/IP (сьомий рівень моделі OSI). Агент SNMP отримує запити по UDP-порту 161. Менеджер може посилати запити з будь-якого доступного порту джерела на порт агента. Відповідь агента буде відправлений назад на порт джерела на менеджері.

Менеджер отримує повідомлення (Traps і InformRequests) по порту 162. Агент може генерувати повідомлення з будь-якого доступного порту. При використанні TLS або DTLS запити виходять по порту 10161, а пастки відправляються на порт 10162.

У SNMPv1 зазначено п'ять основних протокольних одиниць обміну (protocol data units – PDU). Ще дві PDU, GetBulkRequest і InformRequest, були введені в SNMPv2 і перенесені в SNMPv3.

Нижче перераховані сім протокольних одиниць обміну SNMP:

1. GetRequest. Запит від менеджера до об'єкту для отримання значення змінної або списку змінних. Необхідні змінні вказуються в полі variable bindings (розділ поля values при цьому не використовується). Отримання значень зазначеної змінної повинно бути виконано агентом як Атомарна операція. Менеджеру буде повернений Response (відповідь) з поточними значеннями.

2. SetRequest. Запит від менеджера до об'єкту для зміни змінної або списку змінних. Зв'язані змінні вказуються в тілі запиту. Зміни всіх зазначених змінних повинні бути виконані агентом як атомарна операція. Менеджеру буде повернений Response з (поточними) новими значеннями змінних.

3. GetNextRequest. Запит від менеджера до об'єкту для виявлення доступних змінних і їх значень. Менеджеру буде повернений Response зі зв'язаними змінними для змінної, яка є наступною в базі MIB в лексіграфіческом порядку. Обхід всієї бази MIB агента може бути проведений ітераційним використанням GetNextRequest, починаючи з OID 0. Рядки таблиці можуть бути прочитані, якщо вказати в запиті OID-и колонок в пов'язаних змінних.

4. GetBulkRequest. Покращена версія GetNextRequest. Запит від менеджера до об'єкту для численних ітерацій GetNextRequest. Менеджеру буде повернений Response з декількома пов'язаними змінними, обійденими починаючи зі пов'язаної змінної (змінних) в запиті. Специфічні для PDU поля non-repeaters і max-repetitions використовуються для контролю за поведінкою відповіді. GetBulkRequest був введений в SNMPv2.

5. Response. Повертає зв'язані змінні і значення від агента менеджеру для GetRequest, SetRequest, GetNextRequest, GetBulkRequest і InformRequest. Повідомлення про помилки забезпечуються полями статусу помилки і індексу помилки.

Ця одиниця використовується як відповідь і на Get-, і на Set-запити, в SNMPv1 називається GetResponse.

6. Trap. Асинхронне повідомлення від агента – менеджеру. Включає в себе поточне значення sysUpTime, OID, що визначає тип trap (пастки), і необов'язкові зв'язані змінні. Адресація одержувача для пасток визначається за допомогою змінних trap-конфігурації в базі MIB. Формат trap-повідомлення був змінений в SNMPv2 і PDU перейменували в SNMPv2-Trap.

7. InformRequest. Асинхронне повідомлення від менеджера менеджеру або від агента менеджеру. Повідомлення від менеджера менеджеру були можливі вже в SNMPv1 (за допомогою Trap), але SNMP зазвичай працює на протоколі UDP, в якому доставка повідомлень не гарантована, і не повідомляється про втрачені пакетах. InformRequest виправляє це зворотним відправленням підтвердження про отримання. Одержувач відповідає Response-му, що повторює всю інформацію з InformRequest. Цей PDU був введений в SNMPv2.

Незважаючи на те що я працював над ПЗ один в реалізації програми я використовував підходи пришвидшення розробки на основі методологій Agile.

Гнучка́ розробка програмного забезпечення (Agile software development, agile-методи) – клас методологій розробки програмного забезпечення, що базується на ітеративній розробці, в якій вимоги та розв'язки еволюціонують через співпрацю між самоорганізовуваними багатофункціональними командами.

Гнучка розробка – найкращий засіб для підвищення продуктивності розробників програмного забезпечення.

Більшість гнучких методологій націлені на мінімізацію ризиків, шляхом зведення розробки до серії коротких циклів, що мають назву ітерацій, які зазвичай тривають один-два тижні. Кожна ітерація сама по собі виглядає як програмний проект в мініатюрі, і включає всі завдання, необхідні для видачі мінімального приросту за функціональністю: планування, аналіз вимог, проектування, кодування, тестування і документування. Хоча окрема ітерація, як правило, недостатня для випуску нової версії продукту, мається на увазі те, що гнучкий програмний проект готовий до випуску наприкінці кожної ітерації. Після закінчення кожної ітерації, команда виконує переоцінку пріоритетів розробки.

Agile акцентує увагу на безпосередньому спілкуванні «віч-на-віч». Більшість agile команд розташовані в одному офісі, його іноді називають bullpen. Як мінімум вона включає і «замовників» (замовники, які визначають продукт, також це можуть бути менеджери продукту, бізнес аналітики або клієнти). Офіс може також включати тестувальників, дизайнерів інтерфейсу, технічних авторів і менеджерів.

Основною метрикою agile методів є робочий продукт. Віддаючи перевагу безпосередньому спілкуванню, agile-методи зменшують обсяг письмової документації в порівнянні з іншими методами. Це привело до критики цих методів як недисциплінованих.

