Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра інформатики факультету інформатики

АНАЛІЗ МЕТОДІВ РЕІНЖИНІРИНГУ ТА ОПТИМІЗАЦІЇ ПРОГРАМНИХ СИСТЕМ

Текстова частина до курсової роботи за спеціальністю "Комп'ютерні науки" 122

Керівник курсової роботи
кандидат технічних наук,
доцент Савченко Т.В.

_____
(підпис)
"____" _____ 2025 р.

Виконала студентка КН-3
Романюк Н.Р.

_____
(підпис)
"____" _____ 2025 р.

Київ 2025

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ
Зав. кафедри інформатики,
кандидат наук, доцент
Гороховський С.С.
_____
(підпис)
„____”_____2025 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на курсову роботу

студентці 3 року навчання БП “Комп’ютерні науки”
Романюк Надії Ростиславівні
на тему:
Аналіз методів реінжинірингу та оптимізації програмних систем

Зміст ТЧ до курсової роботи:

Індивідуальне завдання

1. Перелік використаних скорочень
2. Анотація
3. Abstract
4. Вступ
5. Дослідження реінжинірингу бізнес-процесів
6. Реінжиніринг та оптимізація програмних систем
7. Практичне застосування методів реінжинірингу бізнес-процесів
8. Висновки
9. Список використаних джерел

Дата видачі „ 5 ” жовтня 2024 р.

Керівник Савченко Т.В. _____
(підпис)
Завдання отримала _____

(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапу курсової роботи	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу.	05.10.2024	
2.	Огляд літератури за темою роботи.	15.11.2024	
3.	Написання текстової частини роботи.	15.02.2025	
4.	Створення практичної частини роботи	07.04.2025	
5.	Корегування роботи згідно з результатами перевірки роботи науковим керівником.	20.04.2025	
6.	Створення презентації.	25.04.2025	
7.	Остаточне оформлення презентації.	01.05.2025	
8.	Подання роботи на кафедру для перевірки на плагіат.	08.05.2025	
9.	Захист курсової роботи.	16.05.2025	

Студент _____

Керівник _____

"_____" _____

Зміст

ПЕРЕЛІК ВИКОРИСТАНИХ СКОРОЧЕНЬ

IEEE – Інститут електротехніки та електронної техніки

IDE – інтегроване середовище розробки

ESB – корпоративна сервісна шина

MVP – мінімально життєздатний продукт

CDN – мережа доправлення контенту

BPMN – модель та нотація бізнес-процесів

EPC – ланцюжок керованих подіями процесів

UML – уніфікована мова моделювання

BPR – реінжиніринг бізнес-процесів

BPML – мова моделювання бізнес-процесів

BPQL – мова запитів бізнес-процесів

BPEL – мова виконання бізнес-процесів

OKR – цілі та ключові результати

АНОТАЦІЯ

У даній роботі аналізуються методи, засоби та технології реінжинірингу та оптимізації програмних систем, створюється класифікація моделей бізнес-процесів, порівняльний аналіз підходів та методологій реінжинірингу, а також розглядаються цілі, передумови та переваги. Аналізується бізнес-процес обраного бізнесу, проводиться редизайн бізнес-процесу, створюються план реінжинірингу бізнес-процесу згідно з обраною методологією та проводиться аналіз ризиків процесу реінжинірингу.

ABSTRACT

This paper analyzes the methods, tools and technologies of reengineering and optimization of software systems, creates a classification of business process models, a comparative analysis of reengineering approaches and methodologies, and also considers goals, prerequisites and advantages. The business process of the selected business is analyzed, the business process is redesigned, a business process reengineering plan is created according to the chosen methodology and a risk analysis of the reengineering process is being conducted.

ВСТУП

У сучасному світі інтенсивний розвиток інформаційних технологій та складність програмних систем стають критично важливими для функціонування бізнесу, державних структур та промисловості. Зростання обсягу, складності та вимог до програмних систем обумовлюють необхідність їх вдосконалення. При цьому доцільно використовувати підходи до реінжинірингу та оптимізації програмних систем, що дозволяють переосмислити, перепроектувати та модернізувати існуючі програмні рішення без повної їх заміни.

Актуальність обраної теми. Застарілі процеси та системи часто не відповідають сучасним вимогам і суттєво зменшують конкурентоспроможність, ефективність та збільшують час виконання процесів. Наслідком стає втрата прибутку. Саме тому виникає необхідність проведення реінжинірингу, який слугує інструментом адаптації до актуальних вимог. При цьому, реінжиніринг є ефективним лише за умови розуміння сфери його застосування та методів, інструментів та засобів, які є ефективними у тій чи іншій ситуації. В інших випадках проведення реінжинірингу може бути безрезультатним та збитковим.

Було поставлено такі завдання:

- Проаналізувати технології, методи та засоби реінжинірингу бізнес-процесів;
- Проаналізувати технології, методи та засоби реінжинірингу програмних систем;
- Розглянути цілі, передумови та переваги реінжинірингу;
- Створити порівняльний аналіз підходів до реінжинірингу програмних систем;
- Створити порівняльний аналіз методологій реінжинірингу бізнес-процесів;
- Розглянути моделі бізнес-процесів та створити їх класифікацію;
- Дослідити методи оптимізації програмних систем;

- Обрати один з ключових бізнес-процесів у медичному центрі «Дитина», проаналізувати його та здійснити редизайн бізнес-процесу;
- Проаналізувати дотичні елементи веб-сайту на доступність та адаптивність та здійснити їх редизайн;
- Обрати методологію та створити план реінжинірингу.
- Ідентифікувати, проаналізувати можливі ризики реінжинірингу бізнес-процесу та запропонувати заходи оптимізації.

Об’єкт дослідження – реінжиніринг та оптимізація програмних систем та бізнес-процесів.

Предмет дослідження – слабкі сторони та можливі покращення бізнес-процесу запису на прийом у Медичному центрі “Дитина”, доступність та адаптивність дотичних елементів веб-сайту.

Метою курсової роботи є дослідження поняття реінжинірингу та оптимізації, а також застосування отриманих знань для здійснення аналізу, редизайну бізнес-процесу, обрання методології та формування плану реінжинірингу, а також ідентифікація, аналіз можливих ризиків та формування заходів оптимізації.

Наукова новизна роботи полягає не у створенні принципово нових методів реінжинірингу, а в практичному застосуванні існуючих підходів для оптимізації конкретного бізнес-процесу в умовах реальної організації. Робота демонструє доцільність чи недоцільність використання відомих методологій для підвищення якості обслуговування та доступності сервісів, пропонує альтернативи.

Практична цінність полягає у розробці конкретного плану реінжинірингу бізнес-процесу із урахуванням сучасних вимог до доступності та адаптивності веб-сервісів, а також здійсненні редизайну бізнес-процесу та окремих елементів. Запропоновані рішення можуть бути впроваджені для підвищення ефективності та якості надання послуг.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ РЕІНЖИНІРИНГУ БІЗНЕС – ПРОЦЕСІВ

1.1 Реінжиніринг бізнес-процесів. Порівняльний аналіз методів

Реінжиніринг бізнес-процесів не є новою концепцією. Дослідники вважають, що це поняття сформувалось на основі теорій менеджменту 19 століття та школи стратегічного мислення двадцятого століття. Визначення поняття також варіюється: воно розглядається як радикальні зміни до наявних процесів або ж як поступові функціонально-орієнтовані модифікації. Не зважаючи на велику кількість дискусій, обидві інтерпретації мають місце бути та повинні слугувати основою для вибору методу реінжинірингу, спираючись на індивідуальні вимоги. Основні цілі реінжинірингу визначаються незалежно від визначення. Метод має на меті покращення продуктивності існуючих процесів та зменшення витрат. Цей процес часто включає реінжиніринг комплементарних програмних систем [1].

Прикладом успішного реінжинірингу можна вважати компанію General Electrics Aircraft Engines [2]. Метою компанії було зменшення грошових та часових витрат і підвищення якості продукції. Компанія здійснила такі зміни як:

- формування крос-функціональних команд на ранніх етапах розробки,
- впровадження засобів автоматичного проектування та моделювання, стандартизація та модульний підхід до виробництва компонентів,
- оптимізація відносин з постачальниками.

Таким чином, кількість дефектів продукції зменшилась на 50%, а час виготовлення – на 60%/

Можна визначити кілька універсальних етапів реінжинірингу бізнес-процесів [3]. Першим етапом є визначення цілей та концепції бізнесу. Мають бути сформульовані очікування, конкуренція, можливості та бачення майбутнього бізнесу. Наступним етапом є аналіз наявних процесів чи системи в цілому. Одним із інструментів є моделювання, що буде досліджено у підрозділі 1.4. Далі відбувається редизайн процесів, що цього потребують, а саме: неефективні, багатозатратні чи навпаки, мають потенціал до збільшення

продуктивності. На цьому етапі також інструментом є моделювання, як і на заключному етапі, що полягає в імплементації модифікованих процесів.

Зважаючи на унікальність кожного бізнесу вимог до реінжинірингу, сформувалися різні методи впровадження. Було виокремлено та розглянуто основні методології [4] та створено власну порівняльну характеристику (табл. 1.1).

Методологія Шість сігм спирається на 5 кроків, а саме: визначення цілей, вимір поточного процесу, аналіз взаємовідносин, вдосконалення та оптимізація, контроль та окремим кроком виділяють впровадження. Це робить її комплексною та універсальною,

Методологія Давенпорта описує процеси прототипування, проте оминає кроки імплементації та оцінки, робить акцент на зв’язку з технологіями, а також обмежує редизайн процесів до 15 за одну ітерацію. Таким чином вона може розглядатися лише як комплементарна при реінжинірингу.

Методологія Гаммера та Чемпі складається з чотирьох етапів: мобілізації, діагностики, редизайну та реалізації. Унікальним етапом є мобілізація, що передбачає обґрунтування та пояснення змін задіяним особам і ефективно створює підґрунтя для сприйняття реінжинірингу.

Методологія точки зупинки є універсальною та включає етапи відкриття, перепроектування та реалізації. Кожен з цих ключових етапів поділений на модулі, що в свою чергу містять завдання. Методологія спрямована на ефективне використання ресурсів та управління змінами.

Швидкий реінжиніринг фокусується на процесах, що є визначальними для досягнення визначених цілей. Включає такі етапи як формулювання цілей, вибір сфер оптимізації, аналіз наявних процесів та пошук тих, що потребують оптимізації, створення необхідної інфраструктури та впровадження процесів. Очевидним недоліком є розмитість перших трьох етапів, що складають враження непов’язаності між собою.

Методологія підприємницького реінжинірингу спирається та загальні етапи реінжинірингу, проте також бере до уваги культуру та менеджмент

людських ресурсів. Таким чином, відбувається аналіз цінностей підприємства та отримані результати враховуються при редизайні бізнес-процесів.

Методологія життєвого циклу є найбільш деталізованою, що передбачає постійне вдосконалення та є оптимальною для великих бізнесів. Вона місить 7 етапів: формалізація бачення бізнесу, ідентифікація сфер змін, аналіз, редизайн, оцінка, імплементація та покращення. Варто окремо зазначити етап ідентифікації, що передбачає збір даних з внутрішніх відділів підприємства, зокрема шляхом опитувань та анкетувань.

Таблиця 1.1 – Порівняльний аналіз методологій реінжинірингу бізнес-процесів

Методологія	***Етапи***	***Повнота***	***Головна особливість***	***Тип бізнесу***	***Недоліки***
Шість сігм	5	Повна	Виправлення дефектів	Універсальна	Складність та затратність впровадження
Девенпорта	2	Неповна	Акцент на технології	Невеликий бізнес	Не включає етап впровадження змін
Гаммера та Чемпі	4	Повна	Робота з персоналом	Універсальна	Може бути складною під час впровадження
Точка зупину	3	Повна	Ефективне використання ресурсів;	Універсальна	Може не враховувати особливості бізнесу
Швидкий реінжиніринг	5	Повна	Орієнтація на стратегічні процеси	Універсальна	Не чітко визначені межі між етапами
Підприємницька	3	Повна	Врахування технічних та культурних аспектів	Великі підприємства	Може вимагати значних витрат на всі аспекти
Життєвого циклу	7	Повна	Орієнтація на постійне вдосконалення	Великий бізнес	Складність через надмірну деталізацію

Розглянуті методології не складають вичерпний список, проте створений порівняльний аналіз демонструє унікальність кожної з них та наголошує на необхідності ретельного аналізу бізнес-процесів та врахуванні особливостей бізнесу.

1.2 Аналіз технологій та засобів реінжинірингу бізнес-процесів

У даному розділі буде розглянуто засоби реінжинірингу бізнес-процесів, візуалізовано їх (рис 1.1) та наведено приклади актуальних технологій, що можуть бути використані. Було взято до уваги етапи реінжинірингу бізнес-процесів, що було розглянуто у розділі 1.1 [3].

Визначення цілей та аналіз ключових результатів може здійснюватися за допомогою методики OKR [5], що полягає у визначенні не більше 3-4 цілей та 5-6 ключових результатів до кожної з цілей на одному рівні. Дана методологія не вимагає використання спеціалізованого програмного забезпечення, і може бути впроваджена в текстовому документі. Методика є актуальною, адже обмежує постановку цілей, цим самим збільшуючи фокус на пріоритетних задачах, що є важливим для сучасних бізнесів, що стикаються з ризиком перенасичення процесами.

Аналіз даних є засобом формування цілей та розуміння поточного стану бізнесу. Можливим є аналіз конкурентних даних, що також допомагає сформувати напрям змін. Одним з інструментів є платформа Tableau [6], що працює з різними типами даних, та має сервіси для підготовки та аналізу даних, а також хмарне середовище.

Моделювання є потужним засобом реінжинірингу, що є основним для етапів аналізу та редизайну бізнес-процесів детальний розгляд якого було винесено у розділі 1.4

Насамкінець, управління змінами є беззаперечною складовою імплементації, адже забезпечує ефективне використання бюджету, часу та підтримку персоналу під час змін. Згідно з дослідженням Prosci [7], управління змінами збільшило досягнення цілей бізнесами у 7 разів. Фреймворк Prosci 3-Phase Process [7] базується на моделях ADKAR, що включає поняття усвідомлення, бажання, знання, здатності та підкріплення у контексті змін та PCT, що визначає ключові фактори успіху, такі як лідерство, менеджмент

проєкту та менеджмент змін.. У першій фазі підготовки фреймворк передбачає розробку унікальної та масштабованої стратегії впровадження змін. Друга фаза управління змінами включає створення та впровадження адаптивних планів змін для персоналу та організації вцілому за допомогою моделі ADKAR. Насамкінець, третя фаза включає визначення цінності змін та їх подальшу підтримку. Prosci 3-Phase Process не вимагає специфічних інструментів.

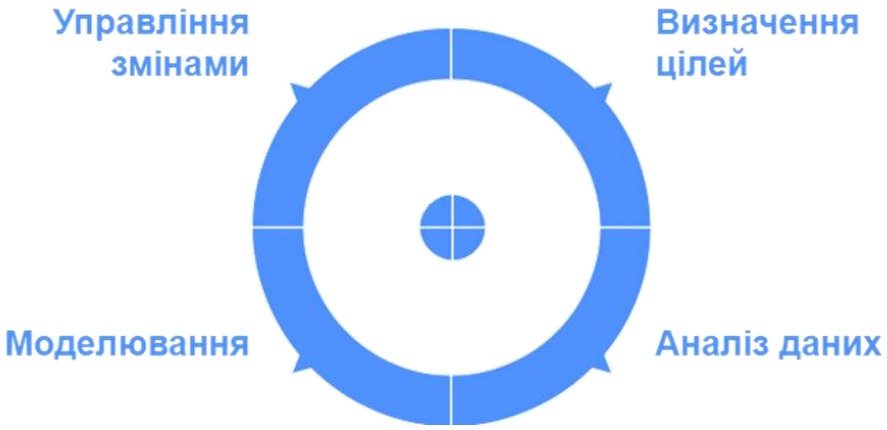

Рисунок 1.1 – Засоби реінжинірингу бізнес-процесів

Можна зробити висновок, що засоби та технології реінжинірингу є різноманітними та їх вибір залежить від багатьох факторів. Зокрема, це: цілі реінжинірингу, обрана методологія реінжинірингу, часові та фінансові ресурси та інші чинники. Було розглянуто приклади технологій, що відображають засоби, проте не є універсальними для бізнесів, адже не враховують вказаних факторів.

1.3. Огляд цілей реінжинірингу бізнес-процесів

Як було зазначено, першим кроком до реінжинірингу бізнес процесів є чітке визначення цілей. Цілі базуються не ретельному аналізі бізнес-процесів, і саме це забезпечує ефективне використання ресурсів під час реінжинірингу та визначає унікальність процесу. У підрозділі буде розглянуто та візуалізовано (рис 1.3) основні цілі, що варто розглядати як перелік [8].

Поліпшення користувацького досвіду можна визначити як основну мету реінжинірингу, адже від рівня задоволеності користувачів великим чином

залежить прибуток бізнесу. Це доводить дослідження кореляції задоволеності користувачів мобільного зв'язку до прибутку постачальників [9], проведене у Греції. Дослідження даних близько 7000 користувачів показало високу лінійну кореляцію між прибутком та задоволеністю (рис. 1.1) [9].

