

”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 проєктування та
розгортання хмарної інформаційної системи з оцінюванням
продуктивності та масштабованості”

Виконав здобувач вищої освіти
IV курсу, групи КН-22
ОПП «Комп’ютерні науки»
спеціальності 122 «Комп’ютерні науки»
_____ Щиченко О.А.
« ____ » _____ 2026 р.

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

Центральноукраїнський національний технічний університет
Факультет Механіко-технологічний
Кафедра Кібербезпеки та програмного забезпечення
Освітній ступінь бакалавр
Галузь знань . 12 “Інформаційні технології”
Спеціальність 122 “Комп’ютерні науки”
Освітньо-професійна (освітньо-наукова) програма “Комп’ютерні науки”

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

Щиченку Олексію Андрійовичу

(прізвище, ім'я, по батькові)

1. Тема роботи

Програмне забезпечення системи проєктування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості

2. Керівник роботи

Смірнова Тетяна Віталіївна, канд. техн. наук, доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 116-02 від 06.02.2026 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 проєктування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості*

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

Смірнова Т.В.
(прізвище та ініціали)

Завдання прийнято до виконання
« 15 » січня 2026 р.

Підпис здобувача

Щиченко О.А.
(прізвище та ініціали)

АНОТАЦІЯ

Щиченко О.А. Програмне забезпечення системи проєктування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості. 122 Комп'ютерні науки. Центральноукраїнський національний технічний університет. Кропивницький. 2026.

В даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи проєктування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості.

Метою розробки є програмне забезпечення системи проєктування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості.

Результат роботи – програмна реалізація системи проєктування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості.

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

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

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

Програму розроблено в середовищі Python.

Ключові слова: комп'ютерні науки, хмарні технології, бізнес-процеси підприємства

ABSTRACT

Shchychenko O.A. Software for the design and deployment of a cloud information system with performance and scalability assessment. 122 Computer Science. 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, which is intended for the design and deployment of a cloud information system with performance and scalability assessment.

The purpose of the development is software for the design and deployment of a cloud information system with performance and scalability assessment.

The result of the work is the software implementation of the design and deployment of a cloud information system with performance and scalability assessment.

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 the Python environment.

Keywords: computer science, cloud technologies, enterprise business processes

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ	2
ВСТУП.....	4
1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ	6
1.1 Призначення системи.....	6
1.2 Область застосування.....	7
2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ	8
2.1 Огляд існуючих систем, технологій, архітектур та програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти.....	8
2.2 Обґрунтування вибору засобів для побудови системи та мови програмування.....	20
2.3 Розгорнута постановка завдання	22
3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ	24
3.1 Опис функціонування системи	24
3.2 Розробка структурної схеми.....	30
3.3 Розробка функціональної схеми	36
3.4 Розробка діаграми процесів.....	39
4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ.....	42
4.1 Розробка блок-схем та опис алгоритмів функціонування системи.....	42
4.2 Захист розробленого програмного забезпечення.....	63
5 ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ	66
6 ОСНОВНІ ВИСНОВКИ.....	72
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	74

					ВКРБ-122.26.0035.00.00.ПЗ			
Вим	Арк.	№ докум.	Підп.	Дата	<i>Програмне забезпечення системи проектування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості</i>	Літ.	Аркуш	Аркушів
<i>Розроб.</i>	Щиченко О.А.					Б	1	81
<i>Перев.</i>	Смірнова Т.В.							
<i>Н.контр.</i>	Коваленко А.С.					ЦНТУ КН-22		
<i>Затв.</i>	Смірнов О.А.							

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

ААУ	—	автономне адаптивне управління
БД	—	блок датчиків
БЗ	—	база знань
БОС	—	блок оцінки стану
БПР	—	блок прийняття рішень
ГПІ	—	графічний інтерфейс користувача (GUI)
ВО	—	виконавчий орган
ЕОМ	—	електрона обчислювальна машина
КА	—	кінцевий автомат
НРС	—	недетермінований автомат Рабина-скотта
НМ	—	нейронна мережа
МНРС	—	модифікований недетермінований автомат Рабина-скотта
НАНУ	—	Національна академія наук України
ОУ	—	об'єкт управління
ПЧО	—	просторово-часовий образ
ВВ	—	випадкова величина
ПЗ	—	програмне забезпечення
СПДНМ	—	система побудови й дослідження нейронних мереж
УС	—	управляюча система
ФР	—	функція розподілу
ФРО	—	апарат формування й розпізнавання образів
Z^+	—	множина ненегативних цілих чисел
$G(V, N)$	—	граф із множиною вершин V і множиною ребер N
$i \rightarrow j$	—	ребро, спрямоване з вершини i у вершину j
$X \leftrightarrow Y$	—	взаємодозначне відображення множини X на множину Y
$\pi(X)$	—	множина кінцевих підмножин множини X

$R[a,b]$	–	множина речовинних чисел на $[a,b]$, $R \equiv R[-\infty, +\infty]$
B^N	–	простір двійкових векторів розмірності N
Λ	–	порожнє слово із множини вхідних слів КА
0	–	<i>неправда</i> у вираженні тризначної логіки
1	–	<i>істина</i> у вираженні тризначної логіки
$\diamond$	–	<i>невизначеність</i> у вираженні тризначної логіки
$\vec{X} \subset \vec{Y}$	–	$\vec{X}$ є підвектор (сукупність обраних компонентів) вектора $\vec{Y}$
$X \supset Y$	–	клас Y є нащадком класу X

К6ПЗ_2026

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		3

ВСТУП

Актуальність теми. У 2026 році операційна ефективність – це не просто мета, а необхідність. Оскільки компанії орієнтуються на дедалі складніших ринках, програмне забезпечення для управління бізнес-процесами надає набір інструментів для проектування, автоматизації, управління та оптимізації бізнес-процесів. Воно діє як центральна нервова система організації, допомагаючи командам планувати поточні процеси, виявляти вузькі місця, автоматизувати завдання та контролювати продуктивність для оптимізації операцій, зниження витрат та підвищення гнучкості.

Прогнозується, що світовий ринок управління бізнес-процесами зросте до 65,4 мільярда доларів до 2031 року, що підживлюється зростаючою потребою в інструментах автоматизації процесів та цифрової співпраці. Вибір правильного програмного забезпечення – це вже не просте IT-рішення, а стратегічний крок, який може визначити конкурентну перевагу компанії та її майбутній успіх.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи проектування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості.

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

- Огляд існуючих систем проектування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості.
- Дослідження системи проектування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості.
- Програмна реалізація системи проектування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		4

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі проектування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи проектування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

КБПЗ_2026

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		5

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Для продуктового бізнесу характерні інші правила гри й процеси, ніж ті, що властиві проектній діяльності. Якщо по завершенні проекту спілкування із замовником зводиться до мінімуму, то при наданні сервісів із замовником доводиться спілкуватися щодня.

Виділення в окремий бізнес дало крім того більшу свободу дій і дозволило прискорити прийняття рішень. Початий редизайн внутрішніх процесів дозволив оптимізувати роботу й, як затверджується, «позитивно позначився на ціновій політиці».

Хмарні сервіси дозволяють залучати нових замовників, з ким до того ніколи не працював – по внутрішній статистиці 70% це саме нові замовники. При цьому більше 80% замовників розміщують у хмарі бізнес-критичні сервіси. Для забезпечення необхідного ступеня надійності хмара розміщується у власному ЦОД, сертифікованому на відповідність вимогам і Tier 3.

Для пошуку перспективних тиражуємих рішень створена група продуктових інновацій. Чому ми цим зайнялися саме зараз? Ми усвідомили, що вміємо це робити. Прикладами можуть служити як підрозділ хмарних сервісів, так і підрозділ комунікацій, де успішно розвиваються керовані сервіси, однак, щоб процес виводу нових продуктових напрямків прискорився, його треба виділити.

Завдяки трансформації й новим напрямкам вдасться збільшити виторг на 20%, однак основний дохід (90% оборту) принести – і ще видимо довго буде приносити – проектний бізнес.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		6

1.2 Область застосування

Областю застосування є бізнес-процеси підприємства. Управління бізнес-процесами, або BPM, – це дисципліна, що зосереджена на вдосконаленні бізнес-процесів від початку до кінця. Вона включає виявлення, моделювання, аналіз, вимірювання та оптимізацію робочих процесів для відповідності цілям організації. Система управління бізнес-процесами (BPMS) – це програмне забезпечення, яке забезпечує цю дисципліну.

Подумайте про це так: BPM – це стратегія, а BPMS – це технологія, яку ви використовуєте для реалізації цієї стратегії. Програмне забезпечення надає централізовану платформу для управління та автоматизації складних робочих процесів, які часто охоплюють кілька відділів і систем, таких як платформи ERP та CRM. Перетворюючи ручні, повторювані завдання на оптимізовані цифрові робочі процеси, BPMS звільняє вашу команду, щоб вона могла зосередитися на більш стратегічній, високоефективній роботі.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи проєктування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		7

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

Тепер, коли ми розглянули ключові функції, які слід шукати в BPM-рішенні, давайте заглибимося в найкращі платформи, доступні на ринку. Наступні десять варіантів програмного забезпечення для управління бізнес-процесами були обрані завдяки їхньому комплексному набору функцій, зручності використання та високій репутації на ринку. Кожен із цих інструментів пропонує потужний спосіб автоматизації робочих процесів, підвищення ефективності та стимулювання цифрової трансформації у вашій організації.

Appian

Appian об'єднує робочі процеси, дані та штучний інтелект на платформі з низьким кодом, створеній для швидкості в середовищах з високим рівнем відповідності. Якщо вам потрібно швидко створювати безпечні, масштабовані додатки та так само швидко їх змінювати, уніфікований стек Appian забезпечує роботу команд без шкоди для управління.

Найкраще підходить для регульованих підприємств, які бажають швидкої доставки без шкоди для безпеки чи масштабу.

Чому це відчувається швидко:

- Візуальне моделювання BPMN/DMN, яке виконується безпосередньо для узгоджених, керованих робочих процесів.
- Рідний RPA та управління справами для організації роботи людей, ботів та штучного інтелекту в одному місці.
- Штучний інтелект Copilot для пришвидшення будівництва, сортування та аналізу поверхні.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		8

- Попередньо створені конектори, низькокодові API та CI/CD для швидкого підключення вашого стеку.
- Аналіз та аналіз процесів, що виявляють та усувають вузькі місця.
- Розгортання корпоративного рівня: Appian Cloud (FedRAMP/HIPAA) або власне розміщення.
- Ціни та налаштування: короткий огляд: Пробна версія: Безкоштовна версія для спільноти. Від: POR. Модель: Багаторівнева. Налаштування: Допомога (типове запуск першого застосунку приблизно через 8 тижнів).
- Придатність та контроль: ідеально підходить для комплексної, сумісної оркестрації у великих масштабах; зверніть увагу, що експорт моделей BPMN/DMN до сторонніх інструментів обмежений.

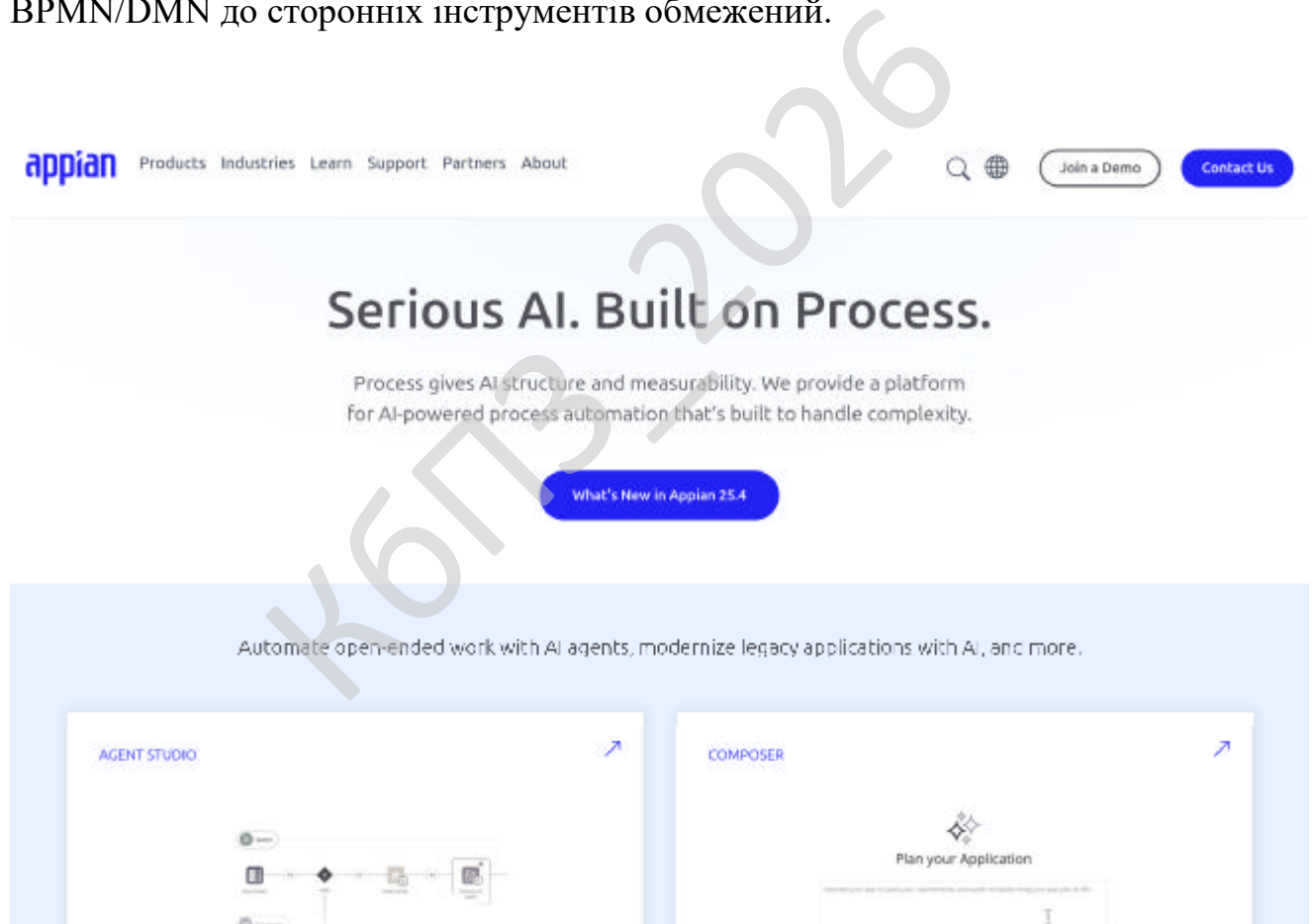


Рисунок 2.1 – Інтерфейс користувача Appian

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		9

Bizagi

Bizagi перетворює моделі BPMN на працюючі, керовані додатки на хмарній платформі, яка масштабується для необмеженої кількості кінцевих користувачів. Завдяки генеративним агентам штучного інтелекту та корпоративному єдиному входу, це скорочує шлях від схеми до розгортання для команд, яким потрібні швидкі результати.