Agile – родина процесів розробки, а не єдиний підхід в розробці програмного забезпечення, і визначається Agile Manifesto. Agile не включає практик, а визначає цінності та принципи, якими керуються успішні команди.

Agile Manifesto розроблений і прийнятий 17 розробниками 11-13 лютого 2001 року на лижному курорті The Lodge at Snowbird в горах Юти. Маніфест підписали представники наступних методологій Extreme programming, Scrum, DSDM, Adaptive software development, Crystal Clear, Feature driven development, Pragmatic Programming. Agile Manifesto містить 4 основні ідеї та 12 принципів. Примітно, що Agile Manifesto не містить практичних порад.

Основні ідеї:

– Особистості та їхні взаємодії важливіші, ніж процеси та інструменти;

– Робоче програмне забезпечення важливіше, ніж повна документація;

– Співпраця із замовником важливіша, ніж контрактні зобов'язання;

– Реакція на зміни важливіша, ніж дотримання плану.

Принципи, які роз'яснює Agile Manifesto:

– задоволення клієнта за рахунок ранньої та безперебійної поставки коштовного програмного забезпечення;

– вітання змін вимог навіть наприкінці розробки (це може підвищити конкурентоспроможність отриманого продукту);

– часта поставка робочого програмного забезпечення (кожен місяць або тиждень або ще частіше);

– тісне, щоденне спілкування замовника з розробниками впродовж всього проекту;

– проектом займаються мотивовані особистості, які забезпечені потрібними умовами роботи, підтримкою і довірою;

– рекомендований метод передачі інформації – особиста розмова (віч-на-віч);

– робоче програмне забезпечення – найкращий вимірювач прогресу;

– спонсори, розробники та користувачі повинні мати можливість підтримувати постійний темп на невизначений термін;

– постійну увагу поліпшенню технічної майстерності та зручному дизайну;

– простота – мистецтво не робити зайвої роботи;

– найкращі технічні вимоги, дизайн та архітектура виходять у самоорганізованої команди;

– постійна адаптація до мінливих обставин.

Маніфест та Принципи гнучкої розробки містять високорівневі ідеї щодо того, як потрібно вибудовувати процес розробки програмного забезпечення, щоб успішно завершувати проекти й створювати команди, в яких приємно та цікаво працювати.

Документи визначають, що потрібно для цього зробити, але не говорять, як це зробити. По-іншому й не могло бути, оскільки Маніфест та Принципи народилися внаслідок консенсусу представників різних (хоча й споріднених) напрямів, які могли знайти спільну основу лише на рівні базових цінностей та принципів.

Критика. Багато керівників проектів, що працюють у традиційних методологіях на кшталт «водоспаду», критикують agile-методи.

Один з повторюваних пунктів критики: при agile-підході часто нехтують створенням «дорожньої карти» розвитку продукту, так само як і управлінням

вимогами, в процесі якого і формується така «карта». Гнучкий підхід до управління вимогами не має на увазі далекосяжних планів (по суті, управління вимогами просто не існує в даній методології), а має на увазі можливість замовника раптом і несподівано наприкінці кожної ітерації виставляти нові вимоги, що часто суперечать архітектурі вже створеного і поставленого продукту. Таке іноді призводить до катастрофічних «авралів» з масовим рефакторингом і переробками практично на кожній черговій ітерації.

Крім того вважається, що робота в agile мотивує розробників вирішувати всі прибулі завдання найпростішим і найшвидшим можливим способом, при цьому часто не звертаючи уваги на коректність коду з точки зору вимог базової платформи (підхід «працює, та й добре», при цьому не враховується, що може перестати працювати при найменшій зміні або ж породити важкі до відтворення дефекти після реального розгортання у клієнта). Це призводить до зниження якості продукту і накопиченню дефектів.

Методології. Існують методології, які дотримуються цінностей і принципів заявлених в Agile Manifesto, деякі з них:

1. Agile Modeling – набір понять, принципів і прийомів (практик), що дозволяють швидко і просто виконувати моделювання і документування в проектах розробки програмного забезпечення. Не включає в себе детальну інструкцію з проектування, не містить описів, як будувати діаграми на UML.

Основна мета – ефективне моделювання і документування; але не охоплює програмування та тестування, не включає питання управління проектом, розгортання і супроводу системи. Однак включає в себе перевірку моделі кодом.

2. Agile Unified Process (AUP) спрощена версія IBM Rational Unified Process (RUP), розроблена Скоттом Амблером, яка описує просте і зрозуміле наближення (модель) для створення програмного забезпечення для бізнес-додатків.

3 Agile Data Method – група ітеративних методів розробки програмного забезпечення, в яких вимоги та рішення досягаються в рамках співпраці різних крос-функціональних команд.

4. DSDM заснований на концепції швидкої розробки додатків (Rapid Application Development, RAD). Являє собою ітеративний і інкрементний підхід, який надає особливого значення тривалій участі в процесі користувача/споживача.

5. Essential Unified Process (EssUP).

6. Екстремальне програмування (Extreme programming, XP).

7. Feature driven development (FDD) – функціонально-орієнтована розробка. Використовуване в FDD поняття функції або властивості (feature) Системи досить близько до поняття прецеденту використання, використовуваному в RUP, істотна відмінність – це додаткове обмеження: «кожна функція повинна допускати реалізацію не більше, ніж за два тижні». Тобто якщо сценарій використання досить малий, його можна вважати функцією. Якщо ж великий, то його треба розбити на декілька відносно незалежних функцій.