KPIs	*p-values*	*Global satisfaction*
Average revenue per user	0.118	0.782
Average revenue per user – voice services	0.070	0.848
Average revenue per user – data services	0.352	–0.535
Average revenue per user – contract	0.000	0.918
Average revenue per user – prepaid	0.420	0.477
Total revenue	0.017	0.941
Total retail revenue	0.002	0.985
Total retail revenue – voice services	0.070	0.848
Total retail revenue – data services	0.133	–0.764
Total retail revenue – per minute	0.063	0.858

Рисунок 1.2 – Задоволеність користувачів мобільного зв'язку

Наступною ціллю можна визначити зменшення часових та грошових витрат не невиправдані кроки та поліпшення менеджменту і розподілення ресурсів. Окрім економії, це призводить до поліпшення ефективності процесів.

Покращення якості продукту передбачає зміну процесів таким чином, аби мінімізувати помилки та дефекти. Цього можна досягти за допомогою стандартизації процесів, впровадження регулярного моніторингу та слідування актуальним практикам.

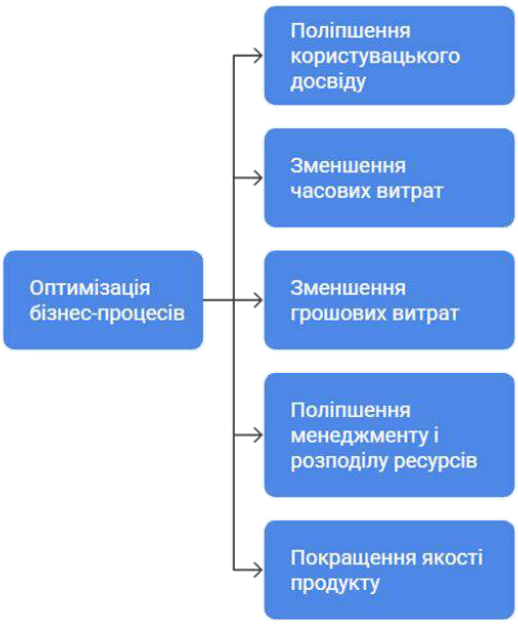

Рисунок 1.3 – Цілі реінжинірингу бізнес-процесів

В умовах сучасного бізнесу важливими є гнучкість та адаптивність процесів. Задля збереження конкурентоспроможності ключовою є здатність підлаштовуватись під тренди, нові правові та економічні вимоги, некеровані зовнішні фактори та мати змогу зменшувати розрив між пропозиціями конкурентів та власними.

Як висновок, цілі можуть мати різний зміст та формулювання, які мають випливати з специфічного для кожного бізнесу аналізу бізнес-процесів.

1.4. Створення класифікації моделей бізнес – процесів

Модель бізнес-процесу – це стандартизований та зрозумілий спосіб представлення бізнес-процесів. Моделювання забезпечує узгодженість виконання завдань, ефективну комунікацію та відповідність нормативним вимогам. Таким чином, забезпечується основа для ефективного вдосконалення процесів та значно спрощується процес реінжинірингу. Варто зазначити, що різні типи моделей слугують різним цілями, саме тому доцільним є створення класифікації. Неможливо створити єдину класифікацію моделей, тож спершу було розглянуто наявні пропоновані класифікації.

Класифікація на семантичне та несемантичне моделювання [10] дозволяє доволі точно визначитись з групою підходящих моделей залежно від сфери застосування, проте не враховує необхідності використання кількох типів моделей під час моделювання та має зовсім не рівний розподіл між двома типами, адже семантичне моделювання є слабо розвиненим, та відповідно, має менше продуктів та сфер застосування. Несемантичне моделювання включає переважно графічні інструменти, як от UML, BPMN та EPC або мовні описи процесів за допомогою інструментів як BPML, BPQL і BPEL. Графічне моделювання стало попередником для BPR та BPM, що включають проєктування, створення документації та управління змінами. Семантичне моделювання є оптимальним для роботи з динамічними та складними

середовищами. До прикладу, NextGRID базується на OWL-WS [BCG+] та позиціонується як універсальна мова для представлення процесів.

Наступна класифікація моделей на BPMN – нотацію для повного документування процесів та EPC – блок-схемне моделювання на основі подій та процесів [11], є надто спрощеною та створена скоріше задля демонстрації принципових відмінностей між двома моделями і не є повною.

Насамкінець, було розглянуто поділ на описові, аналітичні та виконувані моделі [12]. На мою думку, дана класифікація є найбільш повною та доцільною, адже базується на процесі створення моделі бізнес-процесів, а отже є найбільш застосовною до реальних ситуацій. Тим не менше, враховуючи велику кількість та різноманіття задач моделювання, було вирішено доцільним розширення даної класифікації шляхом додавання підгруп до кожної пропонованої групи. Підгрупи формувались на основі розгляду наявних інструментів моделювання, а також з урахуванням задач, що можуть бути виконані інструментами кожної підгрупи. Було створено відповідну візуалізацію (рис 1.4).

Описові моделі підходять для візуалізації процесів з метою швидкого їх розуміння. Можуть застосовуватись для комунікації зі стейкхолдерами, ознайомлення нових працівників. Вони використовують зрозумілі позначення та не є деталізованими. Першою підгрупою було виділено високорівневі моделі, що описують загальні зв’язки, ролі та етапи. Інструментом може бути мапа процесів. Наступною підгрупою є функціональні моделі, що описують залучені компоненти та їх вплив на досягнення мети. Можуть бути створені за допомогою методології IDEF0 [13] чи діаграми Swimlane [14]. Також було виділено покрокові моделі, що зображають послідовність дій у процесах та зображаються за допомогою flowchart [15]. Насамкінець, окремою підгрупою є потокові моделі. Ці моделі підходять для візуалізації ресурсів, таких як інформація чи матеріали та їх взаємодії з постачальниками та отримувачами. Підходящим інструментом є діаграма SIPOC [16].

Аналітичні моделі пропонують глибший погляд на процеси і часто беруть за основу описові моделі. Вони є більш детальними, містять конкретну

інформацію, як от кількість, час чи витрати та допомагають визначити вузькі місця, проблеми і можливі шляхи вирішення. Цей тип поділено на чотири підгрупи, перша з яких оптимізаційні моделі. Фокусом цих моделей є мінімізація будь-яких витрат – часу, коштів, чи будь-яких інших ресурсів. Прикладом інструменту є Minitab [17]. Наступна підгрупа – імітаційні моделі. Вони мають на меті моделювання різних сценаріїв та їх оцінку. Є корисними в реінжинірингу, адже дозволяють оцінити необхідність та сутність змін. Моделювання може виконуватись за допомогою метода Монте-Карло з використанням інструменту Crystall Ball [18]. Третю підгрупу було визначено як моніторингові моделі, метою яких є аналіз продуктивності етапів процесу. Може бути реалізовано за допомогою Balanced scorecard [19]. Останньою підгрупою є моделі ризиків, що підходять для пошуку та аналізу ризиків та знаходження шляхів їх мінімізації. Інструментом може бути FMEA [20].

Виконувані моделі мають найвищий рівень деталізації та включають усі необхідну інформацію для автоматизації процесу. Під автоматизацією найчастіше мається на увазі застосування технологій, проте коректним буде зазначити, що таким чином можуть бути описані і процеси незалежні від них, що було враховано при формулюванні підгруп. Першою підгрупою є неавтоматизовані моделі. Вони відображають процеси, що виконуються виключно людиною, або за мінімального втручання технологій. Інструментом є BPMN [21]. З іншого боку – напівавтоматизовані моделі включають процеси, що збалансовано використовують дій людини та технології. Прикладом може бути інтеграція з ERP чи CRM. Третьою підгрупою було виділено повністю автоматизовані процеси, що не вимагають людського втручання. Інструментом може бути BPMS [22]. Насамкінець, доцільно буде включити і моделі реального часу, що є повністю автоматизованими та відповідають за, здебільшого, комплементарні процеси, чутливі до часу. Інструментом може бути інтеграція з Інтернетом речей.

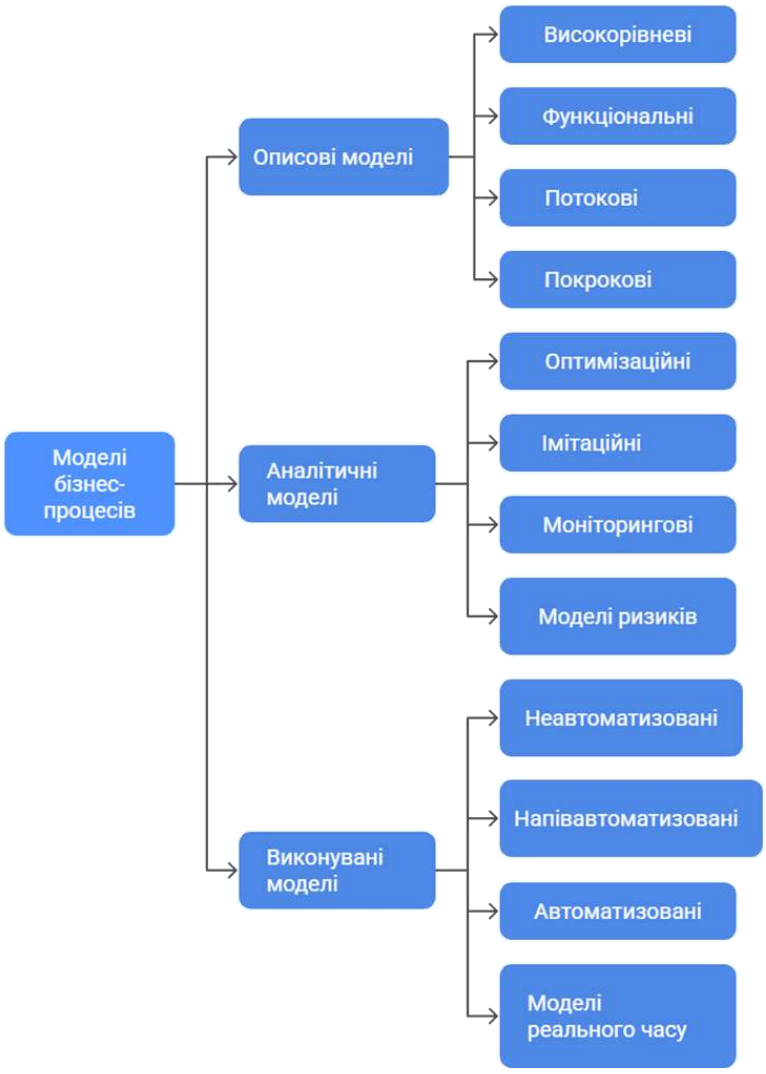

Рисунок 1.4 – Класифікація моделей бізнес-процесів

Отже, було розглянуто поняття бізнес-процесів, їх моделювання, запропоновані класифікації моделей, а також існуючі інструменти для створення моделей. На основі отриманих даних було розроблено власну класифікацію моделей бізнес-процесів та наведено приклади інструментів, що можуть використовуватись для створення моделей.

РОЗДІЛ 2. РЕІНЖИНІРИНГ ТА ОПТИМІЗАЦІЯ ПРОГРАМНИХ СИСТЕМ

2.1 Реінжиніринг програмних систем. Аналіз технологій, методів та засобів

Нагальна потреба у реінжинірингу програмних систем виникла наприкінці двадцятого століття. Цю потребу можна пов’язати з розвитком веб-технологій та закономірною міграцією великої частки програмних систем, що використовувалась бізнесами, на веб-інтерфейси. У 1960-х роках частка витрат на програмне забезпечення складала лише 20% від вартості системи. У той же час, 80% складали витрати на обладнання. За кілька десятиліть показники змінились внаслідок зменшення вартості виробництва і її росту та збільшення витрат на оплату праці. Не менш важливим фактором стали скорочення державних фінансувань на розробку нових систем. Таким чином, пріоритетними стали програмні системи, що легко масштабувати, адаптувати та підтримувати, адже розробка абсолютно нових рішень перестала бути економічно вигідною [23].

Поняття реінжинірингу програмних систем можна визначити як процес аналізу, перебудови та модернізації існуючих програмних систем з метою підвищення їх ефективності, підтримуваності, продуктивності та адаптивності до сучасних вимог. Процес реінжинірингу доцільно розділити на три сегменти, першим з яких є аналіз наявної програмної системи. Метою даного етапу є аналіз переваг та недоліків. Другий етап – визначення функціональних та нефункціональних вимог. Третім етапом є внесення змін до системи, продуктом яких є оновлена система, що задовольняє визначені на попередньому етапі вимоги. Невід’ємними етапами є тестування та створення документації, проте ці процеси відбуваються паралельно із модифікацією системи.

Реінжиніринг поділяють на реінжиніринг даних та реінжиніринг процесів [23]. Основною метою реінжинірингу даних є пошук прихованих моделей та їх використання, а також стандартизація визначень. Зокрема, інструментами для виявлення закономірностей у великих наборах даних є RapidMiner та Apache Spark. Реінжиніринг процесів є більш об’ємним та включає в себе кілька понять

Зворотній інжиніринг [23] можна визначити як аналіз компонентів та зв’язків наявної системи з метою створення її представлення на певному рівні абстракції. Це комплексний процес, що включає в себе не лише дослідження системи безпосередньо, а й аналіз документації та збір користувацької інформації. Проте він є виключно аналітичним і не передбачає внесення змін до системи. Окремими процесами зворотного інжинірингу виділяють редокументацію – створення систематизованої та повної документації, за умови збереження абстракції та семантики, та реархітектуру – збір повної інформації про систему та її роботу, тобто про архітектуру системи. IEEE визначив «архітектуру» як «структуру компоненти, їх зв'язки, принципи і керівні принципи, що регулюють їх дизайн, і еволюцію» (IEEE, 2000). Прикладом інтегрованого середовища розробки, що надає можливість автоматизовано знаходити та візуалізувати зв’язки у системі, проводити тестування на відповідність стандартам, а також створювати документацію, базуючись на коментарях є Understand [24].

Внаслідок комплексного аналізу системи з точки зору даних та процесів та визначення вимог, відбувається її модифікація за принципом прямої інженерії [23], яку можна визначити як перехід від ідеї до фізичної імплементації. Виділяють декілька підходів реінжинірингу, які доцільно застосовувати згідно з потребами, які висувають визначені вимоги, а саме: рефакторинг, реплатформінг, модернізація користувацького інтерфейсу та агрегація програмних систем.

Рефакторинг [23] є процесом покращення внутрішньої структури коду без зміни його зовнішньої поведінки з метою спрощення обслуговування. SonarLint є плагіном до IDE, що інтегрує можливості SonarQube для автоматичного аналізу та рефакторингу під час написання коду.

Реплатформінг передбачає перенесення програмного продукту на іншу платформу або мову програмування з метою покращення продуктивності та сумісності. Варто виділити сучасну тенденцію міграції програмних систем на хмарні платформи. Згідно з дослідженням Foundry [25], 57% компаній

здійснюють міграцію своїх IT середовищ до хмари станом на 2024 рік. Відповідні хмарні сервіси пропонують власні інструменти переносу даних.

Модернізація користувацького інтерфейсу здійснюється з метою покращення привабливості та доступності веб-інтерфейсу. Привабливість інтерфейсів досягається за допомогою аналізу поточних трендів та дотримання фундаментальних принципів дизайну. При створенні доступних інтерфейсів орієнтирами є вказівки доступності WCAG - загальноприйняті вимоги до якісного програмного продукту. Дослідження Всесвітньої організації охорони здоров'я [26] свідчать про те, що 1.3 мільйони людей має серйозну інвалідність. При цьому, було виявлено [27] близько 51 мільйона порушень доступності внаслідок аналізу 1 мільйона веб-сайтів. Закономірно, що доступний користувацький інтерфейс є не менш важливим для досягнення бізнес-цілей чи роботи компанії, ніж привабливий дизайн.

Агрегація [28] систем передбачає злиття кількох підсистем у систему, що функціонує як єдине ціле. Таке рішення спрямоване на зменшення складності та обсягу витрат на підтримку. Вертикальний метод агрегації передбачає створення функціональних самостійних підсистем, що повністю виконують окремі функції. Горизонтальний метод використовує спеціалізовану систему - ESB, що відповідає за пряму комунікацію між усіма підсистемами. Зіркова агрегація застосовує безпосереднє з'єднання між системами без посередника. Агрегація життєвого циклу передбачає повну розробку системи, яка повністю відповідає зазначеним вимогам та забезпечує створення стандартизованої системи. Було самостійно розглянуто та проаналізовано методи агрегації програмних систем (табл 2.1).