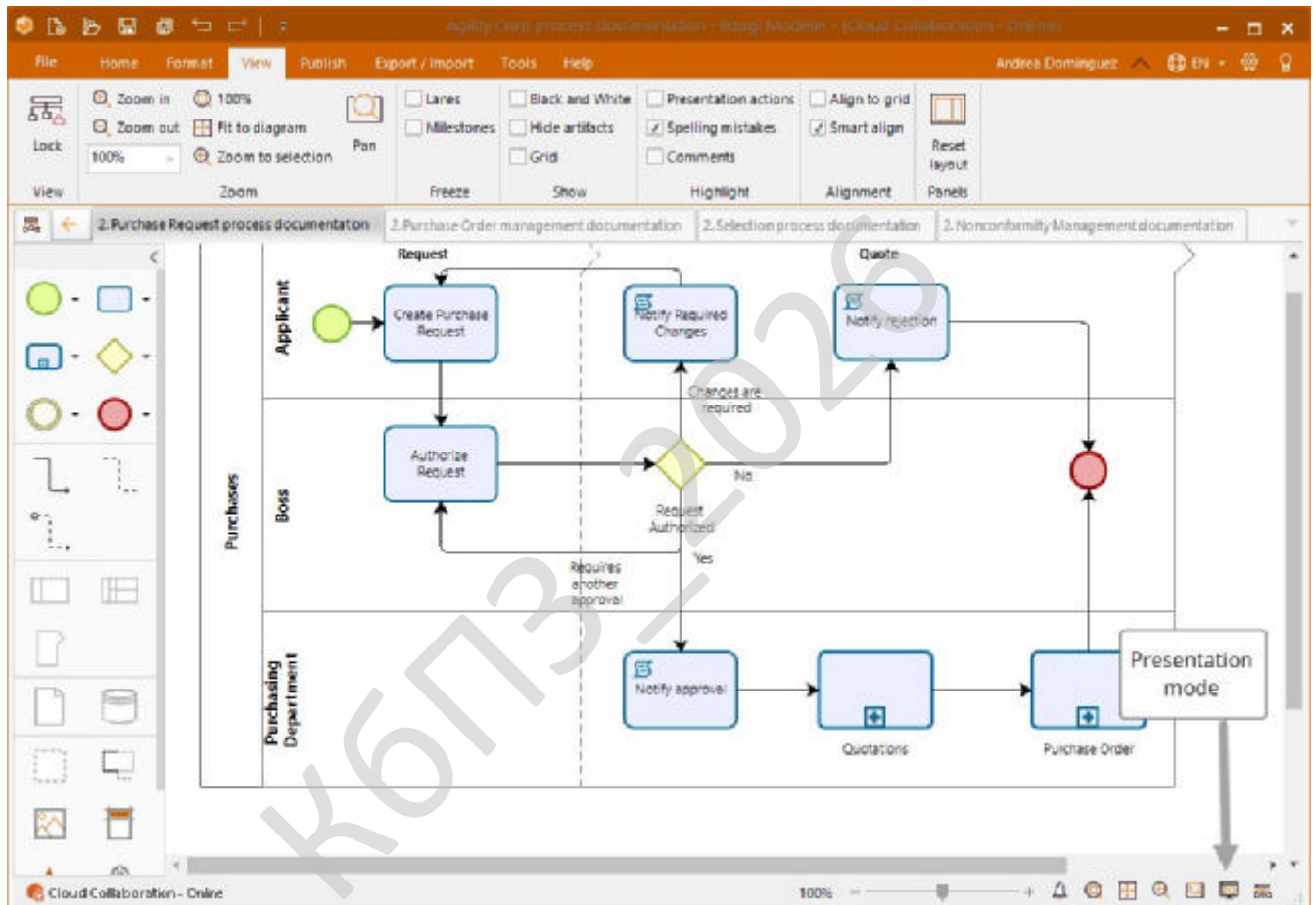


Рисунок 2.2 – Інтерфейс користувача Bizagi

Найкраще підходить для операційних команд середнього та великого бізнесу, яким потрібна схема для виконання додатків у сучасній хмарі.

Чому це відчувається швидко:

- Моделі BPMN 2.0 конвертуються у виконувані робочі процеси за допомогою одного інструменту.

- Низькокодові програми та оркестрація в масштабованому хмарному середовищі виконання.
- Агенти штучного інтелекту для попереднього заповнення форм та автономного керування інтеграціями.
- Конектори для основних систем (наприклад, SAP) плюс REST API.
- Аналітика в реальному часі та надійний захист із вбудованим SSO/RBAC.
- Ціноутворення та налаштування: короткий огляд: Пробна версія: безкоштовний Modeler; пробні версії платформи (до 30 днів) через продажі. Від: POR. Модель: на основі цінової пропозиції. Налаштування: з помічником.
- Підходить для команд, які надають пріоритет швидкому отриманню прибутку завдяки штучному інтелекту; зверніть увагу, що це, перш за все, хмара, і потрібна індивідуальна цінова пропозиція.

Bonita BPM

Bonita поєднує гнучкість відкритого коду з корпоративним підходом, що дозволяє командам розробників швидко створювати додатки для процесів та масштабувати їх за потреби. Її генератор BPMN зі штучним інтелектом та конструктор інтерфейсу користувача з низьким кодом допомагають вам створювати зручні робочі процеси без складного сховища.

Найкраще підходить для команд інженерів, які цінують відкрите ядро, контроль та простий шлях оновлення до корпоративних функцій.

Чому це відчувається швидко:

- Генерація BPMN за допомогою штучного інтелекту для прискорення проектування процесів.
- Оркеструє роботу людей, RPA-ботів та служб штучного інтелекту пліч-о-пліч.
- Конструктор коду з низьким рівнем коду на основі React для відшліфованих, настроюваних інтерфейсів.
- Конектори та API для швидкої інтеграції з існуючими системами.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		11

- Панелі інструментів для відстеження продуктивності та виявлення вузьких місць.
- Гнучке розгортання: локально або в хмарі з корпоративним єдиним вхідним входом (SSO).
- Ціноутворення та налаштування: короткий огляд: Пробна версія: POR. Від: Безкоштовно (Спільнота). Модель: Багаторівнева. Налаштування: Самостійне/допомога. Marketplace: Доступні конектори та програми.
- Підходить та захищає: ідеально підходить для команд, які прагнуть автономії з відкритим кодом; версія для спільноти має обмеження щодо виробництва, тому плануйте з урахуванням потреб підприємства.

Camunda

Camunda – це платформа для оркестрації процесів, розроблена в першу чергу для розробників, яка масштабується від окремого сервісу до критично важливої комплексної автоматизації. Моделювання на основі стандартів та високопродуктивний механізм роблять її улюбленою для інженерних команд, яким потрібна точність та стійкість.

Найкраще підходить для команд розробників програмного забезпечення, які координують складні потоки між мікросервісами, людьми та застарілими системами.

Чому це відчувається швидко:

- Візуальна BPMN/DMN для складної логіки, заснована на відкритих стандартах.
- Високопродуктивний механізм для координації послуг, людей та угод про рівень обслуговування (SLA).
- Інтеграції агентів штучного інтелекту (OpenAI, Gemini) для автономного використання інструментів.
- Конектори та SDK Marketplace для підключення будь-якої кінцевої точки.
- Вбудований моніторинг у режимі реального часу та усунення

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		12

несправностей.

– Ціни та налаштування: короткий огляд: Пробний період: 30 днів. Від: на основі цінової пропозиції. Модель: багаторівнева/безкоштовна. Налаштування: самостійне. Торговельний майданчик: так.

– Придатність та контроль: ідеально підходить для оркестрації під керівництвом інженерів у великих масштабах; зверніть увагу, що багатокористувацька оренда обмежена самостійно керованими розгортаннями.

jBPM

jBPM надає командам Java легкий набір інструментів з відкритим кодом для перетворення діаграм BPMN на працюючі сервіси з повним контролем. Це універсальний дистрибутив і ліцензія Apache 2.0 забезпечують гнучкість і незалежність від постачальника.

Найкраще підходить для команд, орієнтованих на Java, які надають перевагу самостійному розміщенню та детальному контролю над середовищем виконання.

Чому це відчувається швидко:

- Моделювання та виконання BPMN 2.0 та DMN на основі стандартів.
- Оркестрація KIE Server з REST, JMS та Java API.
- Бібліотека конекторів (REST, Slack, Docker) для вашої зручності.
- Вбудовані інформаційні панелі плюс метрики Prometheus/Grafana.
- Локальне розгортання з Docker/Spring Boot, RBAC та SSO.
- Ціноутворення та налаштування: короткий огляд: Спільнота: безкоштовно (Apache 2.0). Підтримка підприємства доступна через IBM Process Automation Manager. Налаштування: самостійне обслуговування (Docker або ZIP).
- Підходить та підлягає контролю: чудово підходить для команд платформ, що працюють на Java; це не керований SaaS, тому очікуйте власного хостингу та операційної діяльності.

Pegasystems (Pega)

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		13

Pega – це потужний low-code-сервер для автоматизації складних процесів корпоративного рівня з чітким управлінням. GenAI Blueprint прискорює шлях від ідеї до робочого плану, щоб великі команди могли швидко рухатися, не втрачаючи контролю.

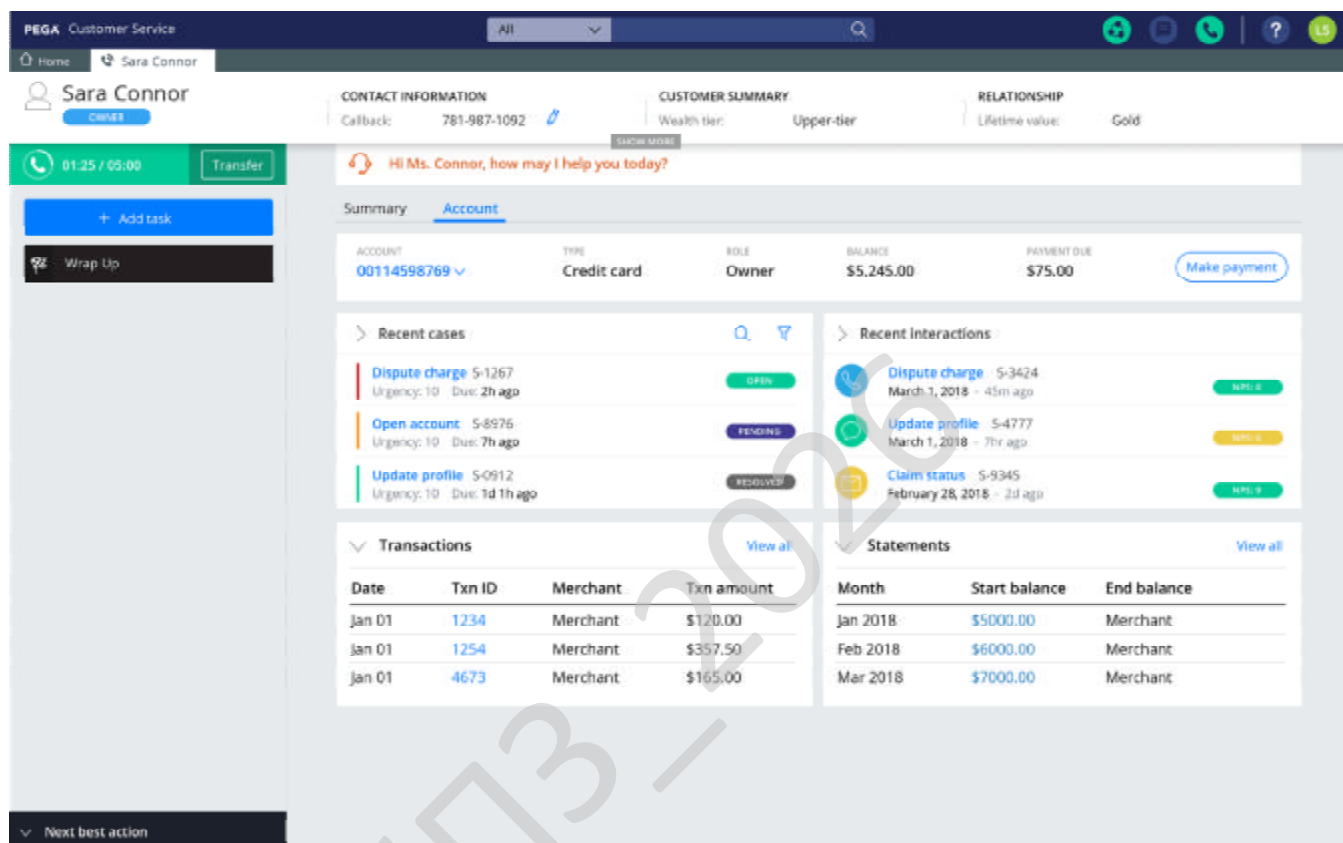


Рисунок 2.3 – Інтерфейс користувача Pega

Найкраще підходить для підприємств, яким потрібні масштаб, відповідність вимогам та постійні зміни в усіх сферах бізнесу.