8. Getting Real – ітераційний підхід без функціональних специфікацій, що використовується для веб-додатків. У даному методі спершу розробляється інтерфейс програми, а потім її функціональна частина.

9. OpenUP – це ітераційно-інкрементний метод розробки програмного забезпечення. Позиціюється, як легкий і гнучкий варіант RUP. OpenUP ділить життєвий цикл проекту на чотири фази: початкова фаза, фази уточнення, конструювання та передачі. Життєвий цикл проекту забезпечує надання зацікавленим особам та членам колективу точок ознайомлення і прийняття рішень впродовж усього проекту. Це дозволяє ефективно контролювати ситуацію і вчасно приймати рішення про задовільність результатів. План проекту визначає життєвий цикл, а кінцевим результатом є остаточний додаток.

10. Scrum встановлює правила керування процесом розробки та дозволяє використовувати вже існуючі практики кодування, коректуючи вимоги або

вносячи тактичні зміни. Використання цієї методології дає можливість виявляти і усувати відхилення від бажаного результату на більш ранніх етапах розробки програмного продукту.

11. Бережлива розробка програмного забезпечення (lean software development). Використовує підходи з концепції бережливого виробництва.

Хоча я реалізовував програму сам, було використано підходи Scrum для саморозвитку та пришвидшенню розробки, розглянемо цей метод. Scrum – підхід управління проектами для гнучкої розробки програмного забезпечення. Скрам чітко робить акцент на якісному контролі процесу розробки.

Підхід вперше описали Гіротака Такеучі та Ікуджіро Нонака в статті The New New Product Development Game (Гарвардський Діловий Огляд, січ–лют 1986). Вони відзначили, що проекти, над якими працюють невеликі, крос–функціональні команди, зазвичай систематично продукують кращі результати, і пояснили це, як «підхід регбі». У 1991 році ДеГрейс та Шталь у книжці Злі проблеми, справедливі рішення послалися на цей підхід, як на Scrum (штовханина; сутичка навколо м'яча (у регбі)), спортивний термін, згаданий в статті Такеучі і Нонака. Кен Швабер на початку 1990–х використовував підхід який привів Scrum в його компанію.

Вперше метод Scrum було представлено на загальний огляд задокументованим, чітко сформульованим та описаним спільно Сазерлендом та Швабером на OOPSLA'96 в Остіні. Швабер та Сазерленд протягом наступних років працювали разом щоб обробити та описати весь їхній досвід та найкращі практичні зразки для індустрії в одне ціле, в ту методологію, що відома сьогодні як Scrum. Швабер об'єднав зусилля з Майком Бідлом в 2001, щоб детально описати метод в книжці Agile Software Development with SCRUM. Не зважаючи на те, що для Scrum нарікли долю управління проектами з розробки ПЗ, він може також використовуватися в роботі команд обслуговувань програмного забезпечення (software maintenance teams), або як підхід управління розробкою і супроводом програм: Scrum of Scrums.

Scrum – це кістяк процесу, який включає набір методів і попередньо визначених ролей. Головні дійові особи – ScrumMaster, той хто опікується процесами, веде їх і працює як керівник проекту, Власник Продукту, людина, що представляє інтереси кінцевих користувачів та інших зацікавлених в продукті сторін, та Команду, яка включає розробників.

Протягом кожного спринту, 15–30 денного періоду (тривалість визначається командою), працівники створюють функціональний ріст програмного забезпечення.

Набір можливостей, які імплементуються кожного спринту, приходять з етапу, що має назву product backlog (документація запитів на виконання робіт), який має найвищу пріоритетність за рівнем вимог до роботи, що повинна бути виконана.

Запити на виконання робіт (backlog items), що визначені протягом наради з планування спринту (sprint planning meeting), переміщуються в етап спринту. Протягом цієї наради Власник Продукту інформує про завдання, які він хоче, аби були виконані. Тоді Команда визначає, скільки з бажаного вони можуть зробити, щоб завершити необхідні частини протягом наступного спринту. Протягом спринту команда виконує визначений фіксований список завдань (т.з. backlog items). Впродовж цього періоду ніхто не має права змінювати перелік запитів на виконання робіт, що слід розуміти, як заморожування вимог (requirements) протягом спринту.

Product backlog – це документ, який має список вимог до функціональності, які упорядковані згідно зі ступенем важливості. Product backlog представляє список того, що повинно бути реалізовано. Елементи цього списку називається «історіями» (user story) або елементами backlog–у (backlog items). Product backlog відкритий для редагування усім учасникам Scrum–процесу.

Обов'язкові поля:

1. ID – унікальний ідентифікатор, порядковий номер, який використовується для ідентифікації історій у разі їх перейменування.

2. Назва (Name) – стислий опис історії. Він повинен бути однозначним, щоб і розробники і product owner могли зрозуміти, про що йдеться і відрізнити одну історію від іншої.

3. Важливість (Importance) – ступінь важливості даної історії на погляд product owner ’а. Зазвичай являє собою натуральне число, іноді для цієї цілі використовуються числа Фібоначчі. Чим більше значення, тим більше пріоритет.

4. Попередня оцінка (initial estimate) – початкова оцінка об'єму робіт, необхідного для реалізації історії порівняно з іншими історіями. Вимірюється у story point'ах. Приблизно відповідає числу «ідеальних людино–днів».