Таблиця 2.1 – Порівняльний аналіз методів агрегації

Метод агрегації	***Горизонтальна***	***Вертикальна***	***Зіркова***	***Життєвого циклу***
Швидкість впровадження	Помірна	Висока	Помірна	Значно залежить від масштабу
Гнучкість	Висока	Низька	Висока	Висока
Вартість впровадження	Низька	Низька	Висока	Висока
Вартість підтримки	Низька	Висока	Висока	Помірна
Масштабованість	Висока	Низька	Низька	Висока
Переваги	Зменшення кількості з'єднань; простота заміни підсистем	Швидкість впровадження; простота у початковій реалізації	Висока гнучкість; повторне використання функцій	Стандартизована архітектура; урахування всіх вимог
Недоліки	Залежність від налашту-вань ESB; висока вартість створення адаптерів	Висока вартість обслуговування; складність масштабування	Висока вартість інтеграції; низька масштабованість	Складність впровадження; висока початкова вартість

Можна зробити висновок, що кожен з методів має свої переваги та недоліки і, відповідно, свої задачі, до яких може бути застосований найбільш ефективно.

- Горизонтальну агрегацію найдоцільніше використовувати у великих проєктах, де є важливими гнучкість та масштабованість.
- Вертикальна агрегація підійде для невеликих проєктів із обмеженими вимогами через низькі швидкість та вартість впровадження.
- Зіркова агрегація найкраще застосовна до динамічних систем з великою кількістю взаємодій. Зокрема, ефективною є при створенні MVP та стартапів, де важливими є висока швидкість впровадження.
- Агрегація життєвого циклу підходить для систем, де важливою є стандартизація, наприклад, державна чи фінансова сфера.

Технології штучного інтелекту здатні аналізувати код програмних систем, виявляти помилки, потенційні ризики та пропонувати шляхи їх вирішення. До

прикладу, CodeGuru [29] від AWS здатен використовувати машинне навчання для формування безпекових рекомендацій, а DeepCode [30] аналізує код на ризики та надає пропозиції рефакторингу. Можна стверджувати, що обсяги використання штучного інтелекту у сфері реінжинірингу програмних систем зростатимуть. Дослідження МакКінзі [31] зазначає, що використання штучного інтелекту зменшило час, витрачений на рефакторинг на 30%. У той же час, старший інженерний менеджер Duolingo описав GitHub Copilot [32] – другий за популярністю помічник написання коду, як ефективний інструмент та додав, що завдяки ньому компанії Duolingo вдалося скоротити середній час перевірки коду на 67% завдяки контекстним пропозиціям.

2.2. Огляд передумов та переваг реінжинірингу програмних систем

Поняття реінжинірингу було визначено у попередньому підрозділі, проте варто розглянути його глибше, а саме сформулювати переваги та передумови реінжинірингу, а також розглянути реінжиніринг як складову модифікації програмної системи.

Модифікація передбачає усі зміни, що вносяться до системи, а зокрема розробка оновлень, вирішення проблем, поліпшення функціональної складової. Доцільно виокремити чотири категорії модифікацій [33]. Першою є адаптивні зміни, що спрямовані на пристосування до нових вимог бізнесу чи технологій. Корекція – це виправлення проблем, що є в системі. Покращення передбачає поліпшення продуктивності та стабільності системи. Насамкінець, превентивна модифікація спрямована на подовження життєвого циклу.

Зазвичай, реінжиніринг є етапом, що слідує за всіма модифікаціями і потреба в ньому виникає, коли підтримка та оновлення системи стає надато дорогою. Варто зазначити, що реінжиніринг може збігатися з концепцією модифікації у певних випадках, проте не завжди, тож ці поняття варто розмежовувати.

Таким чином, можна виділити кілька передумов реінжинірингу (рис 2.1):

- розширення бізнесу,
- застарілі технології,
- зростання складності системи, внаслідок відсутності структури,
- бажання відповідати трендам та підвищення конкуренції,
- зовнішні фактори

Рисунок 2.1 – Передумови реінжинірингу програмних систем

Розширення бізнесу включає потреби обслуговування більшої кількості клієнтів та збільшення обсягів операцій. Застарілі технології вимагають більше витрат на обслуговування, обмежують продуктивність та масштабованість. Складна система ускладнює підтримку, створює загрозу надлишкових процесів та знижує прозорість. Відсутність структури робить складними інтеграції та підтримку. Відповідність трендам можна визначити як одну з найактуальніших потреб бізнесу, яка є значною складовою конкурентоспроможності. Власне підвищення конкуренції може спонукати переосмислення системи задля більш ефективного розподілу ресурсів. Зовнішні фактори є необмеженими, проте найбільш значущими можна виділити зміни у законодавстві, суспільстві та міжнародних стандартах [33].

Можна виділити п’ять основних переваг реінжинірингу програмних систем (рис. 2.2), за умови його виправданого здійснення.

Збільшення здатності до підтримки [34] є часто вирішальним фактором при прийнятті рішення про реінжиніринг. Це закономірно, адже чим легше адаптувати систему до змін та виправляти проблеми, тим менш затратною є

система. До прикладу, рефакторинг системи сприяє збільшенню показника здатності до підтримки, як і інші засоби реінжинірингу.

Покращення продуктивності системи [34] до очікуваних стандартів є необхідністю для стабільної роботи бізнесу. Якщо система стикається із проблемами з продуктивністю, вона вступає у кризу, що вимагає негайного вирішення, адже виникає загроза втрати іміджу компанії чи зупинки процесів.

Підвищення сумісності згідно з IEEE [34] визначається як здатність двох і більше складових чи навіть цілих систем успішно обмінюватись інформацією та використовувати її. Надзвичайно актуальною така ефективна взаємодія є при інтеграції старого програмного забезпечення з новим.

Полегшення міжкомандної та міжособової комунікації [35] може бути прямим наслідком реінжинірингу. Команди розробників та окремі розробники можуть мати абсолютно різну обізнаність щодо старих систем часто через застарілість та складність цих систем, а також відсутність належної документації, а це може ускладнити взаємне розуміння. Як наслідок, реінжиніринг старих систем та редокументація зменшують залежність розробників одне від одного та оптимізують комунікацію.

Реінжиніринг для полегшення процесу тестування [35] не є складним і є виправданим, адже значно зменшує затрати на тестування. Його метою є рефакторинг системи так, аби задовольнити критерії тестування, а також мінімізувати складність та розмір. Такі задачі як рефакторинг, пошук глибокого коду можуть бут виконані автоматично, а от оптимізація алгоритмів та спрощення інтерфейсу, можуть бути виконані вручну з відносно невисокими витратами.

Рисунок 2.2 – Переваги реінжинірингу програмних систем

2.3 Порівняльний аналіз підходів до реінжинірингу

Можна виділити три підходи до реінжинірингу програмних систем [36], кожен з яких має переваги та недоліки, тож, як наслідок, має застосовуватись до відповідних задач.

Перший підхід є покроковим. Він передбачає послідовне оновлення компонентів системи, спираючись на її структуру. Поетапність внесених змін мінімізує ризики порушень роботи системи та забезпечує безперебійну роботу системи. Значно спрощується аналіз та виправлення помилок. Покроковий підхід є часозатратним, адже вимагає детального аналізу компонентів системи і унеможливлює зміну основної архітектури системи, допускаючи реінжиніринг лише окремих компонентів. Також, можуть виникати труднощі інтеграції нових компонентів з вже наявними. Як наслідок, покроковий підхід є оптимальним для складних систем, де повна заміна компонентів чи переосмислення архітектури не є практичними.

Підхід великого вибуху є комплексною модифікацією системи, що відбувається за один великий цикл. Зокрема, такий підхід є застосовним до задач, що вимагають одночасного рішення, як от перенесення системи до хмари, реінжинірингу архітектури чи повної заміни стеку технологій. Варто зазначити, що даний підхід не є оптимальним для об’ємних систем, адже час на впровадження може перевищувати допустимий. Підхід великого вибуху

потребує детального аналізу та планування, проте уникає проблеми сумісності про інтеграції нових та старих складових системи. Підхід великого вибуху ускладнює пошук та виправлення помилок, а також може вимагати припинення роботи систем на час впровадження змін.

Еволюційний підхід схожий за концепцією до послідовного, проте концентрується на поліпшенні функціональної складової, а не на структурі системи, а отже є орієнтованим на адаптацію до нових потреб користувачів чи умов ринку. Впровадження змін є відносно легшим за послідовний підхід, адже зміни спрямовані на розвиток і передбачають плавний перехід без суттєвих змін архітектури. З іншого боку, даний підхід потребує реорганізації схожих функцій у єдину функціональну одиницю. Також варто враховувати структурні обмеження при оновленні чи впровадженні нових функцій, адже невідповідність може викликати проблеми із часом відповіді на запити системи.

Було створено власний порівняльний аналіз підходів (табл. 2.2)

Таблиця 2.2 – Порівняльний аналіз підходів до реінжинірингу

Підхід	***Покроковий***	***Великий вибух***	***Еволюційний***
Предмет орієнтації	Структура	Весь проєкт	Функціональна складова
Темпи змін	Повільні	Швидкі	Помірні
Ризики	Низькі	Високі	Середні
Сфера застосування	Оновлення системи	Міграція; заміна стеку; реархітектура системи	Адаптація до бізнес-вимог
Тип систем	Складні та об’ємні	Необ’ємні	Будь-які

Розглянуті підходи є оптимальними для певних умов використання. До прикладу, покроковий та еволюційний підходи є найвигіднішими з точки зору ризиків, проте мають принципово інші сфери застосування. Так само і підхід великого вибуху має великий рівень ризиків, проте часто стає єдиним можливим підходу до реінжинірингу систем, адже дозволяє вирішити певний специфічний тип задач.

2.4. Аналіз методів оптимізації програмних систем.

Оптимізація програмних систем є однією з задач реінжинірингу, яка заслуговує окремої уваги. Визначити оптимізацію можна як процес удосконалення системи з метою підвищення її продуктивності та ефективності використання ресурсів а також надійності та зручності підтримки. Під ресурсами можна розуміти [37]:

- час виконання запитів системою,
- пам’ять,
- енергоспоживання.

Неналежним чином оптимізована програмна система перешкоджатиме її роботі та негативно впливатиме на задоволення потреб бізнесу. Неналежною можна вважати як недостаню, так і надмірну оптимізацію. Видатний інформатик Дональд Кнут стверджував, що оптимізація заради оптимізації спричиняє багато проблем, коли справа доходить до підтримки та пошуку помилок [38].

Належна оптимізація вимагає ретельного планування, тож оптимізація в контексті реінжинірингу може стати джерелом непередбачених ризиків та як часових, так і економічних витрат. Система, специфікація якої не дозволяє досягти визначених вимог продуктивності з великою ймовірністю потребуватиме суттєвих змін, що може призвести до значного зростання складності проєкту, тож розробці плану реінжинірингу порібно виділяти достатньо ресурсів на аналіз системи, та відповідно, впровадження змін.

Можна виділити комплексний підхід до оптимізації (рис 2.3) та його особливості. Першим кроком є оцінка ефективності системи. Метриками є час відповіді системи, що визначає час, необхідний системі для виконання запиту та пропускна спроможність, що позначає частоту виконання транзакцій та демонструє можливості системи витримувати пікові навантаження. Обидві метрики є невід’ємно пов’язаними з рівнем задоволеності користувачів. З іншого боку, використання ресурсів, таких як пам’яті та мережі, показують ефективність використання обладнання та допомагають ідентифікувати вузькі місця.

Другим кроком здійснення оптимізації є аналіз зібраних метрик та їх використання для формулювання чітких проблем, що потребують вирішення. Цей етап є важливим, адже запобігає здійсненню необґрунтованої оптимізації, що є джерелом невиправданих витрат та проблем в системі.

Наступними етапами є безпосереднє впровадження змін та їх тестування. Тестування має включати в себе не лише перевірку змінених чи додаданих елементів коду на відсутність помилок, а й моніторинг метрик, аналогічний до проведеного на етапі аналізу. Існує кілька типів моніторингу продуктивності, перший з яких – автоматичний. Він дозволяє імітувати сценарії завантаження. Прикладами сервісів є JMeter та LoadRunner.

Користувацький моніторинг передбачає збір даних від кінцевих користувачів у формі опитувань чи інтерв'ю.

Постійний моніторинг аналізує ключові метрики продуктивності у реальному часі та значно підвищує якість не лише тестування, а й обслуговування системи загалом. Поширеними сервісами є Datadog та New Relic.

Також є ефективним впровадження періодичних аудитів продуктивності системи щодо сучасних стандартів [37] [39].

Рисунок 2.3 – Етапи оптимізації програмних систем

Оптимізувати програмну систему можливо у багатьох її складових (рис 2.4), проте варто зазначити, що найбільший вплив має найвищий рівень, а саме рівень проєктування. У той же час, саме його найважче модифікувати. Він передбачає вибір мови програмування та архітектури, які є складними для модифікацій. У випадку з заміною мови програмування, може бути необхідне повне переписування коду або часткова його заміна.

Прикладом є всесвітньо відома стримінгова платформа Netflix [40]. Початково, Netflix мав монолітну архітектуру, проте із ростом користувачів платформу, виправлення частих перебоїв у роботі стало складним. Саме тому архітектуру Netflix було змінено на мікросервісну, де кожен сервіс відповідав за окрему функціональну складову. Ця модифікація суттєво збільшила масштабованість та спростила випуск оновлень. Також, усю систему було переміщено на хмару, а саме Amazon Web Services, що дозволило використання сервісів, як от EC2 обчислення та доставка контенту через Cloud Front. Загалом, такі зміни арахітектури Netflix зробили платформу оптимізованою для своїх вимог та допомогли зайняти лідерську позицію серед стримінгових сервісів.

Наступною складовою є алгоритми та структури даних. У випадку з алгоритмами важливо аналізувати не лише часову та просторову складності, а й дані та тип задачі, до яких алгоритм застосовано. Зокрема, при виборі алгоритму для частково відсортованих даних найкраще підійде сортування вставкою, а не швидке сортування, яке є найшвидшим на несортованому масиві даних.

Для оптимізації структур даних важливо визначити операції, що виконуються над даними та, базуючись на цьому обирати структуру. Зокрема, якщо передбачається часта модифікація даних, структура масив не буде оптимальною, на відміну від зв’язаних списків.

Як приклад успішної оптимізації алгоритмів, було розглянуто Amazon [41]. У 2014 році було проведено оптимізацію алгоритмів персоналізації Prime Video – стримінгової платформи Amazon. До введення змін, для персоналізації контенту використовувалася факторизація матриці, а методом заповнення слугувала нейронна мережа – автоенкодер. Її робота полягала у виведенні вхідних даних через вузьке місце, тож мережа навчалась стискати та розширювати дані. Цей метод порівняли іще з двома – методом персоналізації на основі вже переглянутих фільмів та методом на основі найпопулярніших стрічок наразі. Обидва методи, внаслідок тестувань з користувачами, виявились набагато ефективнішими за енкодер, адже враховували не лише смаки, а й сучасні тренди. Таким чином, енкодер почали тренувати на даних, відсортованих в

хронологічному порядку до певної межі, а оцінка даних відбувалась на основі переглянутих стрічок у двотижневому проміжку після межі.

Також застосувати оптимізацію можна до вихідного коду [37]. Вона передбачає уникнення вкладених циклів, повторних викликів функції, невиправданих обчислень, а також коректне звільнення пам’яті після роботи з об’єктами. Фактично, оптимізація на даному рівні є рефакторингом коду, що може виконуватись як автоматично, так і вручну.

Кешування є ефективним методом оптимізації [42], адже дозволяє зберігати часто використовувані дані у пам’яті та видобувати їх звідти, що значно економить час на виконання операцій. Важливим є ретельний вибір даних для кешування, аби запобігти застаріванню інформації та пропускам у кеші. Також важливою є методологія очищення кешу, що може полягати як у очищенні найменш часто використовуваного, так і найдавніше використовуваного кешу.

Існує декілька типів кешування, а саме процесорне, дискове, кешування пам’яті та браузера. Окремо можна виділити розподілене кешування, яке дозволяє зберігати дані на кількох серверах і, як наслідок, покращує масштабованість.

Кешування є широко застосовним до баз даних, і прикладом такого успішного застосування є компанія Uber [43]. Uber використовував Docstore – базу даних, побудовану на MySQL, що дозволяла працювати з мільйонами запитів на секунду. Проте з розширенням мікросервісів, з’явилась потреба у зменшені затримки та збільшенні масштабованості, тож було розроблено інтегровану систему кешування CacheFront. Основними причинами створення CacheFront Uber виділив повільну роботу з дисковою пам’яттю, складність горизонтального розширення, дисбаланс читання та запису, неефективність та недовговічність збільшення потужностей бази даних, а також невдалу імплементація кешування з Redis. Спроба впровадження мала проблеми з реплікацією та високою затримкою. Таким чином, впровадження CacheFront мало наступні особливості, по-перше, використання стратегії cache-aside, де дані спершу шукаються в Redis, а за умови промаху – у базі даних із подальшим

асинхронним кешуванням. По-друге – впровадження інвалідації кешу, а саме можливостей механічної інвалідації та системи дедуплікації за допомогою порівняння часових міток. По-третє, використання негативного кешування – збереження результатів з порожніми даними та введення режиму compare cache, що дозволяє мінімізувати розбіжності між базою даних та кешем. Також було впроваджено міжрегіональну реплікацію – процес відновлення кешу через повторне виконання вибіркових запитів для узгодженості між регіонами. Таким чином, компанія Uber взяла до уваги свої бізнес – потреби та нефункціональні вимоги та створила продуктивну, енергомічну та єдину систему кешування.