Чому це відчувається швидко:

- GenAI Blueprint допомагає швидко розпочати проектування та створення за допомогою миттєвих прототипів.
- Process Fabric організовує роботу між програмами Pega та інших розробників.
- Поєднує робочий процес з RPA, щоб позбутися ручних, повторюваних

завдань.

- Модельно-орієнтовані API, конектори та потокова передача подій для інтеграцій.
- Вбудований аналіз процесів та моніторинг на базі штучного інтелекту для оптимізації.
- Ціноутворення та налаштування: короткий огляд: Пробний період: 30 днів. Від: POR. Модель: Багаторівнева. Налаштування: Допоміжне. Торгова площа: Доступні шаблони.
- Придатність та контроль: ідеально підходить для керованої автоматизації у великих масштабах; деякі функції GenAI вимагають модернізації Pega Cloud та інтерфейсу користувача.

ProcessMaker

ProcessMaker поєднує низький рівень коду зі штучним інтелектом, щоб допомогти бізнес- та IT-командам розганяти по керованих робочих процесах за години, а не тижні. Генеративні інструменти та агентні завдання скорочують цикли проектування, зберігаючи контроль в руках операторів.

Найкраще підходить для операційних/IT-груп середнього та великого бізнесу, яким потрібне прискорення ІІІ з захистом від пошкоджень.

Чому це відчувається швидко:

- Створюйте моделі BPMN зі звичайного тексту або ескізів за допомогою штучного інтелекту.
- Моделюйте складні правила в таблицях DMN замість власного коду.
- Перетягування конекторів плюс повний REST API для оркестрації.
- Вбудована аналітика та аналіз процесів.
- Безпека підприємства, можливість аудиту та єдиний вхід (SSO) для відповідного розгортання.
- Ціни та налаштування: короткий огляд: Пробна версія: Шаблони з інструкціями. Від: \$3000/місяць (залежно від випадку). Модель: Підприємство. Налаштування: За підтримки спеціального менеджера з питань успіху.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		15

– Придатність та застереження: Чудово підходить для регульованої автоматизації зі штучним інтелектом; початкова ціна може бути високою для використання в невеликих обсягах.

Менеджер автоматизації процесів Red Hat

Цей корпоративний пакет (побудований на jBPM та Drools) надає командам Java/Kubernetes регульований шлях від моделей BPMN/DMN до хмарних нативних сервісів. Він природно підходить для організацій, які вперше використовують OpenShift та стандартизують автоматизацію по всьому стеку.

Найкраще підходить для підприємств зі зрілим рівнем Java/DevOps, яким потрібна швидкість та контроль на власному хостингу.

Чому це відчувається швидко:

– Візуально спроектувати виконувані файли BPMN 2.0 та DMN, а потім розгорнути їх безпосередньо.

– Оркеструйте завдання, виконані людиною, таймери, угоди про рівень обслуговування (SLA) та динамічну роботу з інцидентами.

– Представляйте процеси та рішення як сервіси через REST API.

– Хмарне розгортання на OpenShift з операторами для масштабування.

– Безпечні інтеграції через Kafka, JMS та SSO.

– Ціноутворення та налаштування: короткий огляд: Пробна версія: Самостійне оцінювання. Від: POR (продається за схемою IBM PAMOE). Модель: Передплата. Налаштування: Самостійне обслуговування або за допомогою партнера.

– Відповідність вимогам та застереження: надійний вибір для керованої, самостійно розміщеної автоматизації; зверніть увагу, що брендинг та продажі тепер відповідають стандарту IBM PAMOE.

Creatio

Creatio – це платформа на основі штучного інтелекту, що не потребує написання коду та дозволяє бізнесу та IT-фахівцям спільно розробляти та запускати робочі процеси за лічені дні. Вона поєднує структурований BPMN з

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		16

гнучким управлінням справами та агентами штучного інтелекту, масштабуючись від однієї команди до операцій по всьому підприємству.



Рисунок 2.4 – Інтерфейс користувача Creatio

Найкращий варіант для керівників операційних відділів та ІТ-відділів, які прагнуть швидко окупити витрати, з можливістю самостійного розміщення або запуску в хмарі.

Чому це відчувається швидко:

- Виконуйте структуровані процеси BPMN та ad hoc DCM одночасно.
- Штучний інтелект, агенти без коду та візуальний конструктор для швидкої автоматизації.
- Понад 700 торгових застосунків та відкритих API для інтеграцій.
- Безпечне розгортання в Creatio Cloud або на власному хостингу з SSO/LDAP.

– Панелі моніторингу в режимі реального часу та аналіз процесів для постійного вдосконалення.

– Ціноутворення та налаштування: короткий огляд: Пробний період: 14 днів. Від: \$25/користувач/місяць. Модель: Багаторівнева. Налаштування: Допоміжне. Торгова площа: Так.

– Придатність та застереження: Чудово підходить для уніфікованого BPM на основі штучного інтелекту; очікуйте мінімальних річних зобов'язань та впровадження під керівництвом партнерів у масштабі підприємства.

Платформа K2

K2 (від Nintex) спрощує створення бізнес-додатків за допомогою конструкторів з низьким рівнем коду, об'єктів даних повторного використання та глибокої інтеграції. Він створений для першочергових магазинів Microsoft, яким потрібна безпечна, керована доставка в хмарі та локально.

Найкраще підходить для корпоративних IT-команд, які створюють інтеграційно-важливі додатки з сильним архітектурним контролем.

Чому це відчувається швидко:

– Дизайн робочого процесу з використанням перетягування та повторно використовуваних смарт-об'єктів.

– Немає конекторів коду для REST, SQL, Salesforce та SAP.

– Відкриті API REST та OData для розширення та створення рішень.

– Розгортання на власно розміщеній інфраструктурі або керованій хмарі Nintex K2.

– Управління підприємством за допомогою SSO, RBAC та пакетування CI/CD.

– Ціноутворення та налаштування: короткий огляд: Пробний період: 30 днів (допомога з продажами). Від: на основі цінової пропозиції. Модель: корпоративна ліцензія. Налаштування: допомога.

– Придатність: ідеально підходить, коли важливі глибина інтеграції та управління; зверніть увагу, що конструктор не є суворим механізмом BPMN.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		18

Вартість та ціноутворення на BPMS (загальна вартість володіння)

Розуміння загальної вартості володіння (ТСО) програмним забезпеченням для управління бізнес-процесами є критично важливим для прийняття обґрунтованого інвестиційного рішення. Ціни можуть сильно відрізнятися, а загальні витрати на впровадження коливаються від 10 000 доларів США для базових рішень до понад 500 000 доларів США для складних корпоративних систем.

На загальну ціну впливає кілька факторів:

– Ліцензійна та передплата: це основна вартість програмного забезпечення. Хмарні рішення зазвичай використовують модель передплати на користувача на місяць, яка може коливатися від 25 до понад 200 доларів США на користувача. Локальні рішення часто вимагають більшої початкової плати за безстрокову ліцензію.

– Впровадження та інтеграція: Це часто найбільша категорія витрат, яка становить від 40 до 50% загального бюджету. Вона включає витрати на конфігурацію системи, міграцію даних та інтеграцію з вашим існуючим програмним стеком.

– Налаштування: Якщо ваші процеси дуже складні, вам може знадобитися індивідуальна розробка, що може значно збільшити витрати.

– Навчання та управління змінами: Правильне навчання вашої команди є важливим для успішного впровадження. Зазвичай це займає від 10 до 15% від загальних інвестицій.

– Поточне обслуговування та підтримка: Для локальних рішень це може становити від 15 до 20% від початкової вартості програмного забезпечення щорічно. Для хмарних рішень підтримка зазвичай включена у вартість підписки.

Оцінюючи варіанти, звертайте увагу не лише на початкову ціну. Гнучка, масштабована платформа може мати вищу вартість підписки, але в довгостроковій перспективі може заощадити вам гроші на впровадженні та налаштуванні.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		19

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Як мова програмування обрана Python. Python – високорівнева мова програмування загального призначення з акцентом на продуктивність розроблювача й читаність коду. Синтаксис ядра Python мінімалістичний. У той же час стандартна бібліотека включає великий обсяг корисних функцій.

Python підтримує кілька парадигм програмування, у тому числі структурне, об'єктно-орієнтоване, функціональне, імперативне й аспектно-орієнтоване. Основні архітектурні риси – динамічна типізація, автоматичне керування пам'яттю, повна інтроспекція, механізм обробки виключень, підтримка багатопоточні обчислень і зручні високорівневі структури даних. Код у Python організовується у функції й класи, які можуть поєднуватися в модулі (які у свою чергу можуть бути об'єднані в пакети).

Еталонною реалізацією Python є інтерпретатор CPython, що підтримує більшість активно використовуваних платформ. Він поширюється вільно під дуже ліберальною ліцензією, що дозволяє використовувати його без обмежень у будь-яких застосунках, включаючи пропрієтарні. Є реалізації інтерпретаторів для JVM (з можливістю компіляції), MSIL (з можливістю компіляції), LLVM і інших. Проект PyPy пропонує реалізацію Python на самому Python, що зменшує витрати на зміни мови й постановку експериментів над новими можливостями.

Python – мова програмування, що активно розвивається, нові версії (з додаванням/зміною мовних властивостей) виходять приблизно раз у два з половиною року. Внаслідок цього й деяких інших причин на Python відсутні ANSI, ISO або інші офіційні стандарти, їхня роль виконує CPython.

Python портований і працює майже на всіх відомих платформах – від КПК до мейнфреймів. Існують порти під Microsoft Windows, практично всі варіанти UNIX (включаючи FreeBSD і Linux), Plan 9, Mac OS і Mac OS X, iPhone OS 2.0 і вище, Palm OS, OS/2, Amiga, AS/400 і навіть OS/390, Symbian і Android.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		20

При цьому, на відміну від багатьох портуємих систем, для всіх основних платформ Python має підтримку характерних для даної платформи технологій (наприклад, Microsoft COM/DCOM). Більше того, існує спеціальна версія Python для віртуальної машини Java – Jython, що дозволяє інтерпретаторові виконуватися на будь-якій системі, що підтримує Java, при цьому класи Java можуть безпосередньо використовуватися з Python й навіть бути написаними на Python. Також кілька проектів забезпечують інтеграцію із платформою Microsoft .NET, основні з яких – IronPython і Python.Net.

Python підтримує динамічну типізацію, тобто тип змінної визначається тільки під час виконання. Тому замість «присвоювання значення змінної» краще говорити про «зв'язування значення з деяким ім'ям». У Python є убудовані типи: бульові, рядки, Unicode-рядки, цілі числа довільної точності, числа із плаваючою комою, комплексні числа й деякі інші. З колекцій Python підтримує кортежі (*tuples*), списки, словники (асоціативні масиви) і, починаючи з версії 2.4, безлічі. Всі значення в Python є об'єктами, у тому числі функції, методи, модулі, класи.

Додати новий тип можна або написавши клас (*class*), або визначивши новий тип у модулі розширення (наприклад, написаному мовою C). Система класів підтримує спадкування (одиначне й множинне) і метапрограмування. Можливе спадкування від більшості убудованих типів і типів розширень.

Всі об'єкти діляться на посилальні й атомарні. До атомарного ставляться *int*, *long*, *complex* і деякі інші. При присвоюванні атомарних об'єктів копіюється їхнє значення, у той час як для посилальних копіюється тільки покажчик на об'єкт, таким чином, обидві змінні після присвоювання використовують те саме значення. Посилальні об'єкти бувають змінювані й незмінні. Наприклад, рядки й кортежі є незмінними, а списки, словники й багато інших об'єктів – змінюваними. КORTEЖ у Python є, по суті, незмінним списком. У багатьох випадках кортежі працюють швидше списків, тому якщо ви не плануєте змінювати послідовність, то краще використовувати саме їх.

Мова має чіткий і послідовний синтаксис, продуману модульність й масштабованість, завдяки чому вихідний код написаних на Python програм легко читаємий.

Python – стабільна й розповсюджена мова. Він використовується в багатьох проектах і в різних якостях: як основна мова програмування або для створення розширень і інтеграції застосунків. На Python реалізоване велика кількість проектів, також він активно використовується для створення прототипів майбутніх програм. Python використовується в багатьох великих компаніях.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускню кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи проєктування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості.

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

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		22

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) сформулювати висновки про виконаний обсяг робіт та одержані результати.

К6ПЗ_2026

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		23

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Життєвий цикл BPM: структура для постійного вдосконалення

Успішне впровадження BPM – це не одноразовий проект, а безперервний цикл. Життєвий цикл BPM забезпечує структурований, повторюваний підхід до вдосконалення та покращення ваших операцій. Більшість фреймворків складаються з п'яти ключових етапів.

– **Проектування:** Цей початковий етап включає визначення існуючих процесів та цілей. Зацікавлені сторони складають карту поточних робочих процесів, виявляють неефективність та спільно розробляють ідеальний процес майбутнього стану.

– **Модель:** На цьому етапі ви створюєте візуальне представлення розробленого процесу. Ця модель допомагає візуалізувати потік завдань, даних та рішень, дозволяючи вам моделювати різні сценарії та виявляти потенційні перешкоди перед впровадженням.

– **Виконання:** Тут змодельований процес втілюється в життя. Це включає розробку необхідної автоматизації, інтеграцію систем та навчання членів команди новому робочому процесу. Процес розгортається або як обмежений пілотний проект, або як повномасштабний запуск.

– **Моніторинг:** Після запуску процесу його ефективність відстежується за ключовими показниками ефективності (KPI), такими як час циклу, вартість та рівень помилок. Інформаційні панелі та аналітика в режимі реального часу надають уявлення про те, як функціонує процес, і виявляють області для покращення.

– **Оптимізація:** На основі даних, зібраних під час етапу моніторингу, ви можете приймати обґрунтовані рішення щодо подальшого вдосконалення

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		24

процесу. Цей етап стосується постійного вдосконалення, внесення ітеративних коригувань для підвищення ефективності, зниження витрат та кращої відповідності бізнес-цілям.