5. Як продемонструвати (how to demo) – стисле пояснення того, як завершена задача буде продемонстрована у кінці спринта. Дане поле може являти собою код автоматизованого приймального тесту.

Додаткові поля. Іноді, також, використовуються додаткові поля у product backlog, в основному для того, щоб допомогти product owner'у визначитися з його пріоритетами.

Категорія (track). Наприклад, «панель управління» чи «оптимізація». За допомогою цього поля product owner може легко вибрати усі пункти категорії «оптимізація» і задати їм низький пріоритет.

Компоненти (components) – указує, які компоненти (наприклад, база даних, сервер, клієнт) будуть зачеплені при реалізації історії. Дане поле складається з групи checkbox'ів, які відмічаються, якщо відповідні компоненти потребують змін.

Ініціатор запиту (requestor). Product owner може захотіти зберігати інформацію про усіх замовників, зацікавлених у даній задачі. Це потрібно для того, щоб тримати їх у курсі діла про хід виконання робіт.

ID у системі обліку помилок (bug tracking ID) – якщо ви використовуєте окрему систему обліку помилок, тоді у описі історії корисно зберігати посилання на всі дефекти, які до неї відносяться.

Sprint backlog – містить функціональність, обрану Product Owner із Product Backlog. Всі функції розбиті по задачах, кожна з яких оцінюється командою. Кожен день команда оцінює об'єм роботи, який необхідно провести для завершення задачі.

Burndown chart – показує, скільки вже виконано і скільки ще залишається зробити.

Планування спринта (Sprint Planning Meeting)

Проходить на початку нової ітерації Спринта:

– Із Product Backlog обираються задачі, зобов'язання по виконанню яких за спринт приймає на себе команда;

– На основі обраних задач створюється Sprint Backlog. Кожна задача оцінюється у ідеальних людино–годинах;

– Рішення задачі не повинно займати більше 12 годин або одного дня. При необхідності задача розбивається на підзадачі;

– Обговорюється та визначається, яким чином буде реалізовано цей об'єм робіт;

– Тривалість наради обмежена зверху 4–8 годинами в залежності від тривалості ітерації, досвіду команди тощо;

– (перша частина наради) Беруть участь Product Owner + Команда: обирають задачі із Product Backlog;

– (друга частина наради) Бере участь лише команда: обговорюють технічні деталі реалізації, наповнюють Sprint Backlog.

Щоденна нарада (Daily Scrum Meeting)

Відбувається кожен день протягом спринта. Є «пульсом» ходу спринта. Нараді властиві наступні обмеження:

– починається точно вчасно;

– всі можуть спостерігати, але говорять тільки обрані;

– триває не більш ніж 15 хвилин;

– проводиться в одному і тому ж місці протягом одного спринта.

Протягом наради кожен член команди відповідає на 3 запитання:

– Що зроблено з моменту попередньої щоденної наради?

– Що буде зроблено з моменту поточної наради до наступної?

– Які проблеми заважають досягненню цілей спринта? (Над рішенням цих проблем працює ScrumMaster. Зазвичай це рішення проходить за рамками щоденної наради і у складі осіб, що безпосередньо займаються даною перешкодою.)

Демонстрація (Sprint Review Meeting):

– Проходить у кінці ітерації (спринта).

– Команда демонструє внесок функціональності до продукту всім зацікавленим особам.

– Залучається максимальна кількість глядачів.

– Усі члени команди беруть участь у демонстрації (одна людина на демонстрацію або кожен показує, що зробив за спринт).

– Обмежена 4–ма годинами в залежності від тривалості ітерації і змін у продукті.

Ретроспектива (Sprint Retrospective):

– Члени команди висловлюють свою думку про минулий спринт.

– Відповідають на два основних запитання: Що було зроблено добре у минулому спринті? Що потрібно покращити в наступному?.

– Виконують покращення процесу розробки (вирішують питання та фіксують вдалі рішення).

– Обмежена 1–3ма годинами.

4.2 Захист розробленого програмного забезпечення

Захист розробленого програмного забезпечення буде відбуватися за допомогою Threefish – в криптографії симетричний блоковий криптоалгоритм, розроблений автором Blowfish та Twofish, американським криптографом Брюсом

Шнайером 2008 року для використання в геш-функції Skein і як універсальну заміну наявним блоковим шифрам. Основними принципами розробки шифру були: мінімальне використання пам'яті, необхідна для використання в геш-функції стійкість до атак, простота реалізації та оптимізація під 64-розрядні процесори.

Структура алгоритму

Threefish має дуже просту структуру і може бути використаний для заміни алгоритмів блочного шифрування, бувши швидким і гнучким шифром, що працюють в довільному режимі шифрування. Threefish S-блоки не використовує, заснований на комбінації інструкцій виключаючого або, складання і циклічного зсуву.

Як і AES, шифр реалізований у вигляді підстановочно-перестановочної мережі на оборотних операціях, не будучи шифром мережі Фейстел.

Алгоритм передбачає використання tweak-значення, свого роду вектора ініціалізації, дозволяючи змінювати таким чином значення виходу, без зміни ключа, що має позитивний ефект як для реалізації нових режимів шифрування, так і на криптостойкости алгоритму.

Як результат думки авторів, що кілька складних раундів часто гірше ніж застосування великого числа простих раундів, алгоритм має нетрадиційно велику кількість раундів – 72 або 80 при ключі 1024 біт, проте, за заявою творців, його швидкісні характеристики випереджають AES приблизно вдвічі. Варто зауважити, що через 64-бітної структури шифру, дана заява має місцево лише на 64-розрядної архітектури. Тому, Threefish, як і Skein [1], заснований на ньому, на 32-розрядних процесорах показує значно гірші результати ніж на «рідному» обладнанні.