Можна виділити оптимізацію баз даних як іще один рівень [44]. Найпершим підходом є переписування запитів, що збільшує швидкість, проте може створити проблеми з сумісністю та зрозумілістю. Індексування зменшує час доступу до даних, адже мінімізує необхідність аналізу всієї таблиці, проте вимагає ретельного вибору колонок для індексації. Кешування дозволяє зберігати результати частих операцій та, відповідно, зменшує навантаження і збільшує продуктивність. Зменшення дискових операцій введення-виведення впливає на продуктивність, адже операції на диску є значно повільнішими за операції з пам’яттю. Також використовують шардування, що полягає у горизонтальному масштабуванні та поділ даних, що спрощує паралельну обробку.

Можливою є і оптимізація схеми бази даних, що включає нормалізацію – процес усунення надлишковості та денормалізацію, що зменшує складність об’єдань.

Прикладом може слугувати компанія Uber [45], що висвітлила важливість індивідуальних рішень і оптимізації при роботі з інфраструктурою даних. Компанія використовує та зберігає 100 петабайт даних, тож відкриті бази як от Cassandra не є оптимальним вибором для подібного навантаження. Було детальніше розглянуто рішення компанії для оптимізації пошуку даних. Uber створили компонент глобальний індекс. Цей компонент виконує пошук даних у таблицях Hadoop, що надає можливість ефективно завантажувати дані.

Завантаження поділяється на додавання – це незмінні події, як от історії платежів та на додавання з оновленням – події, що мають кілька оновлюваних станів, як от статус поїздки. Другий тип, вимагає контекст, а саме останні попередні значення, а отже доступ до розташування цих даних. Відповідними є етапи завантажень: початковий – завантаження історичних даних та інкрементальний – додавання оновлених даних. Імплементація глобального індексу відбувається через HBase, Apache Spark, що відповідає за генерацію та HFile, що дозволяє масове завантаження файлів. Аби уникнути перевантаження, було імплементовано механізм обмеження пропускної здатності запитів, проте ця можливість зростає лінійно зі збільшенням кількості серверів. Як висновок, система глобального індексування дозволяє Uber ефективно обробляти та оновлювати величезні обсяги даних, забезпечуючи надійність і масштабованість.

Іще однією складовою є оптимізація мережевих запитів [46]. Вона є важливою для збереження уваги користувача та є не перевагою, а скоріше вимогою у сучасних програмних системах. Можна виділити кілька методів оптимізації. Найпершим є мінімізація циклів. Натомість варто групувати запити, аби обмежити кількість кругових заходів і збільшити швидкість. Також ефективними є поступова передача даних, при якій відразу передаються лише дані, необхідні для завантаження, а решта надається поступово, за необхідності. CDN дозволяють прискорити доставку статичних та навіть динамічних елементів, адже розподіляють їх на кілька серверів та здійснюють доставку отримувачу з найближчого.

Іще одним методом є використання HTTP/2 протоколу [47], що здатен передавати більше одного запиту через одне з’єднання, а також стискати заголовки і використовувати бінарний протокол для полегшення передачі і фреймінгу даних. Також HTTP/2 здатен надсилати пов’язані дані з ініціативи сервера, аби не чекати на запит від клієнта. Також можливе надання переваги критичним запитам, і пріоритизація кешу, аби не виконувати зайвих запитів.

Прикладом може слугувати PicsArt [48] – онлайн відеоредактор та фото редактор, що використовує Cloudflare. Завдяки CDN, що складається з мережі

330 міст, вдалося пришвидшити час завантаження зображень у 2-3 рази та зекономити 50% місячної пропускної здатності.

Задля оптимізації також використовуються паралельне програмування та багатопотоковість [49]. Паралельне програмування базується на можливості розділити задачі на простіші та виконувати їх одночасне обчислення і передбачає кілька процесів та зв’язок між ними. Натомість багатопотокове програмування використовує один процес, проте кілька потоків в ньому і є найкраще застосовним до багатоядерних процесорів. Перевагами багатопотоковості є ефективне використання ядер, висока масштабованість та невибагливість до конфігурацій обладнання. У той же час, це не впливає на користувацький досвід, адже потоки можуть виконувати фонові операції без порушення роботи інтерфейсу.

Часто паралельне програмування та багатопотоковість інтегрують, що дозволяє підвищити продуктивність системи. Під час інтеграції варто звертати увагу на коректність поділу задач та даних, а також уникнення надмірної взаємодії між потоками для мінімізації появи вузьких місць в системі.

Цікавим прикладом використання багатопотоковості є обчислення Excel [50]. Починаючи з версії Excel 2007, підтримується багатопотокове обчислення для прискорення перерахунку формул на робочих аркушах. Операційна система самостійно розподіляє потоки між ядрами, при чому кількість ядер може дорівнювати одному. Максимально можлива кількість потоків становить 1024. Багатопотоковість застосовується лише до певних функцій і обмежується обчисленими клітинками, проте зменшує час обчислень на розподілених обчисленнях, як от робота з віддаленими серверами. Для досягнення ефективності важливо враховувати обмеження обладнання та використовувати коректні формули.

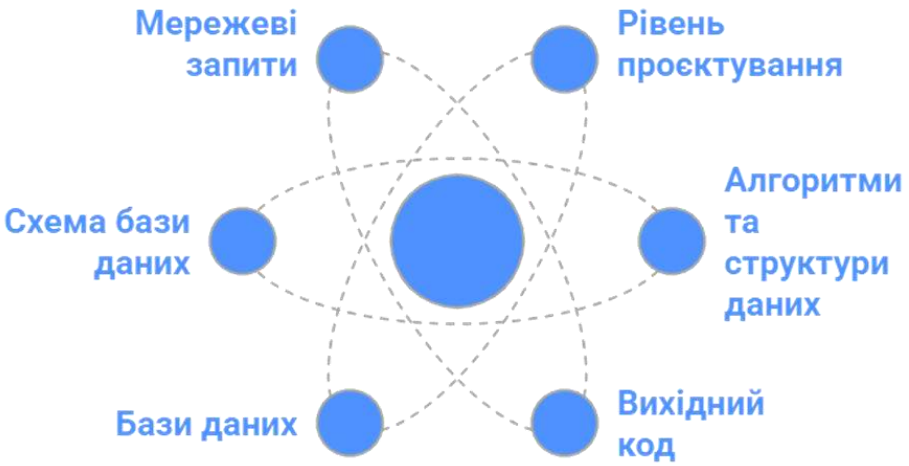

Рисунок 2.4 – Складові оптимізації програмних систем

РОЗДІЛ 3. ПРАКТИЧНЕ ЗАСТОСУВАННЯ МЕТОДІВ РЕІНЖИНІРИНГУ БІЗНЕС-ПРОЦЕСІВ

3.1 Огляд обраного бізнесу та конкурентних бізнесів

Першим завданням даного розділу є безпосередній аналіз та редизайн бізнес-процесу та відповідних елементів веб-сайту, які є невід'ємною частиною відповідних бізнес-процесів. Редизайн елементів веб-сайту є не лише структурним, а й бере до уваги веб-доступність та відповідність сучасним тенденціям дизайну веб-сайтів. Варто зазначити, що ця частина роботи не передбачає висвітлення усіх етапів реінжинірингу, а лише етапи аналізу та редизайну бізнес-процесів. Це пояснюється обмеженим доступом до, зокрема, внутрішніх даних про компанію. Було проведено інтерв'ю з представником компанії, яке дозволило отримати важливі для здійснення аналізу та редизайну дані, проте цих даних недостатньо, аби здійснити усі етапи реінжинірингу. Саме тому висвітлений у даній роботі редизайн можна вважати вірогідним для цієї компанії, а не повністю застосовним. Дані отримані під час інтерв'ю будуть використані під час обґрунтування тих чи інших рішень у редизайні.

Друге завдання є комплексним та включає в себе визначення оптимальної методології реінжинірингу та обґрунтування її вибору, формулювання детального плану реінжинірингу, а також ідентифікація та аналіз ризиків з прозиціями заходів їх оптимізації. Як і у першому етапі, отримані напрацювання є вірогідними, враховуючи обмежений доступ до ресурсів компанії.

Об'єктом було обрано середній бізнес – медичний центр "Дитина". Бізнес спеціалізується на наданні спеціалізованих консультацій у медичному центрі, консультацій вдома у пацієнта, проведенні маніпуляційних процедур, надання медичних довідок, онлайн консультацій та діагностиці. Внаслідок контакту з бізнесом було визначено, що переважною групою пацієнтів є діти, проте послуги вакцинації надаються і дорослим. Медичний центр має три філіали, які об'єднані єдиною системою запису пацієнтів.

Також для аналізу було обрано два конкурентні бізнеси (рис 3.1).

Першим з них є середній бізнес – клініка Гармонія здоров’я. Бізнес надає послуги як дітям так і дорослим. Послуги дітям включають консультації спеціалістів, наданні медичних довідок, діагностиці, проведенні маніпуляційних процедур, а також онлайн-консультації.

Також було обрано середній бізнес – медичний центр “Малючок”. Спеціалізацією бізнесу є надання спеціалізованих консультацій, проведенні маніпуляційних процедур, наданні медичних довідок та діагностиці. Вдалося з’ясувати внаслідок контакту з бізнесом, що послуги вакцинації надаються як дітям, так і дорослим. Решта послуг надається виключно дітям.

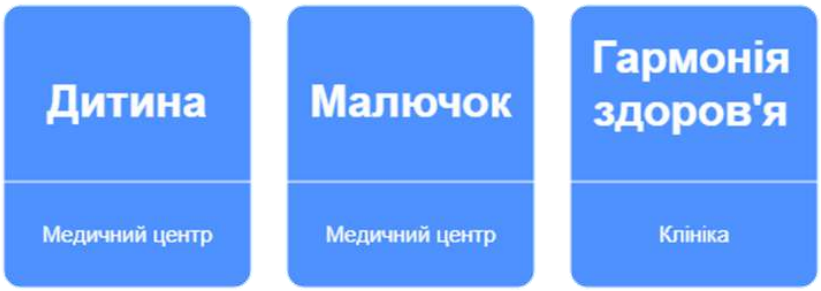

Рисунок 3.1 – Обрані для аналізу бізнеси

3.2. Аналіз та редизайн бізнес-процесу

З метою аналізу та редизайну було обрано бізнес-процес запису на послуги до медичного центру. Саме цей процес відіграє ключову роль у залученні нових клієнтів та повторному записі постійних. Внаслідок інтерв’ю було визначено, що первинний запит запису на будь-яку послугу може відбуватись чотирма способами, а саме:

- запис через форму веб-сайт медичного центру,
- за номерами телефонів, вказаних на веб-сайті,
- через партнерські страхові компанії
- через веб-сайт licarni.com.

Перші два способи станом на 2025 рік є найбільш використовуваними та становлять переважну кількість звернень.

Також на процеси лише цих двох способів бізнес має безпосередній вплив, тому саме їх було обрану для детального розгляду. Також було досліджено, що бізнес наразі не проводить рекламних кампанії чи просувань у соціальних мережах, проте приділяє значну увагу просуванню веб-сайту у пошуковій системі Google за допомогою платних пошукових кампаній та локальному SEO-просуванню.

Для проведення аналізу бізнес-процесу було використано моделювання, аналіз конкурентних бізнесів та SWOT-аналіз.

Першим етапом було самостійне створення порівняльного аналізу функціональних можливостей (табл. 3.1).

Таблиця 3.1 – Порівняльний аналіз функціональних можливостей

Номер	***Функція***	***МЦ “Дитина”***	***МЦ “Малючок”***	***МЦ “Гармонія здоров’я”***
1	Веб-сайт	Так	Так	Так
2	Кнопка запису на прийом	Так	Так	Так
3	Кнопка зворотного зв’язку	Ні	Ні	Так
4	Контактні телефони	Так	Так	Так
5	Можливість самостійного запису пацієнта	Ні	Ні	Частково
6	Можливість залишити коментар у формі запису	Ні	Так	Ні
7	Наявність чату на вебсайті	Ні	Ні	Так
8	Наявність особистого кабінету пацієнта	Ні	Ні	Ні
9	Цілодобове підтвердження запису	Ні	Ні	Ні
10	Нагадування пацієнту про запис	Так	Ні	Так
11	Чи отримує лікар автоматичне сповіщення про запис	Ні	Ні	Так
12	Скасування запису через веб-сайт	Ні	Ні	Ні

З метою візуалізації конкурентного аналізу, було створено діаграму (рис 3.2), яка дозволяє відобразити доступні функції кожного бізнесу.

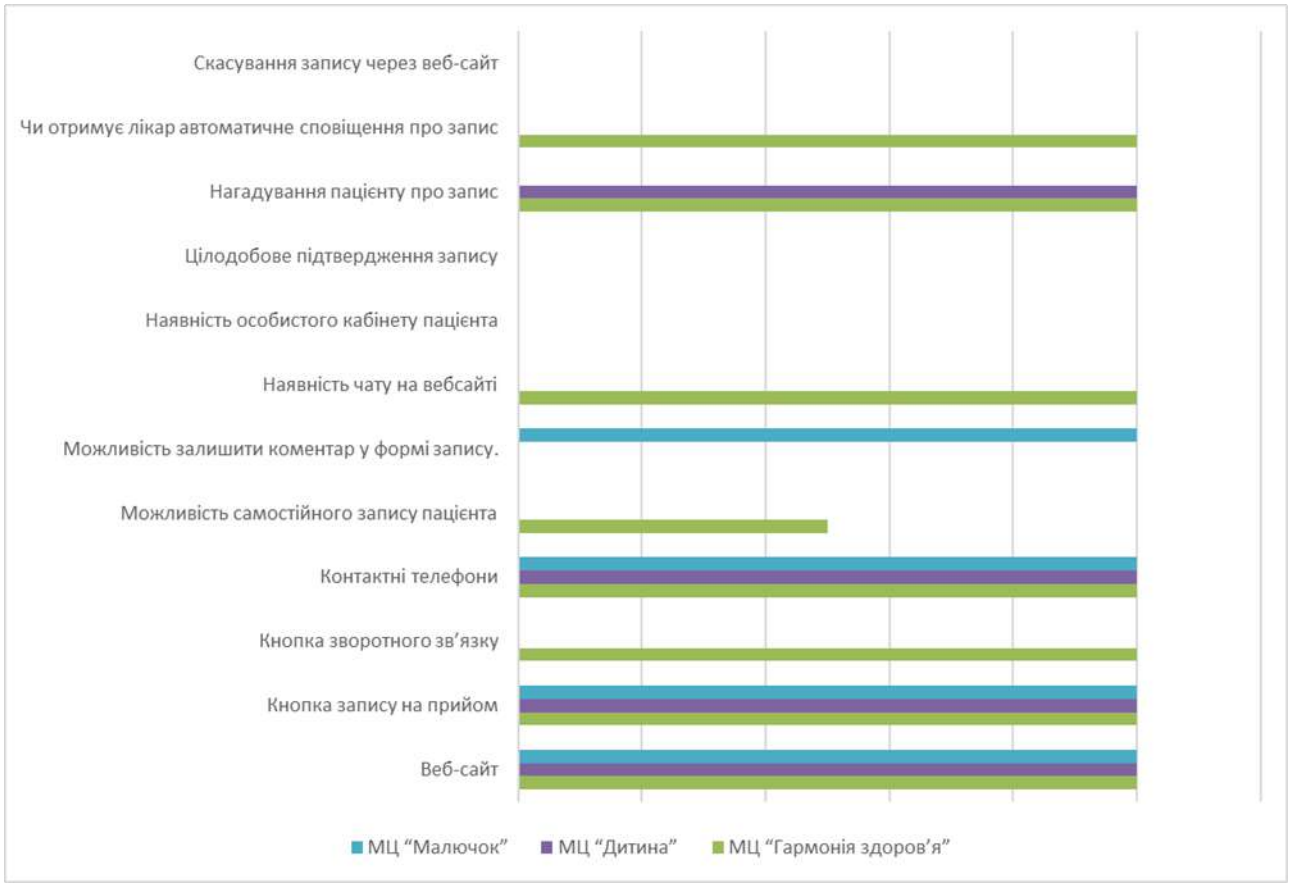

Рисунок 3.2 – Візуалізації конкурентного аналізу

Варто обґрунтувати, чому впровадження певних конкурентних пунктів не є доцільним чи можливим у випадку з МЦ “Дитина”. Перш за все, було розглянуто можливість самостійного запису пацієнта. У МЦ “Гармонія здоров’я” це реалізовано як надсилання запиту на запис, проте факт запису вимагає підтвердження клініки (рис.3.3).