Розуміння типів BPM

Управління бізнес-процесами не є універсальним рішенням. Різні процеси мають різні потреби, що призвело до появи трьох основних типів BPM. Хоча сучасні платформи часто розмивають межі між цими категоріями, їх розуміння допомагає вам вибрати інструмент, який відповідає вашим основним цілям.

Людиноцентричний BPM

Цей підхід зосереджений на процесах, що потребують значної взаємодії з людиною, оцінки та співпраці. До цієї категорії належать такі робочі процеси, як адаптація співробітників, затвердження креативних рішень та управління зверненнями до служби підтримки клієнтів. Людиноорієнтована BPMS чудово справляється з управлінням завданнями, сповіщеннями та наданням інтуїтивно зрозумілих інтерфейсів користувача, які допомагають людям пройти через їхню роботу.

Інтеграційно-орієнтований BPM

Інтеграційно-орієнтоване BPM розроблено для процесів, що працюють у кількох програмних системах з мінімальним втручанням людини. Мета полягає в забезпеченні безперебійного потоку даних та автоматизації між такими програмами, як ERP, CRM та інструменти управління ланцюгами поставок. Цей тип BPM значною мірою спирається на API та конектори для оркестрування складних взаємодій між системами.

Документоорієнтований BPM

Коли процес зосереджений на створенні, перевірці та затвердженні документів, найкращим є підхід, орієнтований на документи. Це поширене явище в юридичних, фінансових та відділах комплаєнсу для таких робочих процесів, як управління контрактами, обробка рахунків-фактур або подання нормативних документів. Ці системи керують контролем версій, автоматизованою

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		25

маршрутизацією та журналами аудиту, щоб забезпечити ефективну та безпечну обробку документів.

Основні характеристики та компоненти BPMS

Хоча кожне програмне забезпечення для управління бізнес-процесами відрізняється, більшість платформ професійного рівня мають основний набір функцій, розроблених для розширення можливостей команд та підвищення ефективності.

Ключові компоненти, на які слід звернути увагу:

– Візуальне моделювання процесів: Зручний інтерфейс з функцією перетягування елементів дозволяє користувачам, навіть тим, хто не має технічних знань, розробляти та візуалізувати робочі процеси. Це спрощує планування етапів процесу та визначення областей для покращення.

– Автоматизація без коду/з низьким рівнем коду: ця функція демократизує розробку, дозволяючи бізнес-аналітикам та іншим нетехнічним зацікавленим сторонам створювати та розгортати додатки. Вона значно скорочує час від ідеї до впровадження, сприяючи інноваціям в організації. Для команд, які хочуть створювати швидко та без обмежень, така платформа, як конструктор веб-додатків без коду від WeWeb, поєднує в собі потужність безкодової роботи з можливістю додавання власного коду для повної гнучкості.

– Механізм бізнес-правил (BRE): BRE – це основний компонент, який відокремлює бізнес-логіку від коду програми. Він дозволяє визначати, керувати та автоматизувати бізнес-правила (наприклад, **ЯКЩО** витрати клієнта > \$1000, **ТОДІ** застосовується знижка 10%). Це дозволяє нетехнічним користувачам легко оновлювати логіку прийняття рішень без необхідності залучення розробника, підвищуючи гнучкість та забезпечуючи послідовне прийняття рішень.

– Можливості інтеграції: Потужна BPMS повинна безперешкодно підключатися до іншого програмного забезпечення, на яке покладається ваш бізнес, такого як CRM, ERP та інструменти автоматизації маркетингу. Це забезпечує безперебійний потік інформації між відділами та запобігає ізоляції

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		26

даних. Дивіться повний список інтеграцій WeWeb .

– Аналітика та звітність у режимі реального часу: Інформаційні панелі та налаштовувані звіти надають уявлення про ефективність процесів. Відстежуючи ключові показники ефективності (KPI), такі як час циклу та використання ресурсів, ви можете приймати рішення на основі даних для оптимізації своїх операцій.

– Інструменти для співпраці: такі функції, як призначення завдань, коментарі та сповіщення в режимі реального часу, сприяють спілкуванню та командній роботі в межах платформи. Це забезпечує узгодженість дій усіх учасників та безперебійну роботу процесів.

– Контроль доступу на основі ролей: Адміністратори можуть керувати дозволами користувачів, щоб гарантувати, що співробітники матимуть доступ лише до даних і процесів, що стосуються їхніх ролей, що підвищує безпеку.

Переваги використання BPMS

Впровадження надійного програмного забезпечення для управління бізнес-процесами може докорінно змінити спосіб функціонування організації, забезпечуючи вимірювані покращення в усіх сферах. Мета полягає в тому, щоб вийти за рамки оптимізації окремих завдань і натомість покращити комунікацію та робочі процеси в усій мережі.

Основні переваги:

– Підвищена гнучкість бізнесу: На ринку, що швидко змінюється, здатність до адаптації є вирішальною. BPMS дозволяє компаніям швидко змінювати робочі процеси у відповідь на нові можливості чи виклики, надаючи їм значну конкурентну перевагу. Організації з покращеним BPM на основі штучного інтелекту продемонстрували в 3,2 рази більшу адаптивність до мінливих умов.

– Підвищена ефективність та продуктивність: Автоматизуючи повторювані ручні завдання, програмне забезпечення BPM звільняє співробітників, щоб вони могли зосередитися на більш стратегічних ініціативах.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		27

Це призводить до вищої віддачі, швидшого виконання процесів та значного підвищення загальної продуктивності.

– Зниження витрат та краще використання ресурсів: Оптимізація процесів природним чином призводить до економії коштів. Виявляючи та усуваючи неефективність, підприємства можуть зменшити операційні витрати та краще використовувати свої ресурси. Організації, що використовують інтегроване штучний інтелект BPM, повідомили про 25% скорочення операційних витрат.

– Покращена прозорість та контроль: програмне забезпечення BPM забезпечує чітке, централізоване уявлення про всі бізнес-процеси. Така прозорість дозволяє менеджерам контролювати продуктивність у режимі реального часу, виявляти вузькі місця та приймати обґрунтовані рішення для оптимізації робочих процесів.

– Гарантована відповідність вимогам та безпека: Багато галузей промисловості підпадають під суворі правила. Система управління BPMS допомагає забезпечити відповідність вимогам, створюючи прозорі, стандартизовані робочі процеси з вбудованими механізмами управління та журналами аудиту. Це зменшує ризик людських помилок та високих штрафів.

Готові побачити, як візуальна розробка може трансформувати ваші бізнес-процеси? Дізнайтеся, що ви можете створити за допомогою WeWeb .

Зростання інтелектуального BPM (iBPMS)

Світ управління бізнес-процесами постійно розвивається. Важливим зрушенням є перехід до інтелектуальних комплексів управління бізнес-процесами (iBPMS), які вдосконалюють традиційне BPM за допомогою передових технологій, таких як штучний інтелект (ШІ), машинне навчання (ML) та аналітика в режимі реального часу. iBPMS не просто автоматизує робочі процеси; вона робить їх розумнішими, адаптивнішими та здатними обробляти складні сценарії.

На відміну від стандартної BPMS, iBPMS може аналізувати дані, щоб надавати прогнозні висновки, автоматизувати прийняття складних рішень та

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		28

динамічно коригувати процеси у відповідь на події в режимі реального часу. Ця еволюція є частиною ширшої тенденції, яка називається гіперавтоматизацією, яка поєднує штучний інтелект, машинне навчання та роботизовану автоматизацію процесів (RPA) для автоматизації навіть складної роботи, що ґрунтується на судженнях.

Тенденції, що формують BPMS у 2026 році

У 2026 році кілька ключових тенденцій, значною мірою зумовлених можливостями iBPMS, прискорюють цифрову трансформацію.

– Штучний інтелект та гіперавтоматизація: Штучний інтелект більше не є новинкою; він стає стандартною функцією. Штучний інтелект допомагає з такими завданнями, як узагальнення інформації, класифікація документів та пропонування наступних кроків. До 2026 року близько 64% організацій планують впровадити інструменти ШІ та машинного навчання у своїх системах BPM. Такі інструменти, як WeWeb Copilot, використовують ШІ для перетворення простих англійських підказок на робочі сторінки та робочі процеси.

– Платформи з низьким та нульовим кодом: Зростання популярності розробки з низьким та нульовим кодом продовжує демократизувати автоматизацію. Ці платформи дозволяють нетехнічним користувачам створювати та змінювати власні робочі процеси за допомогою візуальних інтерфейсів перетягування, таких як WeWeb UI Builder . Це прискорює інновації та зменшує залежність від IT-відділів, дозволяючи командам вирішувати власні проблеми з процесами.

– Оркестрація процесів: Компанії відходять від автоматизації ізольованих точок до комплексної оркестрації процесів. Це передбачає об'єднання різномірних систем і команд в єдиний, уніфікований процес, що забезпечує кращу прозорість і контроль над усім робочим процесом.

– Хмарні рішення: Перехід до хмари продовжує домінувати, понад 95% нових цифрових робочих навантажень зараз розгорнуто на хмарних платформах. Хмарні рішення BPM пропонують більшу гнучкість, масштабованість та

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		29

доступність, особливо для малих та середніх підприємств.

– Зосередженість на клієнтському досвіді: Сучасні ініціативи BPM дедалі більше зосереджені на оптимізації робочих процесів, орієнтованих на клієнтів. Завдяки оптимізації таких процесів, як адаптація та підтримка клієнтів, компанії можуть забезпечити більш плавний та прозорий досвід, що сприяє формуванню лояльності.

Ці тенденції підкреслюють рух до більш інтелектуального, доступного та пов'язаного програмного забезпечення для управління бізнес-процесами. Такі платформи, як WeWeb, які побудовані на перетині генерації штучного інтелекту та візуального редагування без коду, ідеально підготовлені, щоб допомогти компаніям скористатися цими досягненнями.

3.2 Розробка структурної схеми

В епоху посилення правил конфіденційності даних, таких як GDPR та ССРА, безпека є невід'ємною частиною будь-якого програмного забезпечення для управління бізнес-процесами. Якщо ви оцінюєте постачальників, перегляньте Угоду про обробку даних WeWeb, щоб дізнатися більше про обробку даних та відповідність вимогам. Система управління бізнес-процесами (BPMS) часто обробляє конфіденційні дані клієнтів та компаній, що робить надійні заходи безпеки важливими для захисту від порушень та забезпечення відповідності вимогам.

Основні функції безпеки, на які слід звернути увагу, включають:

– Шифрування даних: Інструменти BPM вищого рівня використовують передові стандарти шифрування (наприклад, AES 256) для захисту даних як під час їх передачі, так і під час зберігання.

– Контроль доступу на основі ролей: це гарантує, що користувачі матимуть доступ лише до інформації та функцій, необхідних для їхніх ролей, обмежуючи ризик несанкціонованого доступу.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		30

– Безпечні API: Якщо ваша BPMS інтегрується з іншими програмами, вона повинна робити це через безпечні, автентифіковані API, щоб запобігти витоку даних.

– Журнали аудиту: Система повинна реєструвати всі дії, забезпечуючи чіткий облік того, хто що робив і коли. Це має вирішальне значення для дотримання вимог та розслідування будь-яких інцидентів безпеки.

– Безпечний хмарний хостинг: Для хмарного програмного забезпечення хостинг-провайдер повинен мати надійні заходи фізичної та цифрової безпеки, включаючи резервування даних та регулярні аудити безпеки.

Вибір платформи, яка надає пріоритет безпеці, не лише захищає ваш бізнес від фінансової та репутаційної шкоди, але й зміцнює довіру ваших клієнтів.

Управління та підтримка BPMS

Впровадження BPMS – це не просто технічний проект; це довгострокове стратегічне зобов'язання, яке вимагає сильної системи управління та надійної підтримки. Управління процесами встановлює правила, ролі та обов'язки для управління вашими ініціативами BPM, забезпечуючи їх відповідність бізнес-цілям.

Надійна модель управління повинна враховувати:

– Відповідальність за процес: Чітке визначення відповідальних за кожен процес має вирішальне значення для підзвітності. Власники процесів несуть відповідальність за моніторинг ефективності та забезпечення постійного вдосконалення.

– Стандартизація: Встановлення стандартів для проектування, документування та впровадження процесів забезпечує узгодженість та якість у всій організації.

– Управління змінами: Формальний процес пропонування, оцінки та впровадження змін у робочі процеси запобігає відхиленню від процесу та гарантує, що всі зміни принесуть цінність.

– Вимірювання ефективності: Управління визначає ключові показники успіху та регулярну частоту перевірки ефективності, забезпечуючи, щоб зусилля BPM приносили очікувану рентабельність інвестицій.

Не менш важливою є модель постійної підтримки. Чи то через внутрішній Центр передового досвіду (CoE), чи через підтримку постачальників, вашій команді потрібен доступ до технічної експертизи для усунення несправностей, обслуговування та навчання, щоб забезпечити довгостроковий успіх та впровадження користувачами.

Оцінка та впровадження

Успішне впровадження програмного забезпечення для управління бізнес-процесами вимагає більше, ніж просто вибору правильного інструменту. Воно вимагає чіткої стратегії та ретельного планування. Дотримання структурованого посібника може допомогти забезпечити плавний перехід та максимізувати рентабельність інвестицій.

– Визначте свої цілі: Перш за все, чітко визначте, чого ви хочете досягти. Ви намагаєтеся зменшити операційні витрати, покращити обслуговування клієнтів чи підвищити гнучкість? Встановлення конкретних, вимірюваних, досяжних, релевантних та обмежених у часі (SMART) цілей буде орієнтиром для всього проекту.

– Оцініть та складіть карту своїх поточних процесів: Ви не можете покращити те, чого не розумієте. Взаємодійте із зацікавленими сторонами та складіть карту своїх існуючих робочих процесів, щоб виявити вузькі місця, надлишки та області для покращення. Це створює базову основу для вимірювання успіху.