Ядром шифру є проста функція «MIX», перетворювальна два 64-бітових беззнакових числа, в процесі якої відбувається складання, циклічний зсув (ROL / ROR), і додавання по модулю 2 (XOR).

Нижче представлений код MIX-функції для Threefish-1024 [2]:

```
<syntaxhighlight lang="C">
// Константи для циклічного зсуву
int R16 [8] [8] = {
  {55, 43, 37, 40, 16, 22, 38, 12},
  {25, 25, 46, 13, 14, 13, 52, 57},
  {33, 8, 18, 57, 21, 12, 32, 54},
  {34, 43, 25, 60, 44, 9, 59, 34},
  {28, 7, 47, 48, 51, 9, 35, 41},
  {17, 6, 18, 25, 43, 42, 40, 15},
  {58, 7, 32, 45, 19, 18, 2, 56},
  {47, 49, 27, 58, 37, 48, 53, 56},
};
// D – раунд, j – індекс в таблиці циклічного зсуву
void mix (int j, int d) {
  usigned long long rotl;
  y [0] = x [0] + x [1];
  rotl = R16 [d% 8] [j];
  y [1] = (x [1] << rotl) | (x [1] >> (64 - rotl));
  y [1] ^ = y [0];
}
</Source>
```

Процедура розшифрування обернена процедурі зашифрування і містить зворотну функцію DEMIX.

Кожен з 72 раундів Threefish-256 і Threefish-512 має чотири MIX перетворення, Threefish-1024 – вісім звернень до MIX функції.

Безпека

За заявою авторів, алгоритм має більш високий рівень безпеки, ніж AES. Існує атака на 25 з 72 раундів Threefish, в той час як для AES – на 6 з 10. Threefish має показник фактора безпеки 2.9, в свою чергу, AES всього 1.7 [3]

Для досягнення повної дифузії, шифру Threefish-256 досить 9 раундів, Threefish-512 – 10 раундів і Threefish-1024 – 11 раундів. Виходячи з цього, 72 і 80 раундів відповідно в середньому, забезпечать кращі результати, ніж існуючі шифри. [4]

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

На рисунку 5.1 зображено розроблене у бакалаврської дипломної роботі система підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P. З рисунку можна побачити що інтерфейс головного вікна розподілено на наступні функціональні розділи: Функціональних кнопок ПЗ; Навігаційного меню яке викликається натисканням правої клавіші маніпулятора миші; Верхнього меню; Розділу обрання групи; Розділу виведення результату роботи системи.

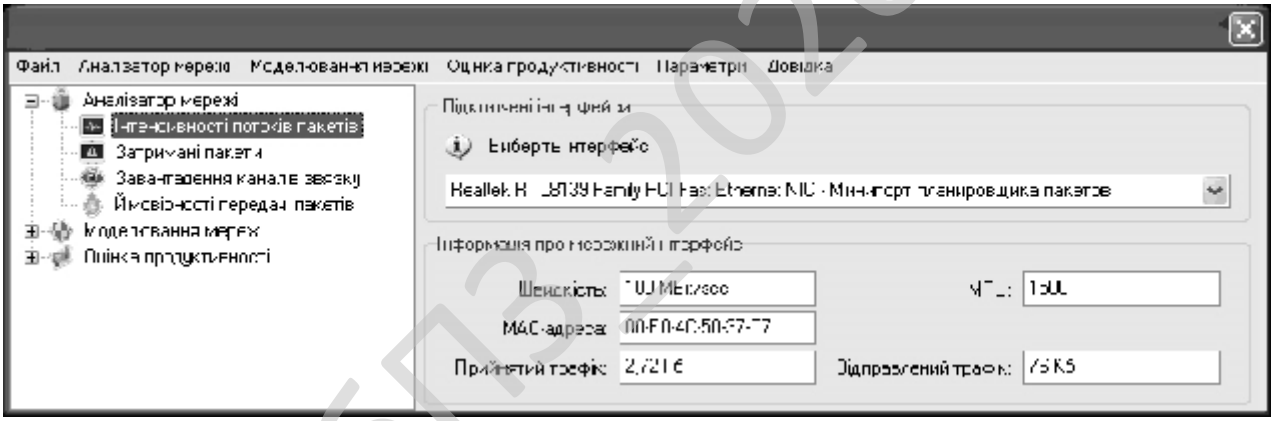

Рисунок 5.1 – Головне вікно ПЗ

Областю застосування є SAN. SAN – Storage Area Network – призначена для консолідації дискового простору серверів на спеціально виділених дискових сховищах. Суть у тому, що так дискові ресурси ощадливіше використовуються, легше управляються й мають більшу продуктивність. А в питаннях віртуалізації й кластеризації, коли декільком серверам потрібний доступ до одного дискового простору, подібні системи зберігання даних взагалі незамінна штука.

Розроблена програма має дуже простий і зрозумілий інтерфейс з користувачем. Кожен, хто в достатньому обсязі володіє операційним

середовищем Windows без особливих складностей освоїть і цю програму, оскільки її інтерфейс інтуїтивно зрозумілий. Якщо програма не видала ніяких помилок, і працює, то можна використовувати, інакше слід слідувати інструкціям, які пропонує програма.