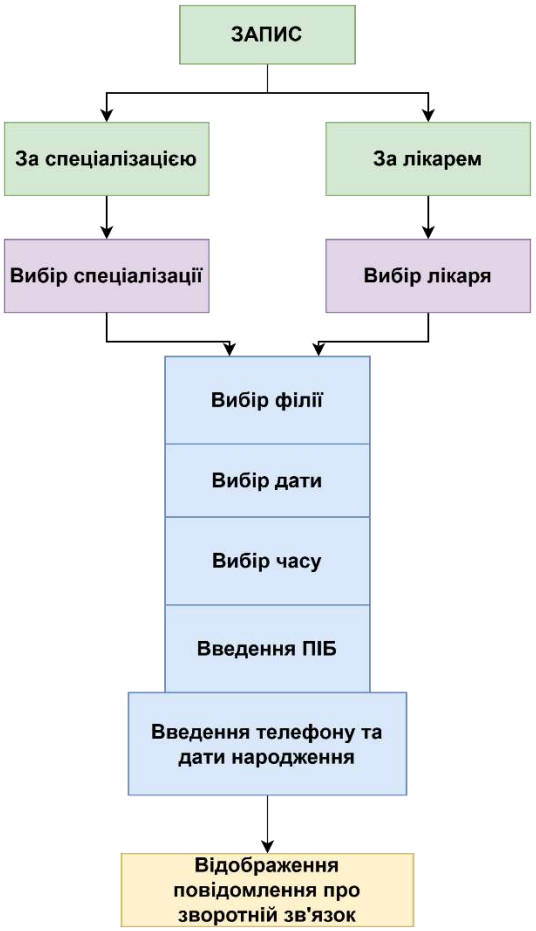

Рисунок 3.3 – Алгоритм самостійного запису до МЦ "Гармонія здоров'я"

МЦ "Дитина" також впроваджувала такий тип запису, проте він запобігав гнучкості. Різні типи звернень, як от вакцинація, консультації вимагають різної кількості часу. Також рекомендується записувати вакцинації поряд за графіком, аби оптимізувати роботу медсестри. Система самозапису унеможливлює таку гнучкість, адже пацієнти самі обирають час та дату. Більше того, лікар може бути гнучким впродовж дня та працювати у різних філіях. Прикладом може бути випадок, коли за системою самозапису пацієнт здійснив запис на 18:00 у філію А, у якій в той день було всього 2 записи о10:00 та 12:00. Тим часом у філії В не вистачало спеціалістів. Лікар, котрий міг змінити локацію на філію В, не зробив цього, адже мав пацієнта на 18:00. Внаслідок подібних незручностей такий запис було скасовано. Варто зазначити, що такий тип запису може бути ефективним для більших бізнесів, де гнучкість не є нагальною потребою.

Чат на веб-сайті є потужним інструментом у бізнесах загалом, проте він також не виявився ефективним у випадку медичного центру. Переважна

кількість надавала перевагу дзвінкам, адже так інформацію можна було отримати швидше. Тим не менше, цей віджет вимагає залучення додаткового персоналу, що наразі не є можливим для бізнесу.

Розробка особистого кабінету не є доцільною, адже вимагає значних затрат, до яких бізнес наразі не готовий.

Наступним етапом було самостійно створено As-is BPMN модель (рис 3.4) для бізнесу на основі експериментального дослідження бізнес-процесів та безпосереднім контактом з бізнесом. Для моделювання бізнес-процесів використовувалось середовище Camunda Modeler [51].

Під час аналізу моделі та на основі інформації з інтерв'ю, було зроблено певні висновки.

По-перше, бізнес-процес не передбачає ефективного інформування лікаря про запис та скасування запису. Диспетчер спершу інформує адміністратора про запис чи скасування через різні ненормовані джерела зв'язку, як от повідомлення чи дзвінок. Адміністратор передає цю інформацію лікарю у його робочий час. Це не є ані ефективним, а ні зручним, адже адміністратор та диспетчер витрачають час на інформування, а лікар не завжди отримує інформацію вчасно. До того ж, робочий день диспетчера та адміністратора є довшим, і лікар часто не отримує інформацію про запис на наступний день.

По-друге, було зазначено, що навіть після телефонних дзвінків від адміністратора з підтвердженням запису, пацієнти могли забувати про нього. Такі випадки є відносно частими, і спричиняють незручності, адже пацієнти таки приходять до клініки, проте при цьому графік зміщується.

По-третє, неефективною є переадресація певних звернень до адміністратора диспетчером. Такі звернення, зазвичай, стосуються фінансових питань, розгорнутих запитань про послуги чи роботу клініки, скарг чи підписання контрактів. Вони займають багато часу та є специфічними, тож їх має опрацьовувати оператор. На веб-сайті ніяк не ідентифіковано номер адміністратора, тож усі звернення найчастіше надходять саме диспетчерам. Це сповільнює їх роботу та може викликати незадоволення клієнтів.

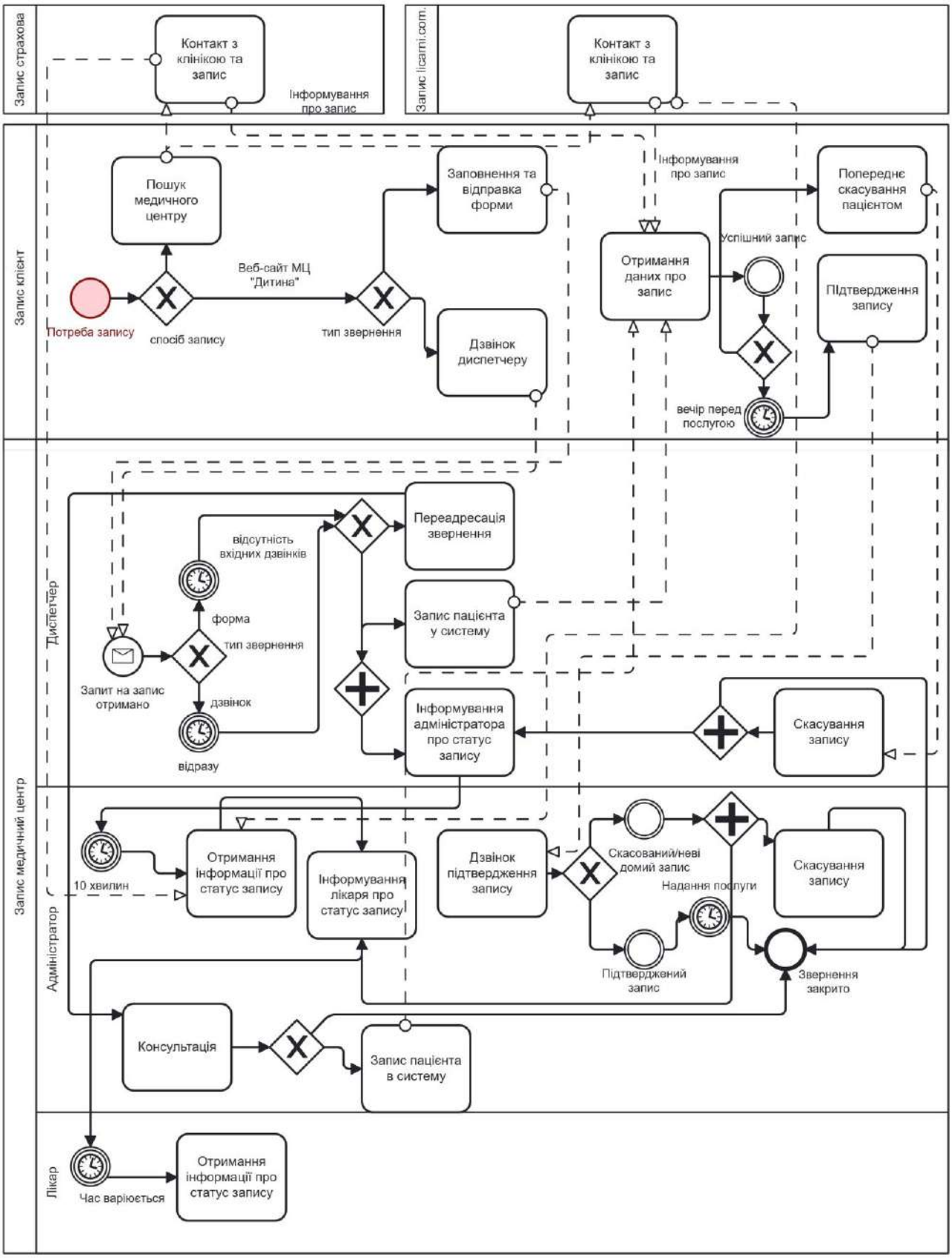

Рисунок 3.4 – As-is BPMN модель бізнес-процесу

По-четверте, бізнес не контактує з пацієнтами після їх запису та/або візиту до медичного центру. Це було пояснено тим, що найвигіднішим та найменш конфліктним є надання послуг клієнтам, що самостійно вирішують повернутися до медичного центру. Внутрішня статистика центру свідчить про те, що більше половини клієнтів звертаються повторно. Тим не менш, центр надає послуги вакцинації, де може вимагатись введення повторної дози вакцини. Навіть якщо клієнт і задоволений своїм зверненням, ця людина може забути про необхідність ревакцинації. Це може бути недоліком як для пацієнта, так і для центру.

Було побудовано експертну систему (рис 3.5) за допомогою програми GeNIe Academic, яка дозволяє оцінювати явку пацієнта на ревакцинацію. Байєсівська мережа довіри містить три вершини, і при цьому кожна з них може приймати лише один із двох можливих станів:

- Рекомендовано ревакцинацію за поточним станом здоров'я: 'рекомендовано' чи 'не_рекомендовано';
- Пацієнт забув: 'пацієнт_забув' чи 'пацієнт_не_забув';
- Пацієнт хоче: 'пацієнт_хоче' чи 'пацієнт_не_хоче';
- Пацієнт_Не_Приходить_Вчасно: 'пацієнт_приходить_вчасно' чи 'пацієнт_не_приходить_вчасно'.

Наведена на рисунку мережа довіри моделює причинно-наслідкову залежність від вершини "Не_Рекомендовано" до вершини "Пацієнт_Не_Приходить_Вчасно", від вершини "Пацієнт_Забув" до вершини " Пацієнт_Не_Приходить_Вчасно" і від вершини "Пацієнт_Не_Хоче" до вершини "Пацієнт_Не_Приходить_Вчасно".

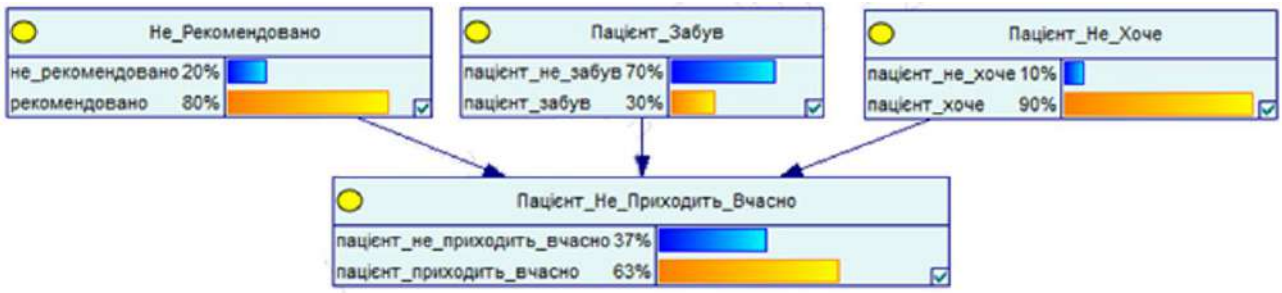

Рисунок 3.5 – Байєсівська мережа довіри у вигляді Bar Chart

Попередньо було задано умовні ймовірності на основі експертних оцінок (рис.3.6).

Не_Рекомендовано	не_рекомендовано				рекомендовано			
Пацієнт_Забув	пацієнт_не_забув		пацієнт_забув		пацієнт_не_забув		пацієнт_забув	
Пацієнт_Не_Хоче	пацієнт_не...	пацієнт_хоче	пацієнт_не...	пацієнт_хоче	пацієнт_не...	пацієнт_хоче	пацієнт_не...	пацієнт_хоче
▶ пацієнт_не_приходить_вчасно	0.9	0.15	0.99	0.95	0.9	0.01	0.99	0.95
пацієнт_приходить_вчасно	0.1	0.85	0.01	0.05	0.1	0.99	0.01	0.05

Рисунок 3.6 – Умовні ймовірності для вершини "Пацієнт_Не_Приходить_Вчасно"

Також було задано апріорні ймовірності на основі експертних оцінок (рис. 3.7, рис. 3.8, рис.3.9)

▶ пацієнт_не_хоче	0.1
пацієнт_хоче	0.9

Рисунок 3.7 – Апріорні ймовірності для вершини "Пацієнт_Не_Хоче"

▶ пацієнт_не_забув	0.7
пацієнт_забув	0.3

Рисунок 3.8 – Апріорні ймовірності для вершини "Пацієнт_Забув"

▶ не_рекомендовано	0.2
рекомендовано	0.8

Рисунок 3.9 – Апріорні ймовірності для вершини "Не_Рекомендовано"

Можна зробити висновок, що ймовірність того, що пацієнт не приходить вчасно на ревакцинацію становить 0.37 (рис 3.5)

Проте якщо задати докази та порівняти результати можна побачити, що найбільше на неявку впливає забуття пацієнтом про необхідність ревакцинації, ймовірність становитиме 0.95 (рис. 3.10).

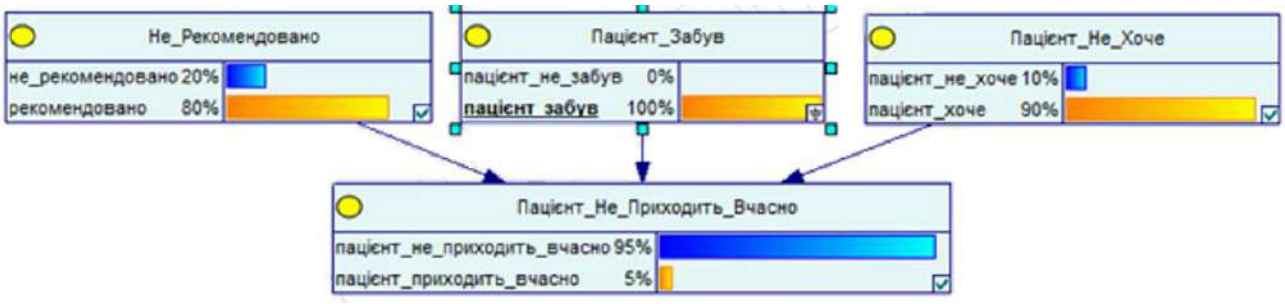

Рисунок 3.10 – Ймовірність неявки пацієнта на вчасну ревакцинацію за заданого доказу

Наступним кроком було проаналізовано елементи веб-сайту на адаптивність та доступність.

За допомогою інструментів розробника в браузері Google, було з'ясовано, що форма запису на прийом не є адаптивною до екранів менших розмірів, а також не коректно відображає поле з датою. Окрім цього не зрозуміло, чи варто вводити дату бажаного візиту, чи дату народження дитини.

Також, за допомогою інструменту визначення контрасту WCAG Color Contrast Checker [52] та інструменту EyeDropper, було з'ясовано, що контраст тексту та кольору кнопки запису на прийом, а також форми та тексту в цілому не є належним і становить 4.12:1. Це саме стосується і кнопки контакти для мобільних пристроїв. Контраст становить 2.22:1. Усі знайдені недоліки було враховано під час створення прототипів.

При цьому елементи, на яких перебував фокус відповідно виділялись. Далі було здійснено SWOT-аналіз бізнес-процесу МЦ "Дитина" (табл. 3.2).

Таблиця 3.2 – SWOT-аналіз бізнес-процесу запису на прийом.

Отже, внаслідок проведеного аналізу, було розглянуто бізнес-процес запису на послугу до медичних центрів за допомогою:

- порівняльного аналізу конкурентних бізнес-процесів;
- SWOT-аналізу;

- BPMN-моделювання;
- аналізу елементів веб-сайту на адаптивність та доступність;
- урахування внутрішніх вимог та викликів бізнесу.

Наступим етапом можна визначити задачі редизайну.

Першою задачею є додання до бізнес-процесу автоматизованої розсилки смс-повідомлень спеціалісту, що повідомлятиме про факт запису на прийом, а також нестиме ключову інформацію про пацієнта, таку як прізвище та ім’я пацієнта, вік пацієнта, причину звернення, дату та час звернення, а також коментар від диспетчера. Також надсилання повідомлення за умови скасування запису. Це дозволить розвантажити адміністраторів та дозволить лікарю планувати робочий час ефективніше. Було створено прототипи таких повідомлень (рис. 3.12, рис. 3.13).

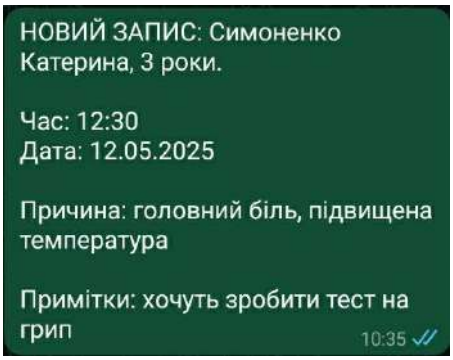

Рисунок 3.12 – Прототипи повідомлення спеціалісту про новий запис.

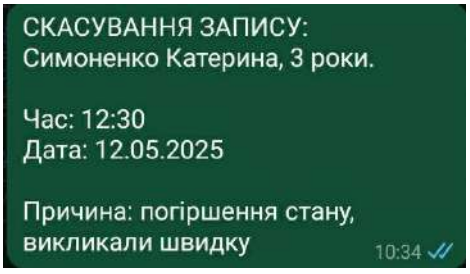

Рисунок 3.13 – Прототипи повідомлення спеціалісту про скасування запису.