– Оберіть правильну платформу та пілотний процес: оберіть BPMS, яка відповідає вашим цілям та має необхідні функції. Не намагайтеся переглянути все одразу. Виберіть один високоефективний процес для початку як пілотний проект.

– Розробка та моделювання нового робочого процесу: Використовуйте візуальні інструменти вашої BPMS для розробки оптимізованого процесу.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		32

Залучіть власника процесу та кінцевих користувачів до цього етапу, щоб отримати їхній відгук та переконатися, що новий робочий процес є практичним та ефективним.

– Впровадження та тестування: Створіть новий робочий процес у вибраному вами програмному забезпеченні для управління бізнес-процесами. Ретельно протестуйте його на реальних даних, щоб виявити будь-які помилки перед повним розгортанням. Залучення кінцевих користувачів до тестування має вирішальне значення для впровадження.

– Навчіть свою команду та запустіть: Забезпечте комплексне навчання для всіх користувачів. Академія WeWeb пропонує безкоштовні курси та навчальні посібники, щоб команди могли швидко опанувати систему. Переконайтеся, що кожен розуміє свої ролі та як ефективно використовувати нову систему. Як тільки всі будуть готові, запустіть новий процес.

– Моніторинг, аналіз та постійне вдосконалення: впровадження – це не одноразова подія. Використовуйте функції аналітики та звітності вашої BPMS для постійного моніторингу ефективності відповідно до ваших ключових показників ефективності (KPI). Збирайте відгуки та вносьте ітеративні вдосконалення для оптимізації процесу з часом.

Як обрати та досягти успіху з BPMS

Вибір та впровадження програмного забезпечення для управління бізнес-процесами – це стратегічна ініціатива, яка може призвести до значних трансформацій. Успіх залежить від пошуку рішення, яке не лише відповідає вашим технічним вимогам, але й узгоджується з вашими бізнес-цілями та розширює можливості вашої команди. Шукайте платформу, яка пропонує потужний візуальний редактор, безперешкодну інтеграцію та гнучкість масштабування в міру зростання.

Найефективнішими BPM-платформами у 2026 році будуть ті, що впроваджують штучний інтелект, сприяють співпраці та роблять автоматизацію процесів доступною для всіх, а не лише для розробників. Зосередившись на чіткій

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		33

стратегії впровадження, від визначення цілей до постійного вдосконалення, ви можете розблокувати нові рівні ефективності, гнучкості та інновацій.

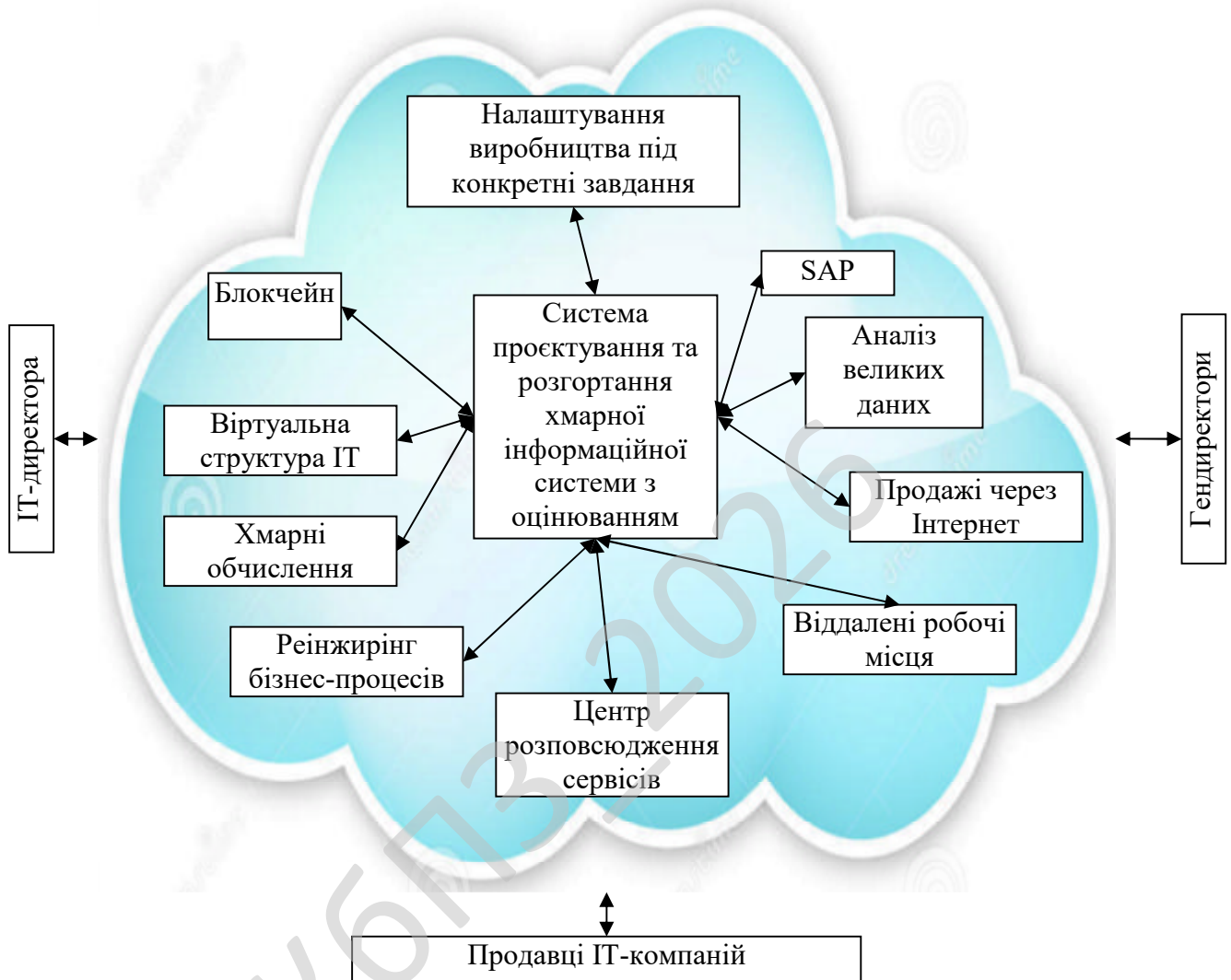


Рисунок 3.1 – Структурна схема системи

Конкретні ІТ-технології, які гендиректори українських компаній, звичайно асоціюють із цифровою трансформацією бізнесу:

- Продажі через Інтернет.
- Омніканальність (робота із замовниками через різні канали).
- Мобільний доступ до корпоративних інформаційних систем.
- Налаштування виробництва під конкретні замовлення.

- Аналіз і прогноз поведження замовників.
- Цифрове проектування й моделювання.
- Соціальні мережі: продажу й робота із претензіями.
- Автоматизація керування логістикою.
- 3D-друк.
- Блокчейн.

ІТ-керівники хочуть і далі продовжувати впровадження нових ІТ-технологій, вважаючи що цифрова трансформація бізнесу – це завдання бізнес-менеджерів по реінжинірингу бізнес-процесів на базі ІТ-технологій (особливо нових).

Цифрова трансформація бізнесу: як її бачать керівники ІТ-служб українських компаній:

- «Хмарні обчислення».
- Віртуальна інфраструктура ІТ.
- Мобільний доступ.
- Вилучені робочі місця.
- SaaS, PaaS, IaaS.
- Аналіз більших даних.
- Омніканальність.
- Реінжиніринг бізнес-процесів.
- Автоматизація служби підтримки клієнтів компанії.
- Майнинг криптовалют.

Продавці хочуть вигідно продати що те зараз, ну або через півроку.

Цифрова трансформація бізнесу: як її бачать продавці ІТ-компаній в Україні:

- «Хмарні обчислення».
- Впровадження продуктів SAP.
- Впровадження продуктів IBM.
- Впровадження продуктів Oracle.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		35

- Аналіз великих даних.
- ЦОД.
- Віртуальна інфраструктура ІТ.
- Інтернет речей.
- Штучний інтелект.
- Shares Service Center.

3.3 Розробка функціональної схеми

Бізнес-процес (Business Process) – установлена послідовність дій, що вимагає певного входу, що досягає певного виходу й використовує певних ресурсів, що служить для реалізації роботи або послуги для клієнта. В англomовній літературі бізнес-процес представляється як безліч із однієї або декількох зв'язаних операцій або процедур, у сукупності які реалізують деяку ціль виробничої діяльності, здійснюваної звичайно в рамках заздалегідь певної організаційної структури, що відбиває відносини між учасниками.

Поняття бізнес-процесу

Поняття одержало поширення у зв'язку з переходом до процесно-орієнтованої організації й процесно-орієнтованому менеджменту підприємства. Характерними для компаній бізнес-процесами є виконання замовлення, розробка продукту, керування компанією, доставка продукції. На практиці в кожній компанії існують типові для їхньої сфери й взаємозалежні один з одним бізнес-процеси, що мають своєю метою створення й реалізацію вартості продуктів і послуг. Обов'язково ознайомтеся зі статтею "Як побудувати бізнес-процес у компанії – інструкція в 4 кроки", щоб зрозуміти, як створюються бізнес-процеси на практиці. Складне стане наочним і зрозумілим.

У відповідності зі стандартом ENISO 9001:2000 процес – це набір взаємозалежних засобів і дій, що перетворюють вхід у результат. Процеси викликають зміни відповідного об'єкта.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		36

У компаніях існують процеси різних видів, які можуть залежати друг від друга й у той же час розрізнятися по багатьом параметрам. Такими параметрами є:

– Вид діяльності: виробничі процеси або надання послуг. Результатом виробничого процесу є матеріальний продукт (наприклад, верстати), надання послуг – нематеріальний (наприклад, інформація).

– Вид виконання: оперативні, тобто виконувані процеси, такі як обробка, розрахунок, або диспозитивні процеси, як планування, керування.

– Створення вартості: прямі, що змінюють об'єкт (монтувати), або непрямі процеси створення вартості, підготовчі, або підтримуючі процеси (перевіряти, транспортувати).

– Комплексність: макропроцеси або мікропроцеси. Макропроцеси описують загальні процеси на підприємствах (виробництво легкових автомобілів). Мікропроцеси – їхні складові частини (виробництво кузова).

– Комерційний успіх: ключові, управлінські й підтримуючі процеси. Ключові процеси є специфічними для компанії й мають особливе значення для досягнення загальної цілі компанії.

Бізнес-процеси часто являють собою комбінацію ключових, управлінських і підтримуючих процесів.

Ключові процеси (створення вартості) поєднують завдання й роботу для виконання певних вимог клієнта із застосуванням ключових виробничих компетенцій. Вони є стратегічно важливими й у той же час специфічними (унікальними, тому що, наприклад, внаслідок застосування фірмових знань їх складно скопіювати). До них ставляться:

- обробка й виконання замовлення;
- розробка, проектування й дизайн продукту;
- виробництво й монтаж і ін.

Управлінські процеси містять у собі завдання й діяльність, спрямовані на довгостроковий розвиток компанії й реалізацію цілей компанії. До них ставляться:

- стратегічний розвиток компанії;
- довго- і середньострокове планування в компанії;
- розвиток персоналу;
- інвестиційне планування;
- мотивація персоналу й ін.

Підтримуючі процеси містять необхідні завдання й роботи для підтримки ключових процесів, але не приводять до безпосередньої цінності для клієнта, наприклад:

- обробка даних;
- технічне обслуговування;
- логістика;
- адміністративні процеси й ін.

На рисунку 3.2 наведена функціональна схема системи.

Формування й структурування припускає розгляд не тільки типології, але й облік рівня процесу.

Для опису процесу з якісно-кількісної, просторово-організаційної й технічно-технологічної точок зору використовуються характеристики (параметри), які задані стандартом ENISO 9001:2000. Параметри процесу – дані для позначення результативності й ефективності, наприклад, витрати, час виконання, якість, точність.



Рисунок 3.2 – Функціональна схема системи

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.



Рисунок 3.3 – Діаграма взаємодії процесів

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 проходить 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).

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.

К6ПЗ_2026

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		41

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.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		42

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 елемента. 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.

Система керування базами даних (СКБД, Database Management System, DBMS) – набір взаємопов'язаних даних (база даних) і програм для доступу до цих даних. Надає можливості створення, збереження, оновлення та пошуку інформації в базах даних з контролем доступу до даних.

Першим поколінням СКБД прийнято вважати ієрархічні й мережеві системи. Ці системи отримали широке поширення в 1970-х роках, а першою комерційною системою цього типу була система IMS компанії IBM.

У 1980-х роках ці системи були витіснені системами другого покоління – повсюдно використовуваними і донині реляційними СКБД.

У цих системах використовувалися непроцедурні мови управління даними (SQL) і передбачався значний ступінь незалежності даних. Реляційні системи внесли значні удосконалення в управління даними: графічний користувацький інтерфейс (GUI), клієнт-серверні застосунки, розподілені бази даних, паралельний пошук даних та інтелектуальний аналіз даних.