На рисунку 5.2 зображено авторські дані розробленого програмного забезпечення.

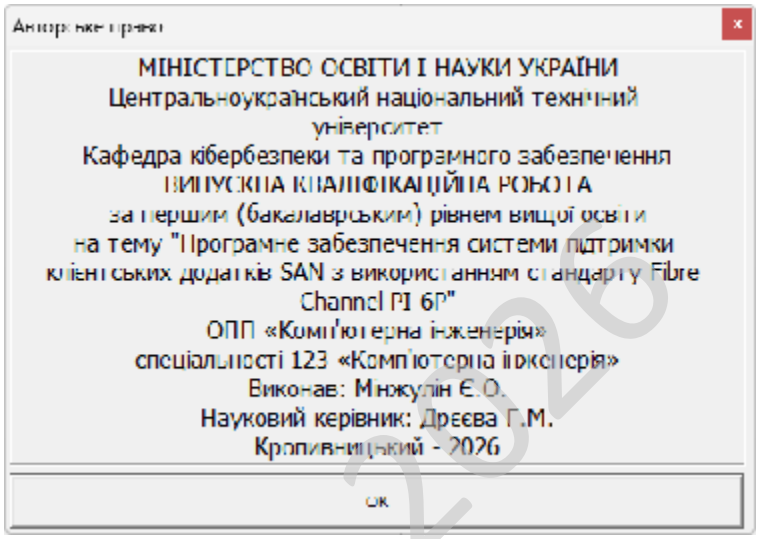

Рисунок 5.2 – Авторське право

Розглянемо процес впровадження програмного забезпечення, це процес налаштування програмного забезпечення під певні умови використання, а також навчання користувачів роботі з програмним продуктом. Впровадження програмного забезпечення це усі дії, що роблять розроблену програмну систему готовою до використання. Даний процес є частинною життєвого циклу програмного забезпечення.

Загалом процес розгортання складається з кількох взаємопов'язаних дій із можливими переходами між ними. Ця активність може відбуватися як з боку виробника так і з боку споживача. Оскільки кожна програмна система є унікальною, то усі процеси та процедури під час розгортання важко передбачити. Тому, "розгортання" можна трактувати як загальний процес відповідно до певних

вимог та характеристик. Розгортання може здійснюватись програмістом і в процесі розробки програмного забезпечення.

До діяльностей пов'язаних із розгортанням програмного забезпечення відносять:

– Випуск.

– Встановлення та активація.

– Деактивація.

– Адаптація.

– Обновлення.

– Вмонтування.

– Відстежування версій.

– Видалення.

– Віддалення з обігу.

При впровадженні програмного забезпечення потрібно урахувати наступні дії:

– Виділення критичних, з точки зору загального результату, процедур в діяльності організації. Коли набір таких процедур визначений, необхідно в першу чергу використовувати IT рішення для автоматизації операцій усередині саме цих процедур. Таким чином, розроблене IT рішення автоматично стає життєво важливим і затребуваним для організації, а також буде забезпечена публічність процесу впровадження;

– Розширення нормативної бази організації шляхом включення до неї регламентів, що описують порядок виконання процедур автоматизованих процесів. В іншому випадку є небезпека виникнення неузгодженості між автоматизованими процедурами та іншими процесами організації.

– Виконання робіт з загальної стандартизації існуючої діяльності організації, коли виділяються кращі практики виконання процедур і включаються в IT рішення за принципом найбільшої корисності для більшості учасників. Відсоток таких процедур щодо загального обсягу автоматизації може бути

невеликий, але це надає процесу побудови рішення вагу в організації за рахунок збільшення його необхідності.

Проводилось тестування чорної скриньки. Основне місце програми тестів «чорної скриньки» – інтерфейс ПЗ. Відомі: функції програми. Досліджується: робота кожної функції на всій області визначення.

Ці тести демонструють:

– Як виконуються функції програми.

– Як приймаються вихідні дані.

– Як виробляються результати.

– Як зберігається цілісність зовнішньої інформації.

При тестуванні «чорної скриньки» розглядаються системні характеристики програм, ігнорується їхня внутрішня логічна структура. Вичерпне тестування, як правило, неможливе.

Наприклад, якщо в програмі 10 вхідних величин і кожна приймає по 10 значень, то кількість тестових варіантів становитиме 10^10. Тестування «чорної скриньки» не реагує на багато особливостей програмних помилок.

Тестування «чорної скриньки» (функціональне тестування) дозволяє отримати комбінації вхідних даних, які забезпечують повну перевірку всіх функціональних вимог до програми.

Програмний виріб тут розглядається як «чорна скринька», чию поведінку можна визначити тільки дослідженням його входів та відповідних виходів. При такому підході бажано мати:

– Набір, утворений такими вхідними даними, які призводять до аномалій у поведінці програми (назвемо його ITc).

– Набір, утворений такими вхідними даними, які демонструють дефекти програми (назвемо його OT).

Будь-який спосіб тестування «чорної скриньки» повинен:

– Виявити такі вхідні дані, які з високою ймовірністю належать набору ITc.

– Сформулювати такі очікувані результати, які з високою імовірністю є елементами набору ОТ.

Принцип «чорної скриньки» не альтернативний принципу «білої скриньки». Скоріше це доповнює підхід, який виявляє інший клас помилок.

Тестування «чорної скриньки» забезпечує пошук наступних категорій помилок:

– Некоректних чи відсутніх функцій.