Наступною задачею стало додання до бізнес-процесу автоматизованої розсилки SMS-нагадування пацієнтам про запис на послуги в медичному центрі, за винятком домашніх консультацій. Такі SMS-повідомлення пропонується надсилати вранці у день проведення послуги, як доповнення до телефонних дзвінків у попередній день. Якщо пацієнт записаний на вакцинацію, SMS

міститиме застереження, що у випадку наявності очевидних симптомів захворювання вакцинацію не може бути проведено. Натомість, запропонувати прийти на огляд чи змінити дату запису на вакцинацію (рис. 3.14, рис. 3.15).

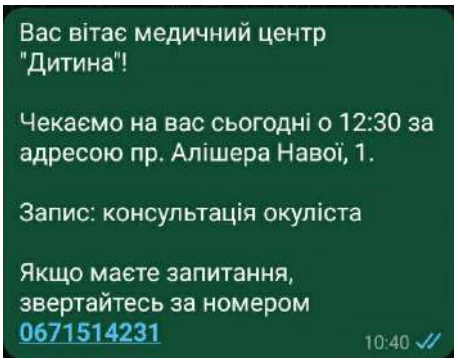

Рисунок 3.14 – Прототип повідомлення-нагадування пацієнту про запис

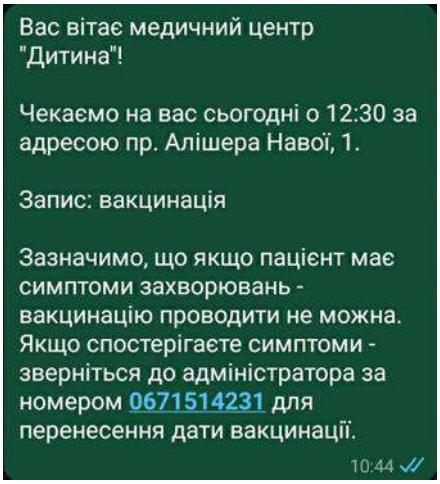

Рисунок 3.15 – Прототип повідомлення-нагадування пацієнту про запис (вакцинація)

Іще однією задачею є додання до бізнес-процесу надсилання SMS-повідомлень, якщо пацієнту було проведено вакцинацію, що потребує повторної дози. Було визначено за допомогою моделювання Байєсівської мережі, що забуття пацієнтом про вакцинацію найбільше впливає на явку.

SMS має мати форму нагадування про потребу введення повторної дози та пропозицію звернення до клініки. Повідомлення пропонується надсилати за два тижні до орієнтовної дати вакцинації, аби врахувати час на замовлення клінікою вакцини. Повідомлення не має бути автоматичними, адже замовлення вакцин є індивідуальним та нестабільним процесом, що залежить від багатьох факторів, як от наявність вакцини на ринку, доцільність замовлення (якщо доза

розрахована на кількох людей) та особливості кількості дози для кожного пацієнта. Повідомлення надсилає адміністратор після узгодження з лікарем. Було створено прототип повідомлення (рис.3.16).

Рисунок 3.16 – Прототип повідомлення-нагадування пацієнту про ревакцинацію

Пропонується додати до веб-сайту кнопку зворотного зв’язку з адміністратором клініки. Згідно з дослідженням ResponseTap [53], використання кнопки зворотного виклику, збільшує показник конверсії до 22%. Також пропонується використати цю кнопку як засіб оптимізації роботи диспетчера, який зможе приділити більше часу на обробку типових дзвінків та зменшити час на переадресацію. Було створено прототип кнопки та відповідної форми (рис. 3.17).

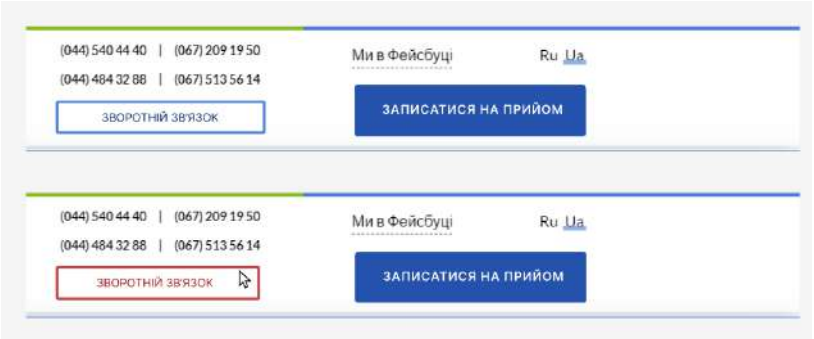

Рисунок 3.17 – Прототип кнопки зворотного зв’язку веб-версія

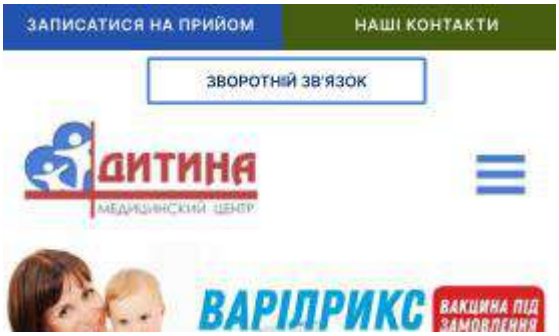

Рисунок 3.18 – Прототип кнопки зворотного зв'язку (мобільна версія)

При створені форми розглядалось наповнення форми на конкурентному веб-сайті "Гармонія здоров'я" (рис 3.19).

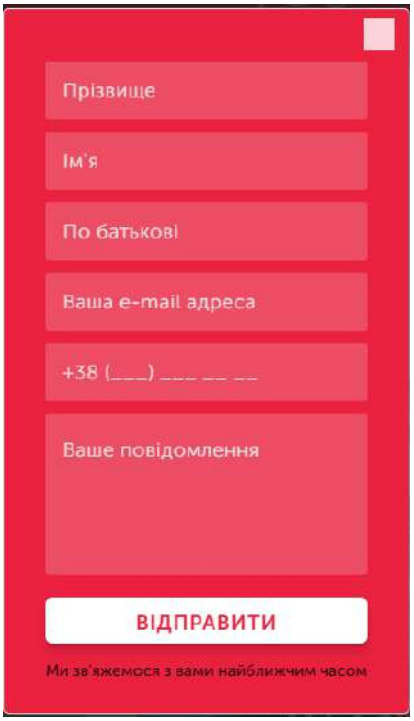

Рисунок 3.19 – Форма зворотного зв'язку клініки "Гармонія здоров'я"

Було визначено, що поля "По батькові", "Ваша e-mail адреса" є недоцільними у випадку МЦ "Дитина", адже не використовуються бізнесом. Було вирішено зробити більший акцент повідомлення про зв'язок (рис 3.20).

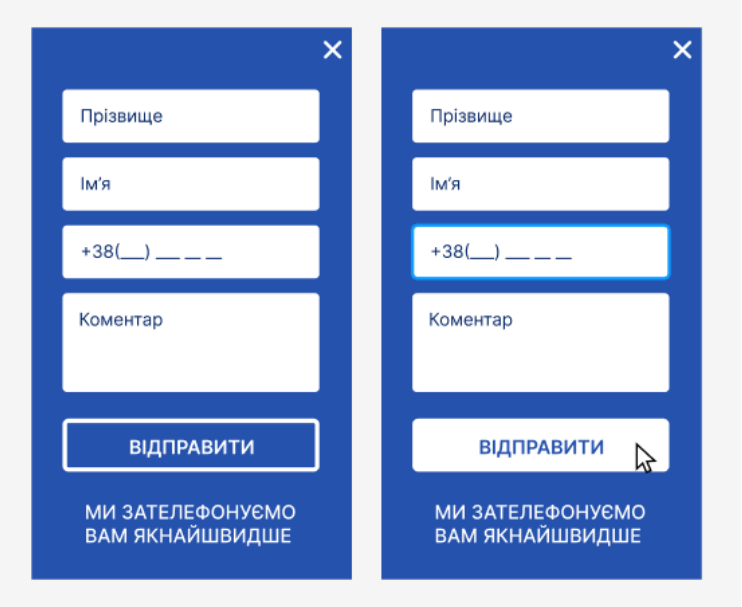

Рисунок 3.20 – Прототип кнопки форми зв'язку

Пропонується модифікувати форму запису на веб-сайті, додавши поле для опційного введення коментарів. Це оптимізує розмову з диспетчером, адже він матиме додаткову інформацію про звернення. Було створено прототипи такої форми (рис. 3.21, рис. 3.22).

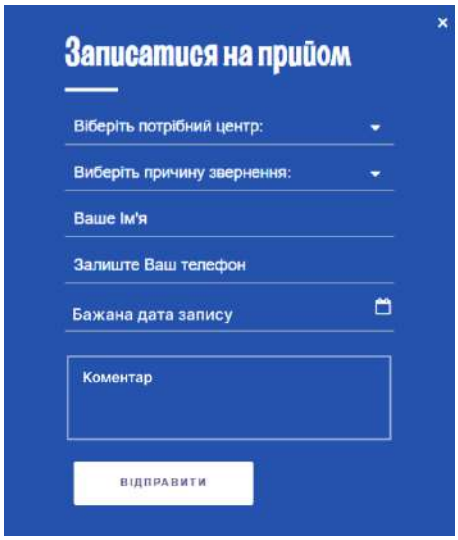

Рисунок 3.21 – Прототип модифікованої форми запису (веб-версія)

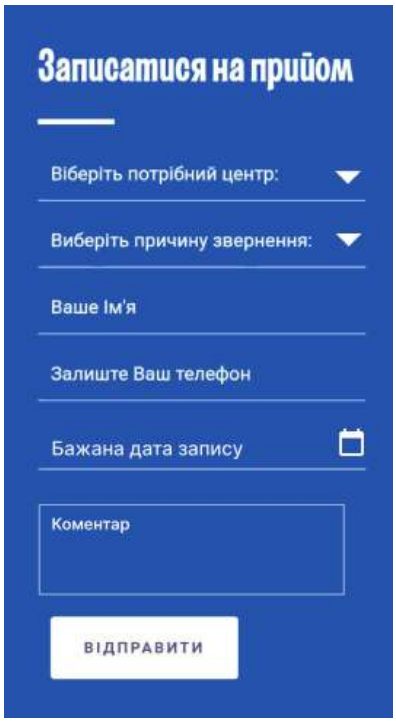

Рисунок 3.22 – Прототип форми запису (мобільна версія)

Також було враховано неефективність переадресації звернень та відповідно змінено бізнес-процес.

Насамкінець, було створено To-be BPMN модель для бізнес-процесу, яка ілюструє зміни (рис. 3.23).

Варто зазначити, що Camunda Modeler, що використовувався для створення моделі є загалом зручним та підходящим інструментом. Він має зрозумілий інтерфейс, а також усі необхідні елементи для створення моделі. Серед недоліків можна виділити недостаню кастомізацію окремих елементів (як от збільшення ширини заголовка доріжки чи малий вибір кольорів), а також незручна зміна розмірів доріжки, що не враховувала наявність елементів всередині. Через це доводилось вручну перетягувати елементи кожного разу, коли виникала необхідність змінити розмір доріжки, що сповільнювало роботу.

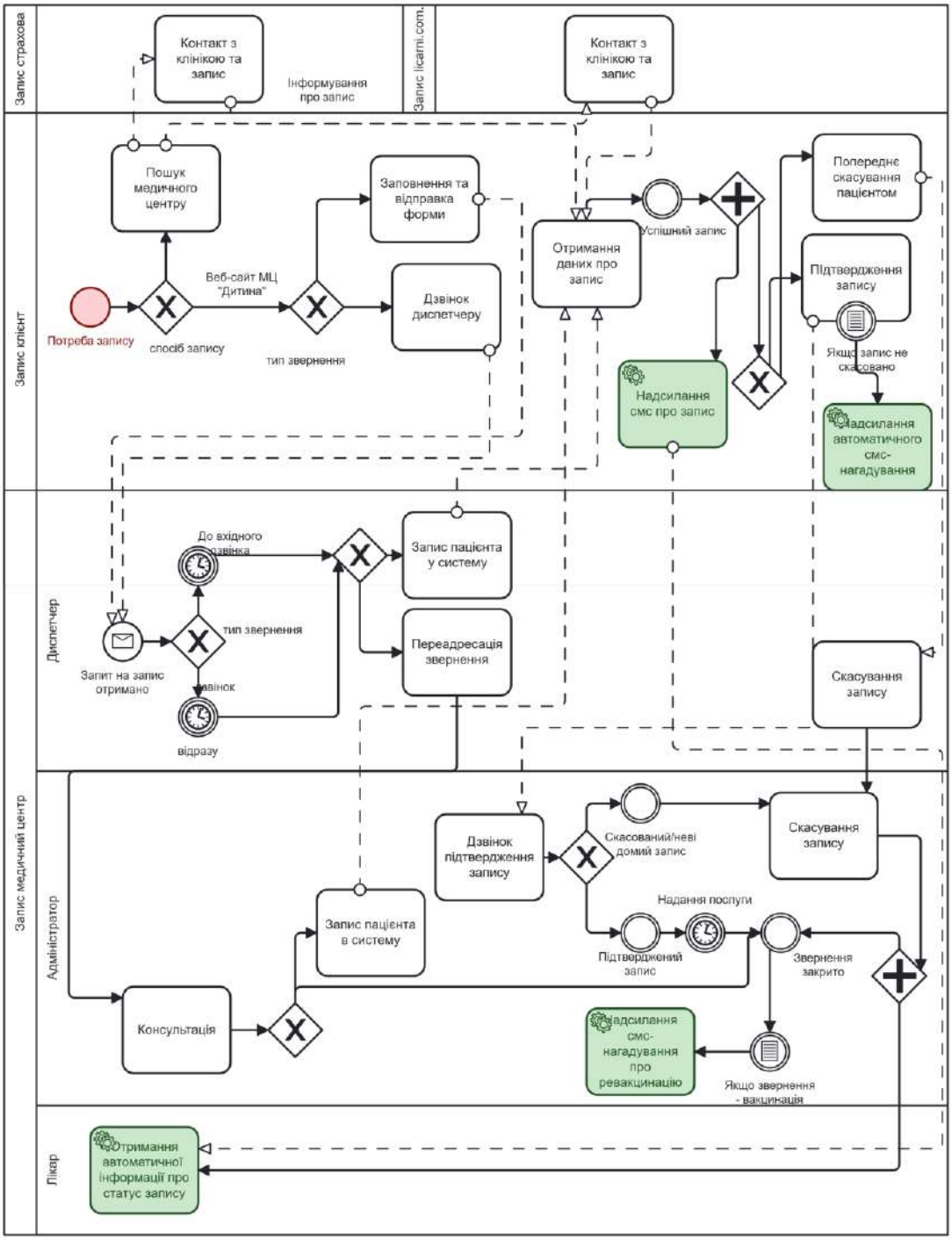

Рисунок 3.23 – To-be BPMN модель бізнес-процесу

3.3 Вибір методології впровадження та формування плану реінжинірингу

У першому розділі було розглянуто та порівняно методи реінжинірингу бізнес-процесів. Було виокремлено недоліки, особливості, застосовність до типів бізнесів, повноту та кількість етапів. Здійснений порівняльний аналіз було використано для прийняття рішення методом аналізу ієрархій.

Спершу було побудовано трирівневу ієрархію (рис. 3.24):

1. **Головна мета** – обрати методологію реінжинірингу.
2. **Критерії вибору** (методології реінжинірингу) – кількість етапів, повнота, тип бізнесу, критичність недоліків, критичність головної особливості.
3. **Альтернативи** – Шість сігм, Девенпорта, Гаммера та Чемпі, точка зупину, швидкий реінжиніринг, підприємницька, життєвого циклу.

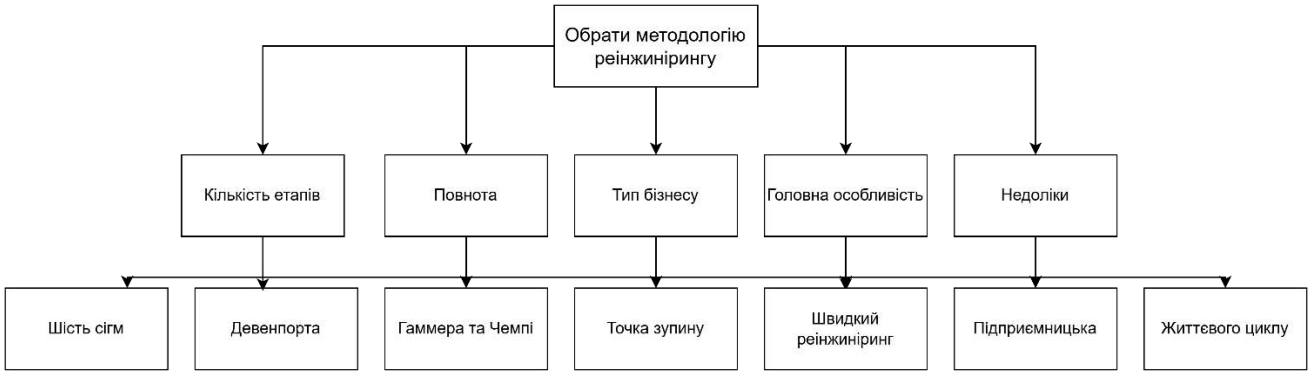

Рисунок 3.24 – Трирівнева ієрархія

Для обчислення компонентів власного вектору використовувалось середнє геометричне.