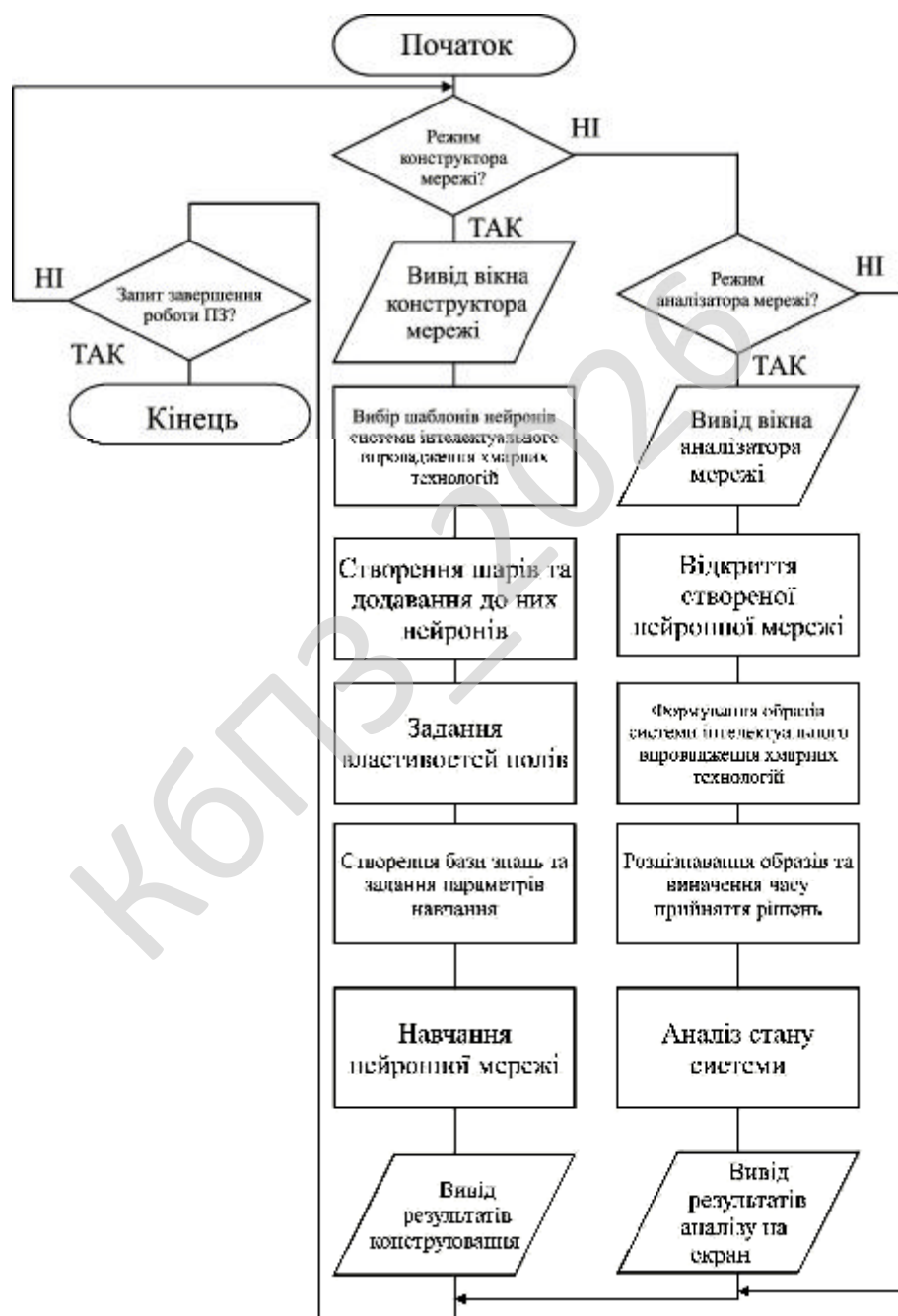


Рисунок 4.1 – Блок-схема основної програми

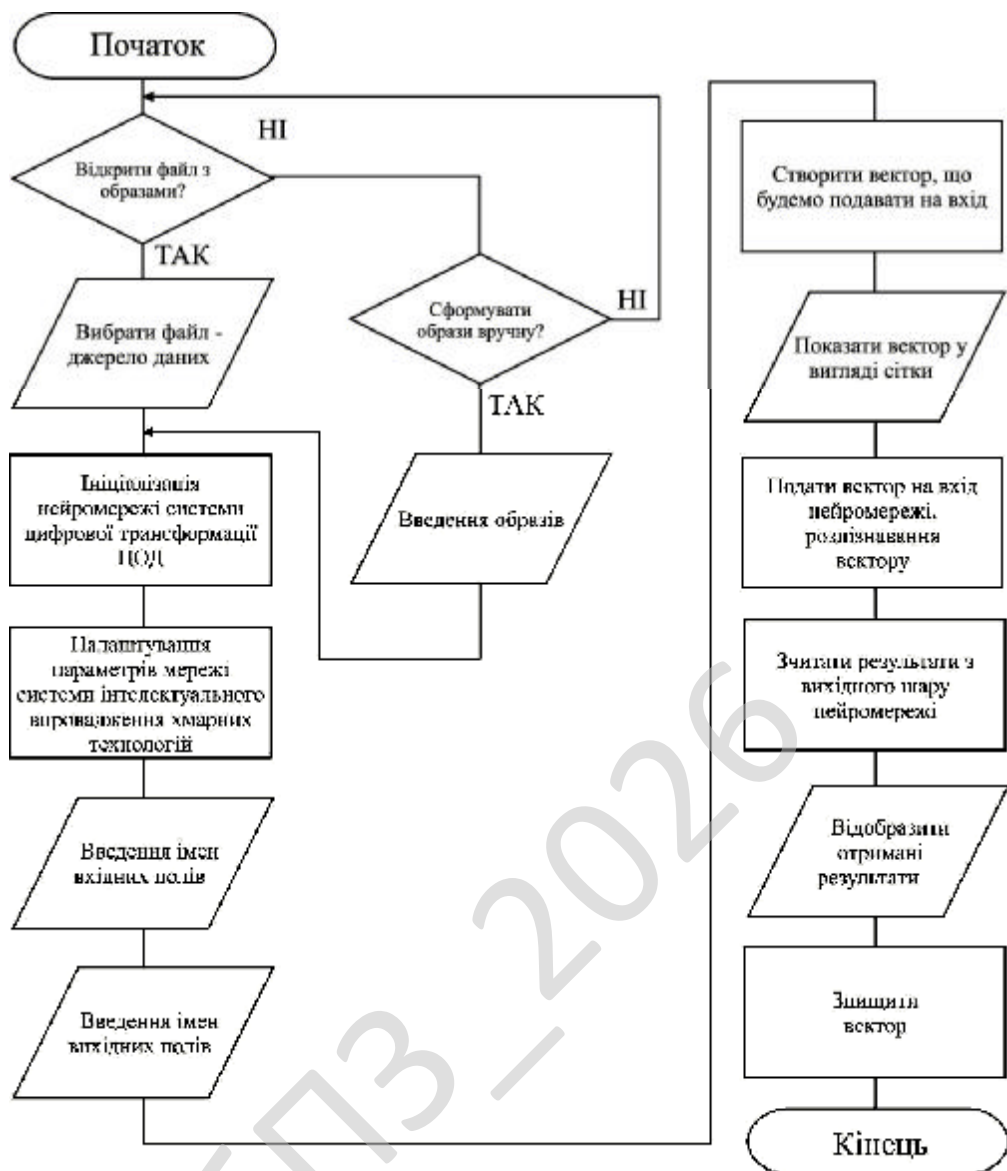


Рисунок 4.2 – Блок-схема роботи підпрограми

Але вже до кінця 1980-х років існуюча тоді реляційна модель перестала задовольняти розробників в силу низки обмежень. Відповіддю на зростаючу складність програм баз даних стали два нових напрямки розвитку СКБД: об'єктно-орієнтовані СКБД і об'єктно-реляційні СКБД.

У 1991 був утворений консорціум ODMG, основною метою якого стало вироблення промислового стандарту об'єктно-орієнтованих баз даних. Між 1993 та 2001 роками ODMG опублікувала п'ять ревізій своїх специфікацій. Остання версія стандарту має індекс 3.0, після чого група розпустилася. До кінця 1990-х

існувало близько десяти компаній, що виробляли комерційні продукти, що позиціонуються на ринку як ООСКБД. Найбільш відомими системами даного класу стали Objectivity, Versant виробництва однойменних компаній, а також СКБД Jasmine, випущена компанією СА. Незважаючи на переваги, що дозволяють ефективніше вирішувати певний ряд завдань, об'єктно-орієнтовані системи так і не змогли завоювати значущу частку ринку СКБД, залишившись «нішевим» продуктом.

Постачальниками традиційних реляційних СКБД також була проведена значна робота з об'єднання об'єктно-орієнтованих і реляційних систем. Розробники постаралися розширити мову SQL, щоб включити в неї концепції об'єктно-орієнтованого підходу, зберігаючи переваги реляційної моделі (об'єктні розширення мови SQL були зафіксовані в стандарті SQL:1999).

Основний принцип – це еволюційний розвиток можливостей СКБД без поломки попередніх підходів та зі збереженням наступності з системами попереднього покоління.

Поняття СКБД третього покоління, якими, власне кажучи, і є об'єктно-реляційні СКБД, з'явилося після опублікування групою відомих фахівців в області баз даних «Маніфесту систем баз даних третього покоління». Основні принципи СКБД третього покоління, позначені в маніфесті:

Крім традиційних послуг з управління даними, СКБД третього покоління повинні забезпечити підтримку розвиненіших структур об'єктів і правил. Розвинутіша структура об'єктів характеризує засоби, необхідні для зберігання і маніпулювання нетрадиційними елементами даних (тексти, просторові дані, мультимедіа).

СКБД третього покоління повинні включити в себе СКБД другого покоління. Системи другого покоління внесли вирішальний вклад у двох областях – непроцедурний доступ за допомогою мови запитів SQL і незалежність даних. Ці досягнення обов'язково повинні враховуватися в системах третього покоління.

СКБД третього покоління повинні бути відкриті для інших підсистем. Це включає оснащення різноманітними інструментами підтримки прийняття рішень, доступом з багатьох мов програмування, інтерфейсами до існуючих популярних систем і бізнес-застосунків, можливістю запуску програм з бази даних на іншій машині і розподілені СКБД. Весь набір інструментів і СКБД має ефективно функціонувати на різноманітних апаратних платформах з різними операційними системами.

Крім того, СКБД, що розраховує на широку сферу застосування, повинна бути оснащена мовою четвертого покоління (4GL).

У середині 1990 років було лише кілька дослідних прототипів СКБД, які поєднали найкращі риси реляційних і об'єктно-орієнтованих СКБД. Першим комерційним продуктом, якому були властиві об'єктно-реляційні риси, став Universal Server компанії Informix(згодом була поглинена IBM). В даний час більшість цих ідей вже втілено в реальних комерційних рішеннях, в тому числі і в продуктах основних постачальників СКБД (Oracle Database і IBM DB2).

Розвиток індустрії систем керування базами даних базується на значних фундаментальних наукових дослідженнях. Найчастіше, між самими дослідженнями та їхньою конкретною реалізацією в прикладних рішеннях минають роки, а іноді й десятиліття. Роботу в області управління даними проводять як університетські дослідницькі групи (MIT, Berkeley), так і центри розробок основних постачальників СКБД (Oracle, IBM, Microsoft). Інвестування в управління даними – це довгострокове, і разом з тим, вигідне вкладення коштів. В даний час дослідники мають у своєму розпорядженні засоби, що дозволяють ефективно реалізувати найскладніші запити, що маніпулюють терабайтами й петабайтами різних даних.

Основними тенденціями, які дали привід для проведення різних масштабних досліджень в області баз даних стали:

Експонентний ріст даних. Обсяг даних, у тому числі синтетичних, що генеруються автоматизованими системами, значно зріс. Збільшилося і число

прикладних областей, в яких вимагається обробка великих обсягів даних. До таких областей тепер відносяться не тільки традиційні корпоративні програми, пошук у веб, але також і наукові дослідження, обробка природних мов, аналіз соціальних мереж тощо.

Значне ускладнення структур використовуваних даних. Прості види даних у вигляді чисел і символьних рядків стали доповнятися численною мультимедійною інформацією, просторовими, процедурними даними та великою кількістю інших складних форматів.

Широке поширення дешевих високопродуктивних апаратних засобів. Щорічно ми спостерігаємо зростання обчислювальних можливостей мікропроцесорів, збільшення ємності і зниження вартості доступних і зручних в експлуатації пристроїв дискової оперативної пам'яті.

Активний розвиток засобів комунікації та «всесвітньої павутини» World Wide Web. WWW стає єдиним інформаційним середовищем, що пронизує весь світ і об'єднує величезне число користувачів та електронних пристроїв.

Поява нових важливих областей застосування СКБД. У першу чергу, це пов'язано з інтелектуальним аналізом даних, сховищами даних, а останнім часом – з паралельними обчисленнями і хмарними технологіями.

Основні характеристики СКБД

1. Контроль за надлишковістю даних.
2. Несуперечливість даних.
3. Підтримка цілісності бази даних (коректність та несуперечливість).
4. Цілісність описується за допомогою обмежень.
5. Незалежність прикладних програм від даних.
6. Спільне використання даних.
7. Підвищений рівень безпеки.

Можливості СКБД

1. Дозволяється створювати БД (здійснюється за допомогою мови визначення даних DDL (Data Definition Language)).

2. Дозволяється додавання, оновлення, видалення та читання інформації з БД (за допомогою мови маніпулювання даними DML, яку часто називають мовою запитів)

3. Можна надавати контрольований доступ до БД за допомогою: Системи забезпечення захисту, яка запобігає несанкціонованому доступу до БД; Системи керування паралельною роботою прикладних програм, яка контролює процеси спільного доступу до БД; Система відновлення – дозволяє відновлювати БД до попереднього несуперечливого стану, що був порушений в результаті збою апаратного або програмного забезпечення.

Основні компоненти середовища СКБД

1. Апаратне забезпечення.
2. Програмне забезпечення.
3. Дані.
4. Процедури – інструкції та правила, які повинні враховуватись при проектуванні та використанні БД.
5. Користувачі: адміністратори даних (керування даними, проектування БД, розробка алгоритмів, процедур) та БД (фізичне проектування, відповідальність за безпеку та цілісність даних); розробники БД; прикладні програмісти; кінцеві користувачі.

Архітектура СКБД

Існує трирівнева система організації СКБД ANSI-SPARC, при якій існує незалежний рівень для ізоляції програми від особливостей представлення даних на нижчому рівні.

Рівні:

1. Зовнішній – представлення БД з точки зору користувача.
2. Концептуальний – узагальнене представлення БД, описує які дані зберігаються в БД і зв'язки між ними. Підтримує зовнішні представлення, підтримується внутрішнім рівнем.
3. Внутрішній – фізичне представлення БД в комп'ютері.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		50

Логічна незалежність – повна захищеність зовнішніх моделей від змін, що вносяться в концептуальну модель.