– Помилок інтерфейсу.

– Помилок у зовнішніх структурах даних або в доступі до зовнішньої бази даних.

– Помилок характеристик (необхідна ємність пам'яті і т.д.).;

– Помилок ініціалізації та завершення.

Обрано умови розповсюдження – commercial software. Програмне забезпечення, створене комерційною організацією з метою отримання прибутку від його використання іншими, наприклад, шляхом продажу копій.

Найважливішою особливістю комерційний програмних продуктів є підтримка великих компаній, прямо зацікавлених у поширенні програм. Багато організацій надають виключно платну підтримку своїх продуктів, такий підхід, як правило, використовують організації надають відкриті вихідні коди. Для продуктів, що розповсюджуються на комерційній основі діють зазвичай безкоштовні служби підтримки, покликані збільшити рівень довіри у клієнтів і потенційних покупців.

Далеко не завжди, але як правило терміни критично важливих змін в комерційних продуктах значно менше, ніж у некомерційних проектів. Це пов'язано з тим, що над комерційним продуктом працюють цілі групи розробників і ця робота є їх основним заняттям. Розробникам-початківцям як правило доводиться шукати додаткові способи заробітку, і це збільшує час, що витрачається на доповнення і зміни програм.

Так як основним рушійним фактором створення комерційного ПЗ є одержання прибутку, то комерційні програмні продукти першими заповнюють вільні ніші та пропонують варіанти вирішення завдань відразу по мірі виявлення вакууму в будь-якому секторі ринку.

Окремий вид комерційних програм, коли їх розробка оплачується безпосередньо замовником. Такі програми найчастіше позбавлені всіх переваг комерційних продуктів, оскільки мають обмежений бюджет, але більш адаптовані до вимог замовника, ніж аналоги.

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти, призначено для системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

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

Рішення завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

– Досліджена система підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

– На основі отриманих результатів досліджень створена програмна реалізація системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

Розроблені під час виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P.

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 призначені для

системи підтримки клієнтських додатків SAN з використанням стандарту Fibre Channel-PI-6P. 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.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм Threefish.

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. 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.

2. 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.

3. 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.

4. 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.

5. Smirnova, T., Gnatyuk, S., Yudin, O., Sydorenko, V., Polozhentsev, A., «The Model for Calculating the Quantitative Criteria for Assessing the Security Level of Information and Telecommunication Systems». CEUR Workshop Proceedings Volume 3156, 2022, Pages 390-399.

6. Смірнова Т.В., Гнатюк С.О., Сидоренко В.М., Юдін О.Ю., Сидоренко С.Ю., «Модель визначення критичності галузевих інформаційно-телекомунікаційних систем». *Проблеми інформатизації та управління,* № 2(70). 2022. С. 28-37.

7. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Смірнов С.А., Поліщук Л.І., «Дослідження стійкості до диференціального криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в

інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку,* 2022, № 3(69). С. 93-98.

8. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Поліщук Л.І., Смірнов С.А. «Дослідження статистичної стійкості та швидкісних характеристик запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Вісник Хмельницького національного університету. Серія: «Технічні науки»,* № 2 (307). С. 46-52. 2022.

9. Смірнов О.А., Смірнова Т.В., Константинова Л.В., Смірнов С.А., Якименко Н.М., «Дослідження стійкості до лінійного криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку*, 2022, № 1(67). С. 84-89.

10. 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.

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

12. Smirnov O., Alimseitova Zh., Adranova A., Akhmetov B., Lakhno V., Zhilkishbayeva G. «Models and algorithms for ensuring functional stability and cybersecurity of virtual cloud resources». *Journal of theoretical and applied information technology* Vol.98. No 21, 2020, P. 3334-3346.

13. 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.

14. Smirnov O., Kuznetsov A., Pushkar'ov A., Serhiienko R., Babenko V., Kuznetsova T., «Representation of Cascade Codes in the Frequency Domain». In: Radivilova T., Ageyev D., Kryvinska N. (eds) Data-Centric Business and Applications. Lecture Notes on Data Engineering and Communications Technologies, vol 48. Springer, Cham. 2021. pp 557-587.

15. Smirnov, O., Markovets, O. Vovk, N., Turchyn, Y., «Model of informational support for social network administrators' content creation». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 125-136.

16. Smirnov, O., Drieieva, H., Drieiev, O., Polishchuk, Y., Brzhanov, R., Aleksander, M. «Method of fractal traffic generation by a model of generator on the graph». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 366-379.

17. 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.

18. 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.

19. 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.

20. 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.

21. Smirnov, O., Odarchenko, R., Abakumova, A., Usik, P., Kundyz, M., «QoE optimization technique for media delivery in 5G networks». *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T),* Kyiv, Ukraine, 8 – 11 October 2019. P.597-601.

22. 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.

23. 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.

24. Smirnov, O., Kuznetsov, A., Kiian, A., Zamula, A., Rudenko, S., Hryhorenko, V., «Variance Analysis of Networks Traffic for Intrusion Detection in Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 353-358.

25. Smirnov, O., Kuznetsov, A., Kavun, S., Babenko, B., Nakisko, O., Kuznetsova, K., «Malware Correlation Monitoring in Computer Networks of Promising Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 347-352.

26. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Pastukhov, M., Kuznetsova, K., Prokopovych-Tkachenko, D., «Discrete Signals with Special Correlation Properties», *CEUR Workshop Proceedings* Volume 2353, *CEUR Workshop Proceedings* 2019, Pages 618-629.