Таблиця 3.3. Власний, нормований та вектор глобальних пріоритетів на основі матриці попарних порівнянь

	Повнота	*Кількість етапів*	*Тип бізнесу*	*Недоліки*	*Головна особливість*	*Компоненти власного вектору*	*Нормований вектор пріоритетів*	*Вектор глобальних пріоритетів*
Повнота	1	5	3	5	5	3.27	0.47	0.9
Кількість етапів	0.2	1	0.2	0.2	0.2	0.27	0.04	0.84
Тип бізнесу	0.33	5	1	3	3	1.7	0.25	1.22
Недоліки	0.2	5	0.33	1	1	0.8	0.12	1.22
Головна особливість	0.2	5	0.33	1	1	0.8	0.12	1.22
Сума	1.93	21	4.86	10.2	10.2	6.84	1	5.4

ІУ = (5.4-5)/4 = 0.1

ВУ = 0.1/1.11 = 0.09

Найбільш пріоритетним є критерій повноти.

Таблиця 3.4. Розрахунки за критерієм «Кількість етапів»

Кількість етапів	*Шість сігм*	*Девенпорта*	*Гаммера та Чемпі*	*Точка зупину*	*Швидкий реінжиніринг*	*Підприємницька*	*Життєвого циклу*	*Компоненти власного вектору*	*Нормований вектор пріоритетів*	*Вектор глобальних пріоритетів*
Шість сігм	1	2	1	2	1	2	2	1.49	0.18	1.26
Девенпорта	0.5	1	0.5	1	0.5	1	1	0.74	0.09	0.9
Гаммера та Чемпі	1	2	1	2	1	2	2	1.49	0.18	1.26
Точка зупину	0.5	1	0.5	1	0.5	1	1	0.74	0.09	0.9
Швидкий реінжиніринг	1	2	1	2	1	2	2	1.49	0.18	1.26
Підприємницька	0.5	1	0.5	1	0.5	1	1	0.74	0.09	0.9
Життєвого циклу	0.5	1	0.5	1	0.5	1	1	1.49	0.18	1.26
Сума	7	10	7	10	7	10	10	8.18	1	7.74

ІУ = (7.74 – 7)/6 = 0.123

ВУ = 0.123/1.35 = 0.091

Найбільш пріоритетними є шість сігм, Гаммера та Чемпі, Швидкий реінжиніринг, життєвого циклу.

Таблиця 3.5. Розрахунки за критерієм «Повнота»

Повнота	*Шість сігм*	*Девенпорта*	*Гаммера та Чемпі*	*Точка зупину*	*Швидкий реінжиніринг*	*Підприємницька*	*Життєвого циклу*	*Компоненти власного вектору*	*Нормований вектор пріоритетів*	*Вектор глобальних пріоритетів*
Шість сігм	1	3	1	1	1	1	1	1.17	0.16	1.01
Девенпорта	0.33	1	0.33	0.33	0.33	0.33	0.33	0.26	0.04	0.76
Гаммера та Чемпі	1	3	1	1	1	1	1	1.17	0.16	1.01
Точка зупину	1	3	1	1	1	1	1	1.17	0.16	1.01
Швидкий реінжиніринг	1	3	1	1	1	1	1	1.17	0.16	1.01
Підприємницька	1	3	1	1	1	1	1	1.17	0.16	1.01
Життєвого циклу	1	3	1	1	1	1	1	1.17	0.16	1.01
Сума	6.33	19	6.33	6.33	6.33	6.33	6.33	7.28	1	6.82

ІУ = |6.82 – 7|/6 = 0.03

ВУ = 0.03/1.35 = 0.02

Найменш пріоритетним є Девенпорта.

Таблиця 3.6. Розрахунки за критерієм «Тип бізнесу»

Тип бізнесу	*Шість сігм*	*Девенпорта*	*Гаммера та Чемпі*	*Точка зупину*	*Швидкий реінжиніринг*	*Підприємницька*	*Життєвого циклу*	*Компоненти власного вектору*	*Нормований вектор пріоритетів*	*Вектор глобальних пріоритетів*
Шість сігм	1	3	1	1	1	5	5	1.85	0.21	0.99
Девенпорта	0.33	1	0.33	0.33	0.33	0.5	0.5	0.44	0.05	0.85
Гаммера та Чемпі	1	3	1	1	1	5	5	1.85	0.21	0.99
Точка зупину	1	3	1	1	1	5	5	1.85	0.21	0.99
Швидкий реінжиніринг	1	3	1	1	1	5	5	1.85	0.21	0.99
Підприємницька	0.2	2	0.2	0.2	0.2	1	1	0.44	0.05	1.13
Життєвого циклу	0.2	2	0.2	0.2	0.2	1	1	0.44	0.05	1.13
Сума	4.73	17	4.73	4.73	4.73	22.5	22.5	8.72	1	7.07

ІУ = (7.07–7)/6= 0.012

ВУ = 0.012/1.35 = 0.007

Найбільш пріоритетними є шість сігм, Гаммера та Чемпі, точка зупину, швидкий реінжиніринг.

Таблиця 3.7. Розрахунки за критерієм «Головна особливість»

Головна особливість	*Шість сігм*	*Девенпорта*	*Гаммера та Чемпі*	*Точка зупину*	*Швидкий реінжиніринг*	*Підприємницька*	*Життєвого циклу*	*Компоненти власного вектору*	*Нормований вектор пріоритетів*	*Вектор глобальних пріоритетів*
Шість сігм	1	5	0.33	5	0.33	5	5	1.83	0.20	1.56
Девенпорта	0.2	1	0.2	1	0.2	1	1	0.50	0.06	1.14
Гаммера та Чемпі	3	5	1	3	1	3	3	2.36	0.26	0.09
Точка зупину	0.2	1	0.33	1	0.33	1	1	0.58	0.06	0.9
Швидкий реінжиніринг	3	5	1	3	1	5	5	2.73	0.3	0.98
Підприємницька	0.2	1	0.33	1	0.2	1	1	0.54	0.06	1.02
Життєвого циклу	0.2	1	0.33	1	0.2	1	1	0.54	0.06	1.02
Сума	7.8	19	3.52	15	3.26	17	17	9.08	1	6.71

ІУ = |6.71–7|/6= 0.05

ВУ = 0.05/1.35 = 0.03

Найбільш пріоритетними є шість сігм, Гаммера та Чемпі, швидкий реінжиніринг.

Таблиця 3.8. Розрахунки за критерієм «Недоліки»

Недоліки	*Шість сігм*	*Девенпорта*	*Гаммера та Чемпі*	*Точка зупину*	*Швидкий реінжиніринг*	*Підприємницька*	*Життєвого циклу*	*Компоненти власного вектору*	*Нормований вектор пріоритетів*	*Вектор глобальних пріоритетів*
Шість сігм	1	5	0.33	3	0.2	1	1	1	0.13	1.15
Девенпорта	0.14	1	0.14	0.14	0.14	0.14	0.14	0.19	0.02	0.62
Гаммера та Чемпі	3	5	1	3	0.33	1	1	1.47	0.18	1.22
Точка зупину	0.33	5	0.33	1	0.2	1	1	0.72	0.09	1.27
Швидкий реінжиніринг	5	5	3	5	1	5	5	3.69	0.46	1.29
Підприємницька	1	5	1	1	0.14	1	1	0.95	0.11	1.11
Життєвого циклу	1	5	1	1	0.14	1	1	0.95	0.11	1.11
Сума	11.47	31	6.8	14.14	2.15	10.14	10.14	7.97	1	7.77

ІУ = |7.77-7|/6=0.13

ВУ = 0.13/1.35 = 0.96

Найбільш пріоритетним є швидкий реінжиніринг.

Таблиця 3.9. Розрахунки глобальних пріоритетів альтернатив

Критерії/ альтернативи	*Повнота*	*Кількість етапів*	*Тип бізнесу*	*Недоліки*	*Головна особливість*	*Рейтинг альтернатив*
	0.47	*0.04*	*0.25*	*0.12*	*0.12*	
Шість сігм	0.16	0.18	0.21	0.13	0.20	**0.17**
Девенпорта	0.04	0.09	0.05	0.02	0.06	**0.04**
Гаммера та Чемпі	0.16	0.18	0.21	0.18	0.26	**0.19**
Точка зупину	0.16	0.09	0.21	0.09	0.06	**0.15**
Швидкий реінжиніринг	0.16	0.18	0.21	0.46	0.3	**0.22**
Підприємницька	0.16	0.09	0.05	0.11	0.06	**0.11**
Життєвого циклу	0.16	0.18	0.05	0.11	0.06	**0.11**

1. 0.47*0.16 + 0.04*0.18 + 0.25*0.21 + 0.12*0.13 + 0.12*0.2= 0.17
2. 0.47*0.04 + 0.04*0.09 + 0.25*0.05 + 0.12*0.02 + 0.12*0.06= 0.04
3. 0.47*0.16+ 0.04*0.18 + 0.25*0.21 + 0.12*0.18 + 0.12*0.26= 0.19

4. 0.47*0.16+ 0.04*0.09 + 0.25*0.21 + 0.12*0.09 + 0.12*0.06= 0.15
5. 0.47*0.16 + 0.04*0.18 + 0.25*0.21 + 0.12*0.46 + 0.12*0.3= 0.22
6. 0.47*0.16 + 0.04*0.09 + 0.25*0.05 + 0.12*0.11 + 0.12*0.06= 0.11
7. 0.47*0.16+ 0.04*0.18 + 0.25*0.05 + 0.12*0.11 + 0.12*0.06= 0.11

Для усіх матриць значення ВУ < 0,1.

Виходячи з аналізу, було визначено, що найбільш підходящим методом є метод швидкого реінжинірингу. Цей метод є універсальним для всіх типів бізнесу, при цьому фокусується на стратегічно важливих аспектах та не вимагає повного переосмислення бізнес-моделі. При цьому, метод швидкого реінжинірингу описує всі необхідні етапи для проведення реінжинірингу. У випадку з МЦ “Дитина” було виокремлено бізнес-процес запису на прийом, який можна вважати ізольованим, і який не вимагає внесення змін до інших процесів.

Власне метод аналізу ієрархій, що використовувався для обрання методології є зрозумілим, зручним та обґрунтованим підходом, проте внаслідок його застосування, можна зробити висновок, що він є безумовно заскладним для обраного об’єкту. Альтернативними, більш доцільними методами, можуть бути метод виключення, що полягає в послідовному виключенні альтернатив або метод PMI, який полягає у фіксації плюсів та мінусів та подальшої кількісної або якісної оцінки.

Отже, у цьому розділі було сформовано план реінжинірингу бізнес-процесу та представлено план (рис. 3.25) за допомогою діаграми Ганта [54]. Це гістограма, що використовується для планування проєкту та відстеження прогресу. План реінжинірингу формувався, базуючись на обраній методології швидкого реінжинірингу. Також було виокремлено основних учасників процесу реінжинірингу:

- Керівництво медичного центру
- Адміністратор медичного центру
- Диспетчер медичного центру
- IT-менеджер медичного центру

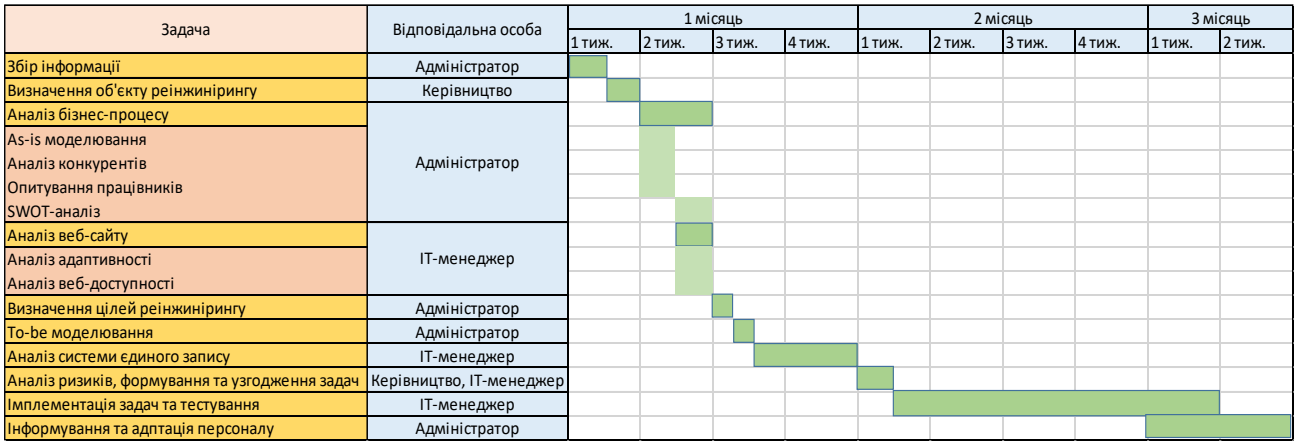

Рисунок 3.25 – План реінжинірингу бізнес-процесу.

3.4 Аналіз ризиків реінжинірингу обраного бізнес-процесу

Насамкінець, було ідентифіковано (табл. 3.10), проаналізовано можливі ризики реінжинірингу бізнес-процесу та запропоновано заходи оптимізації.

Таблиця 3.10. Ідентифікація ризиків для реінжинірингу бізнес процесу запису на прийом

Види ризиків	**Характеристика**
Технічні ризики	Можуть бути пов'язані з проблемами зі створенням, тестуванням та впровадженням інтеграції SMS-повідомлень. Наприклад, несумісність з системою, незручність у використанні, помилки в програмному забезпеченні тощо
Ризики пов'язані з людським фактором	Можлива невчасна імплеметація задач через людський фактор
Ризики щодо потреб користувачів	Можливе незадоволення пацієнтів розсилкою повідомлень, що можуть сприйматися як «надокучливі»
Інноваційні ризики	Можливий повільна адаптація працівників до роботи згідно з новими процесами та, як наслідок, організаційні інциденти під час роботи
Ризики управління проектом	Можливі проблеми з плануванням, координацією та керуванням проектом, затримками у розробці, що може призвести до зміни планових термінів. Наприклад, затримки в графіку чи недостатній контроль над виконанням завдань
Ризик надмірної пріоритизації реінжинірингу	Можливі затримки у інших аспектах роботи медичного закладу через надмірну пріоритизацію

Варто зазначити, що ризики є невід'ємною частиною кожного проєкту, тож важливо їх враховувати та вживати необхідні заходи. Далі було ідентифіковано та згруповано чинники ризиків (рис. 3.26) за рівнями їх управління та утворення.

Рисунок 3.26 – Групування чинників ризику за рівнями управління та утворення

Далі було проведено аналіз ризиків методом експертних оцінок. Узгодженість відповідей експертів було оцінено за допомогою коефіцієнта конкордації, що обчислюється за формулою:

$$W = \frac{\sigma_{\phi}^2}{\sigma_{max}^2} = \frac{\sum_{i=1}^{m}\{a_i - \frac{1}{2}*n*(m+1\}^2}{\frac{1}{2}*n^2*m*(m^2-1)}.$$

Було використано власні експертні оцінки (Екс.1), а також експертні оцінки адміністратора (Екс. 2) та IT-менеджера (Екс. 3) (табл. 3.11). Також було візуалізовано оцінки (рис. 3.27).

Таблиця 3.11. Оцінювання чинників виникнення ризиків методом експертних оцінок

Чинник	**Екс 1.**	**Екс 2.**	**Екс. 3**
Ч1	2	2	2
Ч2	1	2	1
Ч3	3	4	4
Ч4	5	4	4
Ч5	5	5	6
Ч6	7	6	7
Ч7	1	1	1
Ч8	4	5	3

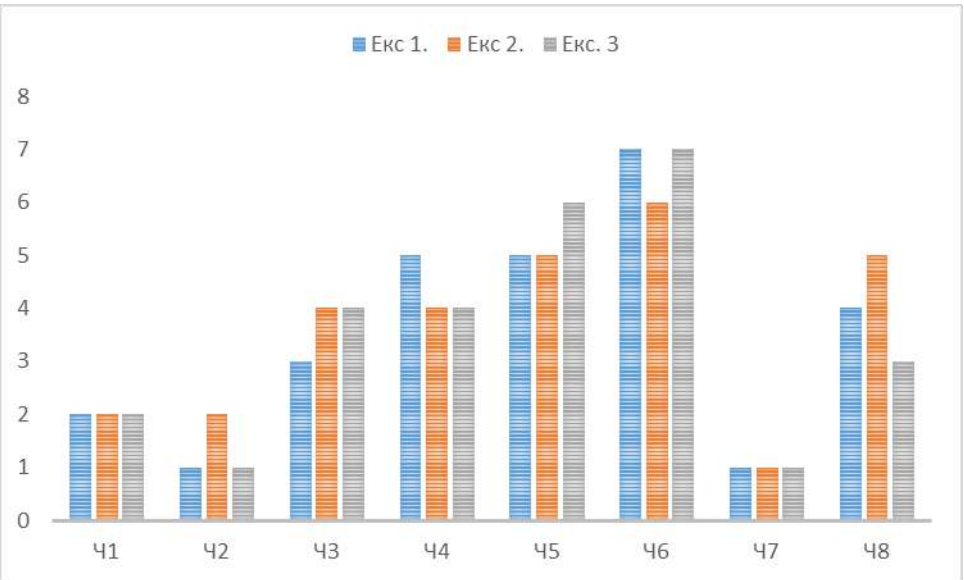

Рисунок 3.27 – Візуалізація оцінювання чинників виникнення ризиків методом експертних оцінок

$W = ((6 – 13.5)^2 + (4 – 13.5)^2 + (11 – 13.5)^2 + (13 – 13.5)^2 + (16 – 13.5)^2 + (20 – 13.5)^2 + (3 – 13.5)^2 + (12 – 13.5)^2) / 378 = 0.83$

Таким чином, коефіцієнт конкордації становить 0.83

Істотність коефіцієнту конкордації було перевірено за допомогою критерію Пірсона за формулою $\chi^2 = W \cdot n \cdot (m - 1)$. Таким чином, χ^2 становить 17.43. Можна сказати, що оцінки експертів істотно узгоджені.