Фізична незалежність – захищеність концептуальної моделі від змін, які вносяться у внутрішню модель.

Було використано **MongoDB** – документо-орієнтована система керування базами даних (СКБД) з відкритим вихідним кодом, яка не потребує опису схеми таблиць.

MongoDB займає нішу між швидкими і масштабованими системами, що оперують даними у форматі ключ/значення, і реляційними СКБД, функціональними і зручними у формуванні запитів.

Код MongoDB написаний на мові C++ і поширюється в рамках ліцензії AGPLv3.

MongoDB підтримує зберігання документів в JSON-подібному форматі, має досить гнучку мову для формування запитів, може створювати індекси для різних збережених атрибутів, ефективно забезпечує зберігання великих бінарних об'єктів, підтримує журналювання операцій зі зміни і додавання даних в БД, може працювати відповідно до парадигми Map/Reduce, підтримує реплікацію і побудову відмовостійких конфігурацій.

У MongoDB є вбудовані засоби із забезпечення шардінгу (розподіл набору даних по серверах на основі певного ключа), комбінуючи який з реплікацією даних можна побудувати горизонтально масштабований кластер зберігання, в якому відсутня єдина точка відмови (збій будь-якого вузла не позначається на роботі БД), підтримується автоматичне відновлення після збою і перенесення навантаження з вузла, який вийшов з ладу.

Розширення кластера або перетворення одного сервера на кластер проводиться без зупинки роботи БД простим додаванням нових машин.

При розробці автори виходили з необхідності спеціалізації баз даних, завдяки чому їм вдалося відійти від принципу «один розмір під усе». За рахунок мінімізації семантики для роботи з транзакціями з'являється можливість

вирішення цілого ряду проблем, пов'язаних з нестачею продуктивності, причому горизонтальне масштабування стає простішим. Використовувана модель документів зберігання даних (JSON/BSON) простіше кодується, простіше управляється (у тому числі за рахунок застосування так званого безсхемного стилю (schemaless style), а внутрішнє угруповання релевантних даних забезпечує додатковий вииграш в швидкодії.

Нереляційний підхід досить зручний для створення баз даних, у яких горизонтальне масштабування означає розгортання на множині машин. Можливість забезпечувати найкращу продуктивність повинна існувати паралельно з підтримкою більшої функціональності, ніж це дозволяє використання пар «ключ-значення» (у чистому вигляді).

Технологія баз даних має працювати скрізь, починаючи з серверів користувача та віртуальних машин і закінчуючи хмарними технологіями.

MongoDB, на думку розробників, має заповнити розрив між простими сховищами даних типу «ключ-значення» (швидкими і легко масштабованими) і великими СКБД (зі структурними схемами і потужними запитами).

Основні можливості MongoDB:

1. Документо-орієнтоване сховище (проста та потужна JSON-подібна схема даних).
2. Досить гнучка мова для формування запитів.
3. Динамічні запити.
4. Повна підтримка індексів.
5. Профілювання запитів.
7. Швидкі оновлення «на місці».
8. Ефективне зберігання бінарних даних великих обсягів, наприклад, фото та відео.
9. Журналювання операцій, що модифікують дані в БД.
10. Підтримка відмовостійкості і масштабованості: асинхронна реплікація, набір реплік і шардінг.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		52

11. Може працювати відповідно до парадигми MapReduce.

СКБД управляє наборами JSON-подібних документів, що зберігаються в бінарному форматі BSON. Зберігання і пошук файлів в MongoDB відбувається завдяки викликам протоколу GridFS.

Подібно до інших документо-орієнтованих СКБД (CouchDB тощо), MongoDB не є реляційною СКБД.

Є докладна і якісна документація, велике число прикладів і драйверів для популярних мов Java, C++, C#, PHP, Python, Perl, Ruby.

При випуску одразу було заявлено, що реліз MongoDB 1.0 готовий до використання у виробництві як одиничний хост, так і у зв'язках master/slave.

Код цього релізу досить стабільний і перевірений в промисловій експлуатації протягом 1,5 років. MongoDB рекомендується розгортати мінімум на двох серверах використовуючи реплікацію Master/Slave.

Це забезпечує наявність актуальних даних при виході з ладу однієї з СКБД. MongoDB – продукт досить молодий, і відтак у ньому зустрічаються помилки, з'являються нові можливості тощо. Характерний високий темп розробки (проект пишуть не тільки волонтери, а й компанія людей на повній зайнятості). Компанія-розробник надає платні підтримку, хостинг, консультації.

Приклади запитів

Запити можуть витягати дані з вбудованих об'єктів та масивів. Якщо в колекцію users вставлений такий об'єкт:

```
{
  "username" : "bob",
  "address" : {
    "street" : "123 Main Street",
    "city" : "Springfield",
    "state" : "NY"
  }
}
```

Ми можемо запитати цей документ (і всі документи з адресою в Нью-Йорку) за допомогою:

```
> db.users.find({"address.state" : "NY"})
```

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		53

Можна також запитати елементи масиву:

```
> db.food.insert({"fruit" : ["peach", "plum", "pear"]})  
> db.food.find({"fruit" : "pear"})
```

Jira – була використана комерційна система відслідковування помилок, призначена для організації взаємодії з користувачами, хоча в деяких випадках використовується і для управління проектами. Розроблено компанією Atlassian, є одним з двох її основних продуктів (поряд з вікі-системою Confluence). Має веб-інтерфейс.

Назва системи отримано шляхом усічення слова «Gojira» – Японського імені монстра Годзилла, що, в свою чергу, є відсиланням до назви конкуруючого продукту – Bugzilla; створювалася в якості заміни Bugzilla і багато в чому повторює її архітектуру. Система дозволяє працювати з декількома проектами. Для кожного з проектів створює і веде схеми безпеки і схеми оповіщення.

До версії 3.13.5 (включно) розрізнялися редакції Enterprise, Professional і Standard, після – Залишилася тільки редакція Enterprise (для великих організацій).

Система заснована на Java EE і працює на кількох популярних системах управління базами даних і операційних системах.

Основний елемент обліку в системі – завдання (ticket або issue). Завдання містить назву проекту, тему, тип, пріоритет, компоненти і зміст. Завдання може бути розширена додатковими полями (також і нові призначені для користувача поля можуть бути визначені), додатками (наприклад – Фотографіями, скріншотами) або коментарями. Завдання може редагуватися або просто змінювати статус, наприклад, з «відкритий» в «закритий». Які переходи між станами можливі, визначається через настраюється потік операцій. Будь-які зміни в задачі записуються в журнал.

Jira має велику кількість можливостей конфігурації: для кожної програми може бути визначений окремий тип завдання з власним workflow, набором статусів, одним або декількома видами уявлення (screens). Крім того, за допомогою так званих «схем» можна визначити для кожного індивідуального Jira-проекту власні права доступу, поведінку і видимість полів і багато іншого.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		54

Завдяки універсальному підходу можна пристосувати Jira для багатьох непрофільних завдань, наприклад, керування вимогами, керування ризиками, аж до реалізації невеликої системи бронювання, автоматизації процесу рекрутингу.

Для інтеграції з зовнішніми системами підтримує інтерфейси SOAP, XML-RPC і REST. Поставляється із засобами інтеграції з такими системами управління версіями, як Subversion, CVS, Git, Clearcase, Team Foundation Server, Mercurial і Perforce.

Існують доповнення, що дозволяють вбудувати Jira в інтегровані середовища розробки, в тому числі Eclipse і IntelliJ IDEA. Перекладена багатьма мовами, включаючи російську, англійську, японську, німецьку, французьку, іспанську.

Для сторонніх розробників надаються кошти розробки розширень системи – плагінів. Розробники розширень можуть викладати плагіни для продажу на спеціальний розділ сайту Atlassian.

Є комерційним продуктом, який може бути ліцензований для роботи на локальному сервері або доступний в якості віддаленого додатки. Ціноутворення залежить від максимального числа користувачів, при цьому близько \$50 за користувача для локального і \$7 на місяць за користувача для віддаленого доступу є типовими цінами.

Для академічних і комерційних клієнтів доступний повний вихідний код під ліцензією розробника.

Для проектів з відкритим вихідним кодом Atlassian надає спеціальну безкоштовну ліцензію при дотриманні наступних правил:

- проект використовує ліцензії, схвалені Open Source Initiative;
- вихідний код проекту доступний для скачування;
- у проекту є публічно доступна веб-сайт;
- програмне забезпечення від Atlassian є на веб-сайті проекту.

Хоча я реалізовував програму сам, було використано підходи Scrum для саморозвитку та пришвидшенню розробки, розглянемо цей метод. Scrum – підхід

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		55

управління проектами для гнучкої розробки програмного забезпечення. Скрам чітко робить акцент на якісному контролі процесу розробки.

Підхід вперше описали Гіротакі 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 ДеҐрейс 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.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		56

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?

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		59

– 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.

Розглянемо визначення API. Це прикладний програмний інтерфейс (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.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		60

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 вилучення 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.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		61

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, ППП 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.

4.2 Захист розробленого програмного забезпечення

Захист розробленого програмного забезпечення буде відбуватися за допомогою RC6 – симетричний блоковий криптографічний алгоритм, похідний від алгоритму RC5. Був створений Роном Рівестом, Меттом Робшау і Реєм Сіднеєм для задоволення вимог конкурсу Advanced Encryption Standard (AES). Алгоритм був одним з п'яти фіналістів конкурсу, був також представлений NESSIE і CRYPTREC. Є власницьким (пропрієтарним) алгоритмом, і

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		63

запатентований RSA Security, однаке дія патентів сплила, і зараз алгоритм знаходиться у відкритому доступі. В той же час, "RC6" залишається зареєстрованою торговою маркою RSA.

Варіант шифру RC6, заявлений на конкурс AES, підтримує блоки довжиною 128 біт і ключі довжиною 128, 192 і 256 біт, але сам алгоритм, як і RC5, може бути налаштований для підтримки більш широкого діапазону довжин як блоків, так і ключів (від 0 до 2040 біт)^[1]. RC6 дуже схожий на RC5 за своєю структурою і також досить простий у реалізації.

Є фіналістом AES, проте одна з примітивних операцій – операція множення, повільно виконується на певному обладнанні і ускладнює реалізацію шифру на ряді апаратних платформ і, що виявилось сюрпризом для авторів, на системах з архітектурою Intel IA-64 також реалізована досить погано. В даному випадку алгоритм втрачає одну зі своїх ключових переваг – високу швидкість виконання, що стало причиною для критики і однією з перепон для обрання як нового стандарту.

Деталі RC6

Так само, як і RC5, RC6 – повністю параметризована сім'я алгоритмів шифрування. Для специфікації алгоритму з конкретними параметрами, прийнято позначення RC6-w/r/b, де

- W – довжина машинного слова в бітах.
- R – число раундів.
- B – довжина ключа в байтах. Можливі значення 0 .. 255 байт.

Для того щоб відповідати вимогам AES, блочний шифр повинен працювати з 128-бітовими блоками. Так як RC5 – виключно швидкий блочний шифр, розширення його, щоб працювати з 128-бітовими блоками, привело б до використання двох 64-бітових робочих регістрів. Але архітектура і мови програмування ще не підтримують 64-бітні операції, тому довелося змінити проект так, щоб використовувати чотири 32-бітних регістри замість двох 64-бітних.

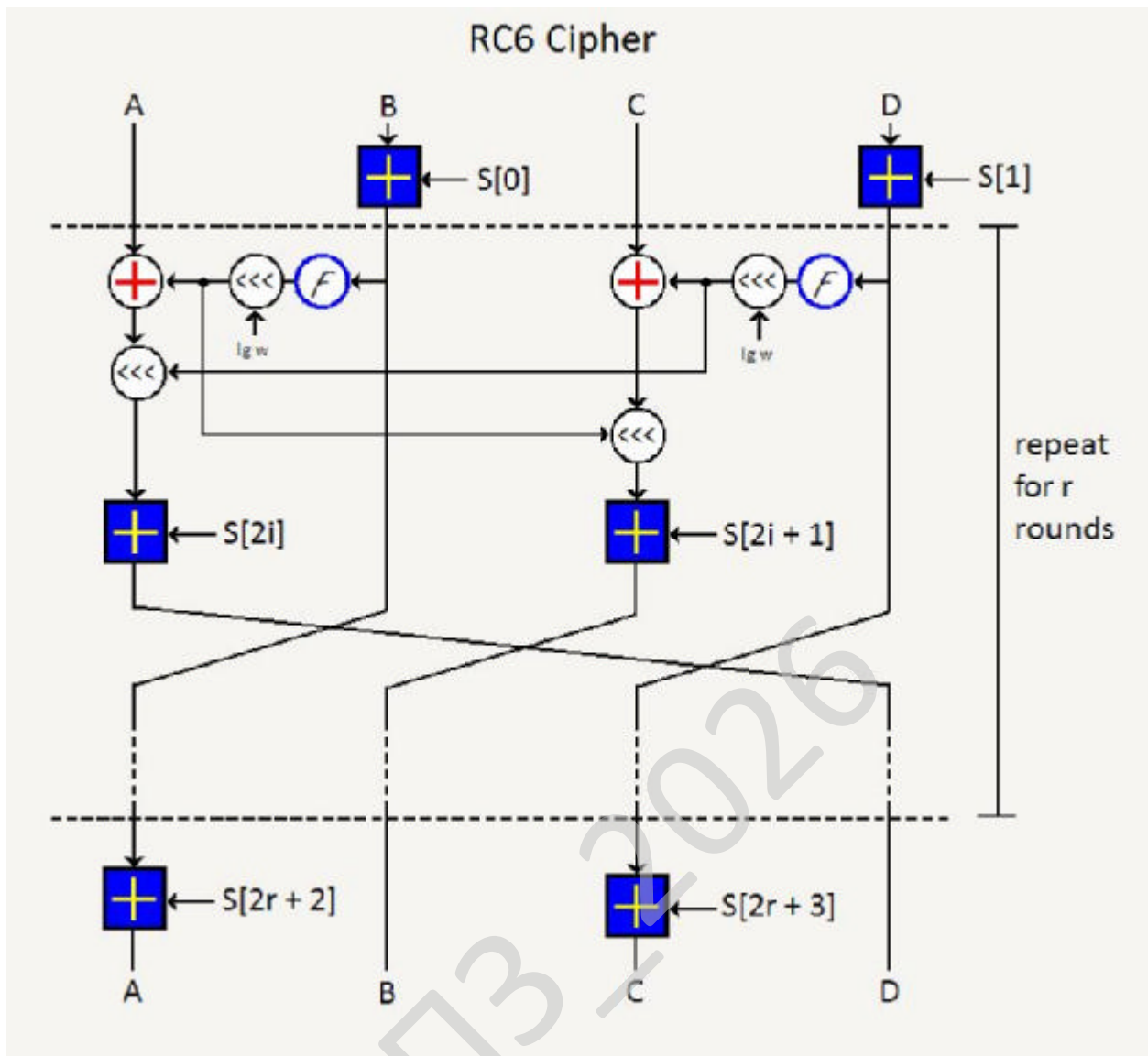


Рисунок 4.3 – Структура RC6

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

На рисунку 5.1 зображено розроблене у бакалаврської дипломної роботі системі проектування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості. З рисунку можна побачити що інтерфейс головного вікна розподілено на наступні функціональні розділи: Розділу виведення результату роботи системи; Функціональних кнопок ПЗ; Навігаційного меню; Верхнього меню.