27. Smirnov A.A., Kuzne tsov A.A., Danilenko D.A., Berezovsky A., «The statistical analysis of a network traffic for the intrusion detection and prevention systems», *Telecommunications and Radio Engineering*. – Volume 74, Issue 1. – Begel House Inc. – 2015. – P. 61-78.

28. Смірнов О.А., Смірнова Т.В., Буравченко К.О., Кравченко С.С., Горбов В.О., «Хмарна система підтримки прийняття рішень технологічного

процесу відновлення поверхонь конструкцій і деталей машин». *Сучасні інформаційні системи*. 2021. Т. 5, № 4. С. 79-95

29. 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.

30. 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.

31. Смірнов О.А., Конопліцька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В., Поліщук Л.І. Інформаційна безпека в комп'ютерних мережах. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2020. – 294 с.

32. О.А. Смірнов, П.С. Усік, «Дослідження перспектив використання технологічних рішень в мережах 5G» у *Кібербезпека та інформаційні технології: монографія.* – Х. : ТОВ «ДІСА ПЛЮС», 2020.С. 122-135.

33. 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.

34. Смірнов О.А., Конопліцька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В. Поліщук Л.І. Проектування комп'ютерних систем та мереж. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2019. – 264 с.

35. 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).

36. Смірнов О.А., Дрєєва Г.М. Метод генерування фрактального трафіку за допомогою моделі генератора на графі. Монографія: Інформаційна безпека та інформаційні технології : монографія / за заг. ред. В. С. Пономаренка. – Х. : Вид. Рожко С.Г. 2019. С. 123-139

37. Дрєєва Г.М., Смірнов О.А., Дрєєв О.М. Метод генерування фрактальноподібної числової послідовності на основі скінченного автомату для моделювання трафіку у мережі. Центральноукраїнський науковий вісник. Технічні науки. № 1(32). с. 173-183, 2019.

38. 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.

39. Смірнов О.А., Смірнов С.А., Поліщук Л.І., Смірнова Т.В., Конопліцька-Слободенюк О.К. Метод формування антивірусного захисту даних з використанням безпечної маршрутизації метаданих. Кібербезпека: освіта, наука, техніка. – Том 3 № 3. – Київ: КУ ім. Бориса Грінченка. – 2019. – С. 63-87.

40. Смірнов О.А., Гнатюк С.О., Кавун С.В., Терейковський І.А., Жмурко Т.О., Смірнов С.А., Коваленко А.С. Основи безпеки в комп'ютерних мережах. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2018. – 177 с.

41. Смірнов О.А., Котелянець В.В. Стійкі до колізій стохастичні моделі функціонування безпроводових сенсорних мереж. Вісник інженерної академії України, №3, с. 145-152, 2018

42. Смірнов О.А., Смірнов С.А., Дідик А.К., Дрєєв А.М. Алгоритми формування безлічі маршрутів передачі метаданих у антивірусні хмарні системи. Збірник наукових праць "Системи обробки інформації". - Випуск 5 (142). - Х.: ХУПС - 2016. - С. 148-152.

43. Смірнов О.А., Смірнов С.А. Дідик А.К., Дрєєв О.М. Моделі системи нейромережевих експертів безпечної маршрутизації у хмарних

антивірусних системах. Збірник наукових праць "Системи обробки інформації". - Випуск 3 (140). - Х.: ХУПС - 2016. - С. 36-39.

44. Смірнов О.А., Смірнов С.А., Дідик А.К., Дрєєв А.М. Спосіб контролю ліній зв'язку телекомунікаційної системи антивірусу. Збірник наукових праць Харківського університету Повітряних Сил. Випуск 2 (47). – Харків: ХУПС. - 2016. - С. 121-127.

45. Смірнов О.А., Смірнов С.А., Дідик А.К. Метод безпечної маршрутизації метаданих у хмарні антивірусні системи. Системи озброєння та військова техніка. - Випуск 2 (46) - Х.: ХУПС - 2016. - С. 146-149.

46. Смірнов О.А., Кавун С.В., Доренський О.П., Вялкова В.І. Інформаційна безпека в комп'ютерних мережах. Навчальний посібник – Кіровоград: РВЛ КНТУ, 2016. – 151 с.

47. Смірнов О.А., Кавун С.В., Коваленко О.В., Дрєєв О.М. Мережні інформаційні технології. Навчальний посібник – Кіровоград: РВЛ КНТУ, 2016. – 159 с.

48. Смірнов О.А., Кавун С.В., Коваленко О.В., Доренський О.П., Дрєєв О.М., Вялкова В.І. Комп'ютерні мережі. Навчальний посібник – Кіровоград: РВЛ КНТУ, 2016. – 233 с.

49. Смірнов О.А., Стасєв Ю.В. Бараннік В.В. Захист інформації в автоматизованих системах управління. Навчальний посібник – Харків: ХУПС, 2015. – 264 с.

50. Смірнов О.А., Мелешко Є.В., Семенов С.Г. Методи та засоби обробки сигналів і даних в інформаційних системах. Навчальний посібник. – Кіровоград: КНТУ 2012. – 250 с.

51. Смірнов О.А., Євсеєв С.П., Жукарев В.Ю., Король О.Г., Сорокін В.Є., Мелешко Є.В. Технології і стандарти комп'ютерних мереж. Навчальний посібник. – Кіровоград: КНТУ 2012. – 454 с