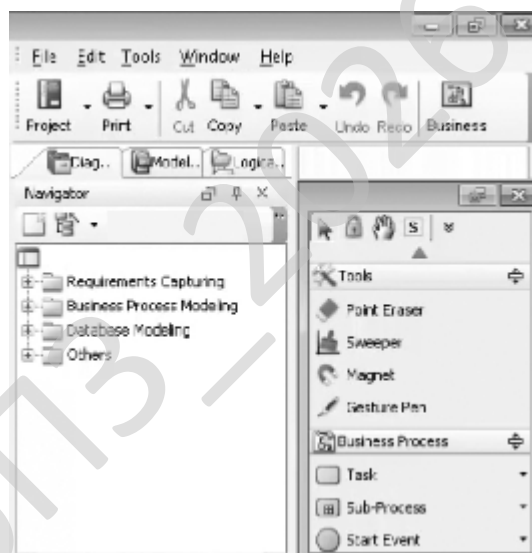


Рисунок 5.1 – Головне вікно ПЗ

Інфраструктурні хмарні сервіси, керовані сервіси в області інфраструктури й експертиза по побудові приватних хмар були об'єднані в підрозділі «Хмарні Сервіси». Для продуктового бізнесу характерні інші правила гри й процеси, ніж ті, що властиві проектній діяльності. Якщо по завершенні проекту спілкування із замовником зводиться до мінімуму, то при наданні сервісів із замовником доводиться спілкуватися щодня.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		66

Виділення в окремий бізнес дало крім того більшу свободу дій і дозволило прискорити прийняття рішень. Початий редизайн внутрішніх процесів дозволив оптимізувати роботу й, як затверджується, «позитивно позначився на цінovій політиці».

Хмарні сервіси дозволяють залучати нових замовників, з ким до того ніколи не працював – по внутрішній статистиці 70% це саме нові замовники. При цьому більше 80% замовників розміщують у хмарі бізнес-критичні сервіси. Для забезпечення необхідного ступеня надійності хмара розміщується у власному ЦОД, сертифікованому на відповідність вимогам і Tier 3. 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.

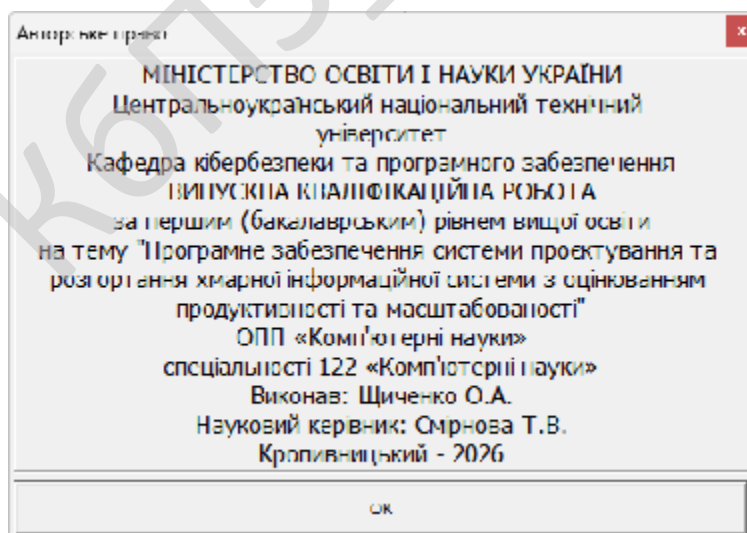


Рисунок 5.2 – Авторське право

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		67

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.
- Оновлення.
- Вмонтування.
- Відстежування версій.
- Видалення.
- Вилучення 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 ІТ рішення для автоматизації операцій усередині саме цих

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		68

процедур. Таким чином, розроблене ІТ 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 ІТ 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.

Під час роботи над програмою було проведено тестування програмного забезпечення, тобто технічне дослідження, призначене для виявлення інформації про якість продукту відносно контексту, в якому воно має використовуватись.

Тестування включає як процес пошуку помилок або інших дефектів, так і випробування програмних складових з метою їх оцінки.

Проводилась оцінка:

- відповідності поставленим вимогам;
- правильна відповідь для усіх можливих вхідних даних;
- виконання функцій за прийнятний час;
- практичність;
- сумісність з ОС та стороннім ПЗ.

Оскільки число можливих тестів для програмних компонент практично нескінченне, тому стратегія тестування полягала в тому, щоб провести всі можливі тести з урахуванням наявного часу та ресурсів.

Як результат ПЗ тестувалось стандартним виконанням програми з метою виявлення помилок або інших дефектів.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		69

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», чію 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 ІТс).
- Набір, утворений такими вхідними даними, які демонструють дефекти програми (назвемо його ОТ).

Будь-який спосіб тестування «чорної скриньки» повинен:

- Виявити такі вхідні дані, які з високою ймовірністю належать набору ІТс;
- Сформулювати такі очікувані результати, які з високою ймовірністю є елементами набору ОТ.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		70

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.
- Помилки інтерфейсу.
- Помилки у зовнішніх структурах даних або в доступі до зовнішньої бази даних.
- Помилки характеристик (необхідна ємність пам'яті і т.д.).
- Помилки ініціалізації та завершення.

Обрано умови розповсюдження – Freeware. Це власницьке програмне забезпечення, котре можна Безоплатно використовувати протягом необмеженого терміну без обмежень у функціональності, і поширюване без сирцевих кодів.

Автори такого програмного забезпечення, як правило, хочуть «дати щось спільноті», але хочуть також контролювати його подальшу розробку. Іноді, коли програмісти вирішують припинити розробку, вони передають сирцевий код іншим програмістам, або ж спільноті як вільне програмне забезпечення.

Дуже часто плутають поняття «безплатне програмне забезпечення» та «вільне програмне забезпечення», хоча вони суттєво відрізняються.

Безплатне програмне забезпечення можна безоплатно встановлювати та використовувати (іноді з певними обмеженнями, як, наприклад, «безплатне для домашнього або некомерційного вжитку»), в той час як вільне програмне забезпечення можна продавати за будь-яку суму, але при тому, у користувача, котрий його отримує, повинні бути права на вивчення, модифікацію та поширення сирцевих кодів одержаної програми.

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти, призначено для системи проєктування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості.

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

Рішення завдання полягало у вирішенні наступних задач:

- Був проведений огляд існуючих систем проєктування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості.

- Досліджена система проєктування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості.

- На основі отриманих результатів досліджень створена програмна реалізація системи проєктування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості.

Розроблені під час виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання проєктування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості.

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

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		72

Програма реалізована на мові високого рівня Python. Дана мова програмування дозволяє найбільш ефективно обробляти дані призначені для системи проєктування та розгортання хмарної інформаційної системи з оцінюванням продуктивності та масштабованості. 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 RC6.

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.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		73

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Russ White & Ethan Banks «Computer Networking Problems and Solutions: An Innovative Approach to Building Resilient, Modern Networks». 2017. – 832 p.
2. Вінтенко Б., Смірнов О., Миронець І., Смірнова Т., Смірнов С. «Імітаційна модель шляхів вхідних даних комп'ютерної інтелектуальної системи підтримки оператора енергоблоку АЕС». *Комбінаторні конфігурації та їхні застосування: Матеріали XXVII Міжнародного науково-практичного семінару, присвяченого 125-річчю Національного університету «Запорізька політехніка»* (Запоріжжя-Кропивницький-Київ, 4-6 червня 2025 р.). Запоріжжя: НУ «Запорізька політехніка», 2025. С.82-91.
3. Al-Azzeh, J., Ayyoub, B., Mesleh, A., Smirnova, T., Gnatyuk, S., Drieiev, O., Smirnov, O., Dorenskyi, O. «Cloud-Based Information System for Evaluating Caverns in the Process of Blasting Metal Surfaces of Details». *International Review on Modelling and Simulations* 18 (1), 2025. pp. 32-42.
4. Смірнова Т.В., Коноплицька-Слободенюк О.К., Буравченко К.О., Смірнов С.А., Кравчук О.В., Козірова Н.Л., Смірнов О.А. «Дослідження технологій забезпечення кібербезпеки хмарних сервісів IaaS, PaaS та SaaS». *Кібербезпека: освіта, наука, техніка*. 2024. №4(24), С. 6-27.
5. Батрак О., Смірнова Т., Гнатюк В., Одарченко Р., Смірнов О. «Дослідження показників ефективності функціонування та перспектив розвитку систем IP-телефонії». *Підводні технології*, 2024, № 13, с. 28-35.
6. Kuznetsov, O., Kryvinska, N., Ilchenko, O., Smirnova, T., Ulianovska, Y. «Comparative Analysis of Cryptocurrency Trading Platforms Using the Analytic Hierarchy Process». *CEUR Workshop Proceedings*, 2023, 3628, pp. 106-115.
7. 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.

8. Smirnov, O., Sydorenko, V., Aleksander, M., Zhyharevych, O., Yenchov, S. «Simulation of the cloud IoT-based monitoring system for critical infrastructures». *CEUR Workshop Proceedings*, Volume 3530, 2023, pp. 256-265.

9. 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.

10. 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.

11. 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.

12. 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.

13. 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.

14. 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

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		75

захисту в інформаційно-комунікаційних системах» *Вісник Хмельницького національного університету. Серія: «Технічні науки», № 2 (307). С. 46-52. 2022.*

15. 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.

16. 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.

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

18. 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.

19. 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.

20. 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.

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		76

21. 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.
22. 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.
23. 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.
24. Smirnov O. Kuznetsov A., Zaichenko 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.
25. 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.
26. 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.
27. 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.

28. 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.

29. 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.

30. 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.

31. 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.

32. 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.

33. Smirnov A.A., Kuznetsov 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.

34. 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

35. 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.

36. 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.

37. Смірнов О.А., Коноплицька-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.

38. 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.

39. Смірнов О.А., Дреєва Г.М., Дреєв 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.

40. Смірнов О.А., Коноплицька-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.

41. 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).

42. Смірнов О.А., Дреєва 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

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		79

43. Дреєва Г.М., Смірнов О.А., Дреєв О.М. Метод генерування фрактальноподібної числової послідовності на основі скінченного автомату для моделювання трафіку у мережі. Центральнотраїнський науковий вїсник. Технїчні науки. № 1(32). с. 173-183, 2019.

44. Смірнова Т.В., Солових Є.К., Смірнов О.А., Дреєв О.М. Побудова хмарних інформаційних технологій оптимізації технологічного процесу відновлення та зміцнення поверхонь деталей. Центральнотраїнський науковий вїсник. Технїчні науки. № 1(32). с. 184-194, 2019.

45. Смірнов О.А., Смірнов С.А., Полїщук Л.І., Смірнова Т.В., Коноплицька-Слободенюк О.К. Метод формування антивірусного захисту даних з використанням безпечної маршрутизації метаданих. Кїбербезпека: освіта, наука, технїка. – Том 3 № 3. – Кїїв: КУ ім. Бориса Грінченка. – 2019. – С. 63-87.

46. 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 посїбник – Кропивницький: вид. Лисенко В.Ф. 2018. – 177 с.

47. Смірнов О.А., Котелянець В.В. Стїйкї до колїзїй стохастичнї моделї функціонування безпроводових сенсорних мереж. Вїсник інженерної академїї України, №3, с. 145-152, 2018

48. Смірнов О.А., Смірнов С.А., Дїдик А.К., Дреєв А.М. Алгоритми формування безлїчї маршрутів передачі метаданих у антивіруснї хмарнї системи. Збірник наукових праць "Системи обробки інформації". - Випуск 5 (142). - Х.: ХУПС - 2016. - С. 148-152.

49. Смірнов О.А., Смірнов С.А. Дїдик А.К., Дреєв О.М. Моделї системи нейромережевих експертів безпечної маршрутизації у хмарних антивірусних системах. Збірник наукових праць "Системи обробки інформації". - Випуск 3 (140). - Х.: ХУПС - 2016. - С. 36-39.

50. Смірнов О.А., Смірнов С.А., Дїдик А.К., Дреєв А.М. Спосїб контролю лїній зв'язку телекомунікаційної системи антивірусу. Збірник наукових

праць Харківського університету Повітряних Сил. Випуск 2 (47). – Харків: ХУПС. - 2016. - С. 121-127.

51. 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.

52. 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.

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

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

					ВКРБ-122.26.0035.00.00.ПЗ	Арк.
Вим.	Арк.	№ докум.	Підпис	Дата		81