Наступним кроком було складено рейтинг чинників ризиків (рис. 3.28) з використанням Min-Max нормалізації (табл. 3.12), де для кожного експерта (мін = 1, макс = 8). Було використано формулу $X' = \frac{X' - X_{min}}{X_{max} - X_{min}}$

Таблиця 3.12. Min-Max нормалізація

Чинник	Екс 1	Екс 2	Екс 3	Середнє
Ч1	0.14	0.14	0.14	0.14
Ч2	0.00	0.14	0.00	0.05
Ч3	0.29	0.43	0.43	0.38
Ч4	0.57	0.43	0.43	0.48
Ч5	0.57	0.57	0.71	0.62
Ч6	0.86	0.71	0.86	0.81
Ч7	0.00	0.00	0.00	0.00
Ч8	0.43	0.57	0.29	0.43

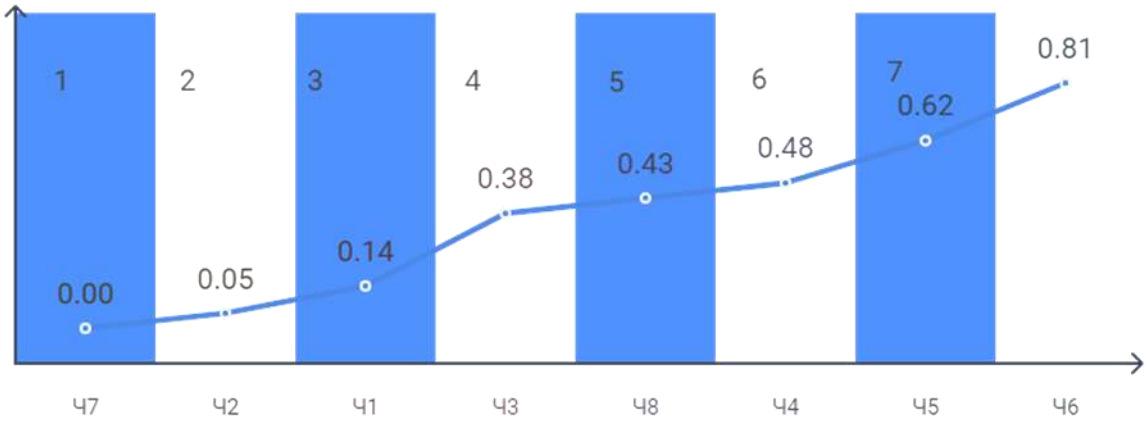

Рисунок 3.28 – Візуалізація рейтингу чинників ризиків

Можна зробити висновок, що розглянутий метод експертних оцінок аналізу ризиків є зрозумілим та застосовним до реальних ситуацій. Результати аналізу легко візуалізувати та використати для прийняття рішень. Метод є оптимальним вибором для аналізу ризиків.

Альтернативно, було додано експерні оцінки ймовірності виникнення того чи іншого чинника, обчислено їх середнє значення, як і середнє значення наданих оцінок (табл. 3.13) та на основі отриманих результатів сформовано рейтинг (рис. 3.29).

Таблиця 3.13. Оцінювання чинників виникнення ризиків кількісним методом

Чинник	**Екс 1.**	**Екс 2.**	**Екс. 3**	**Оцінка середнє**	**Р 1**	**Р 2**	**Р 3**	**Р середнє**	**Кінцеве значення**
Ч1	2	2	2	*2*	0.025	0.025	0	*0.017*	0.03
Ч2	1	2	1	*1.33*	0.1	0.2	0.2	*0.017*	0.02
Ч3	3	4	4	*3.67*	0.2	0.2	0.1	*0.017*	0.06
Ч4	5	4	4	*4.33*	0.1	0.1	0.1	*0.1*	0.43
Ч5	5	5	6	*5.33*	0.1	0.08	0.1	*0.09*	0.48
Ч6	7	6	7	*6.66*	0.06	0.08	0.1	*0.08*	0.53
Ч7	1	1	1	*1*	0.005	0.005	0.005	*0.005*	0.005
Ч8	4	5	3	*4*	0.05	0.1	0.1	*0.08*	0.32

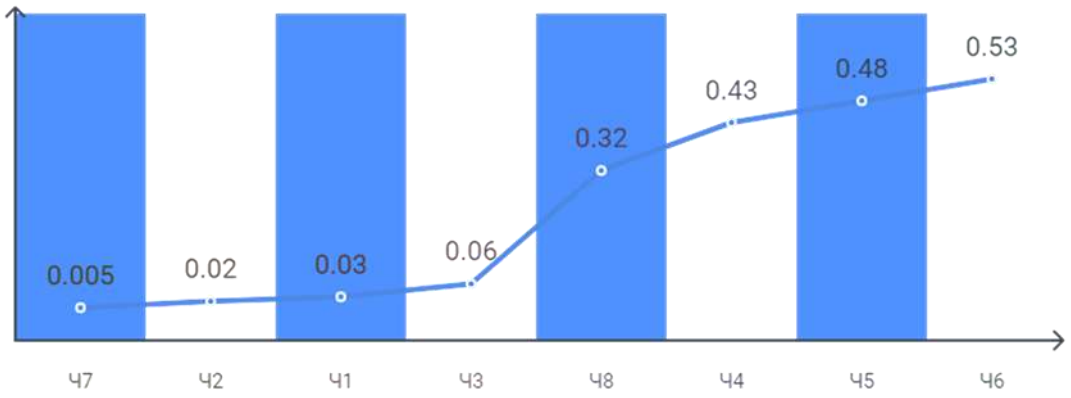

Рисунок 3.29 – Візуалізація рейтингу чинників ризиків

Можна зробити висновок, що отриманий рейтинг співпадає з попереднім аналізом. Найбільший вплив має чинник «Недостатня кваліфікація персоналу», а найменший – «Негативний вплив зовнішніх умов». Також значення ризиків за кожним з чинників не є високим, а отже реінжиніринг може бути реалізовано.

Наступним кроком було сформовано заходи оптимізації (табл. 3.14).

Таблиця 3.14. Заходи оптимізації ризиків

Види ризиків	**Заходи оптимізації ризиків**
Технічні ризики (Ч5, Ч1)	Вибір перевірених технологій; можливість видалити час на навчання
Ризики пов'язані з людським фактором (Ч7)	З урахуванням того, що проєкт не є критично-важливим для бізнесу: можливість тимчасово зупинити проєкт.
Ризики щодо потреб користувачів (Ч8)	Проведення вибіркових опитувань
Інноваційні ризики (Ч1)	Обґрунтування необхідності змін, проведення необхідного навчання; побудова сприятливого психологічного клімату в команді
Ризики управління проектом (Ч4)	Регулярна звітність керівництву
Ризик надмірної пріоритизації реінжинірингу (Ч3)	Дотримання термінів проєкту; можливе надання переваги більш критичним процесам за необхідності.

У разі виникнення чинників 2 та 6, прийнятною буде тимчасова зупинка процесу, з урахуванням того, що проєкт не є критично-важливим для бізнесу

ВИСНОВКИ

У ході виконання даної роботи було досліджено технології, методи та засоби реінжинірингу бізнес-процесів і програмних систем. Як наслідок, було виявлено важливість проведення реінжинірингу для оптимізації діяльності організацій. Було з’ясовано необхідність ретельного підходу до впровадження змін, аби уникнути надмірного чи некоректного реніжинірингу.

Порівняльний аналіз різних методологій реінжинірингу показав, що вибір підходу залежить від особливостей організації та потреб. Наприклад, методології, орієнтовані на системні зміни, можуть бути більш ефективними в складних і великих компаніях, тоді як інші краще підійдуть для малих і середніх бізнесів. У процесі створення класифікації моделей бізнес-процесів було визначено, що існують різні підходи до моделювання, кожен з яких має свої переваги та обмеження.

У рамках дослідження ключового бізнес-процесу медичного центру «Дитина» було здійснено редизайн, що дозволяє значно покращити зручність надання послуг. Аналіз веб-сайту на доступність та адаптивність також дозволив здійснити редизайн елементів, що може стати основою комфортного користувацького досвіду. Завершальним етапом стало визначення методології для планування реінжинірингу, формування плану реінжинірингу ідентифікація, аналіз можливих ризиків реінжинірингу бізнес-процесу та формування заходів оптимізації. Разом з цим було розглянуто відповідні методи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. І-Чан Чень. Empirical modelling for participative business process reengineering: дис. доктора філософських наук : 12.2001 / І-Чан Чень. — Ковентрі, 2001. – 292 с.
2. What is Business Process Reengineering (BPR)? [Електронний ресурс] – Режим доступу до ресурсу: https://creately.com/guides/what-is-business-process-reengineering/
3. Гнилянська О. В. Переосмислення бізнес-процесу через реінжиніринг / О. В. Гнилянська // Вісник Харківського національного університету імені В.Н. Каразіна. — 2022. — №.103. — С. 84—90.
4. Popular Methodologies of Business Process Reengineering [Електронний ресурс] – Режим доступу до ресурсу: https://www.tutorialspoint.com/popular-methodologies-of-business-process-reengineering
5. OKR Templates [Електронний ресурс] – Режим доступу до ресурсу: https://okrtemplates.com/
6. Tableau [Електронний ресурс] – Режим доступу до ресурсу: https://www.tableau.com/products/tableau
7. The Prosci Change Triangle (PCT) Model [Електронний ресурс] – Режим доступу до ресурсу: https://www.prosci.com/methodology/pct-model
8. What is business process reengineering (BPR)? [Електронний ресурс] – Режим доступу до ресурсу: https://www.ibm.com/topics/business-process-reengineering
9. Measuring the impact of customer satisfaction on business profitability: an empirical study / Дімітріс Дросос, Міхаліс Скордуліс // International Journal of Technology Marketing. — 2019. — Т. 13, №. 2. — С. 142—155
10. Янь Чжисянь, Маццара Мануель, Цимпіан Емілія, Урбанец Олександр. Business Process Modeling: Classification and Perspective [Електронний ресурс] / Чжисянь Янь, Мануель Маццара, Емілія Цимпіан, Олександр Урбанец // Текст доповіді Першої міжнародної робочої конференції з обчислення бізнес-процесів і послуг, Лейпциг, 25-26 вересня 2007 р. —

Режим доступу: https://www.academia.edu/4252255/Business_Process_Modeling_Classifications_and_Perspectives.

11. What is business process modeling? [Електронний ресурс] – Режим доступу до ресурсу: https://www.softwareag.com/en_corporate/resources/process-management/article/business-process-modeling-analysis.html
12. What is Business Process Modeling? [Електронний ресурс] – Режим доступу до ресурсу: https://camunda.com/blog/2024/05/what-is-business-process-modeling/
13. Create IDEF0 diagrams [Електронний ресурс] – Режим доступу до ресурсу: https://support.microsoft.com/en-us/office/create-idef0-diagrams-ea7a9289-96e0-4df8-bb26-a62ea86417fc
14. What is a Swimlane Diagram? [Електронний ресурс] – Режим доступу до ресурсу: https://miro.com/diagramming/what-is-a-swimlane-diagram/
15. What is a Flowchart [Електронний ресурс] – Режим доступу до ресурсу: https://www.lucidchart.com/pages/what-is-a-flowchart-tutorial
16. SIPOC [Електронний ресурс] – Режим доступу до ресурсу: https://sipoc.info/
17. Minitab [Електронний ресурс] – Режим доступу до ресурсу: https://www.minitab.com/en-us/
18. Crystal Ball [Електронний ресурс] – Режим доступу до ресурсу: https://www.oracle.com/applications/crystalball/
19. Balanced Scorecard Basics [Електронний ресурс] – Режим доступу до ресурсу: https://balancedscorecard.org/bsc-basics-overview/
20. FMEA [Електронний ресурс] – Режим доступу до ресурсу: https://asq.org/quality-resources/fmea?srsltid=AfmBOorhFBHtN6WKLiQ7SxQlK6vWaT-A7qHTU-ZMVLeVi2r_6PZY5N-P
21. Що таке модель і нотація бізнес-процесів? [Електронний ресурс] – Режим доступу до ресурсу: https://www.microsoft.com/uk-ua/microsoft-365/visio/business-process-modeling-notation

22. Business process management software (BPMS) [Електронний ресурс] – Режим доступу до ресурсу: https://www.techtarget.com/searchcio/definition/Business-process-management-suite-BPMS
23. Руль М. К., Ганн М. Т. Software Reengineering: A Case Study and Lessons Learned // Спеціальна публікація (NIST SP). — 1991. — № 500-193. — С.48
24. Understand [Електронний ресурс] – Режим доступу до ресурсу: https://scitools.com/
25. Cloud Computing Study 2024 [Електронний ресурс] – Режим доступу до ресурсу: https://foundryco.com/research/cloud-computing/
26. Disability [Електронний ресурс] – Режим доступу до ресурсу: https://www.who.int/en/news-room/fact-sheets/detail/disability-and-health
27. he 2024 report on the accessibility of the top 1,000,000 home pages [Електронний ресурс] – Режим доступу до ресурсу: https://webaim.org/projects/million/
28. Бет Голд-Бернштейн Enterprise Integration: The Essential Guide to Integration Solutions / Бет Голд-Бернштейн , Вільям Ру — Бостон : Addison-Wesley Professional, 2004. — 432 с.
29. Безпека Amazon CodeGuru [Електронний ресурс] – Режим доступу до ресурсу: https://aws.amazon.com/ru/codeguru/
30. DeepCode [Електронний ресурс] – Режим доступу до ресурсу: https://www.deepcode.ca/
31. A McKinsey study [Електронний ресурс] – Режим доступу до ресурсу: https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/unleashing-developer-productivity-with-generative-ai
32. Duolingo Customer Stories [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/customer-stories/duolingo
33. Александра Стойков, Желько Стоянов. Software Reengineering: A Qualitative Study in two Software Companies [Електронний ресурс] / Александра Стойков, Желько Стоянов // Текст доповіді 4-го міжнародного

семінару з інформації, обчислень і систем керування для розподілених середовищ, Зренянин, липень 2022 р. — Режим доступу: https://www.researchgate.net/publication/364356818

34. Ахмед Салім Аббас. The Need of Re-engineering in Software Engineering / Ахмед Салім Аббас, В. Джеберсон, В.В. Клінсега // International Journal of Engineering and Technology. — 2012. — Т.2, № 12. — С. 291—295.
35. Гаррі М. Снід. Reengineering for Testability [Електронний ресурс] / Гаррі М. Снід // Workshop on Software Reengineering, Відень, травень 2006. — Режим доступу: https://www.academia.edu/58362148/Reengineering_for_Testability?uc-sb-sw=73066547
36. Майтуб Манар, Куткут Махмуд Х., Одех Юсра. Software Re-engineering: An Overview [Електронний ресурс] / Манар Майтуб, Махмуд Х. Куткут, Юсра Одех // Текст доповіді 8-ї Міжнародної конференції з комп'ютерних наук та інформаційних технологій, Аман, липень 2018 р. — Режим доступу: https://www.researchgate.net/publication/326696263_Software_Re-engineering_An_Overview.
37. Performance Optimization in Software Development [Електронний ресурс] – Режим доступу до ресурсу: https://senlainc.com/blog/performance-optimization-in-software-development/
38. Свонсон Е. Бертон. The dimensions of maintenance // Proceedings of the 2nd International Conference on Software Engineering (ICSE '76). — Вашингтон : IEEE Computer Society Press, 1976. — С. 492–497.
39. Каталін Боха Characteristics for Software Optimization Projects / Каталін Боха, Попа Маріус // Informatica Economica. — 2008. — Т.1, № 45. — С. 45—51.
40. Netflix Tech Evolution: Streaming into the Future [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/@movsesaleksanyan7/netflix-tech-evolution-streaming-into-the-future-2dca27457654

41. [Електронний ресурс] – Режим доступу до ресурсу: https://www.amazon.science/the-history-of-amazons-recommendation-algorithm
42. What Is Caching? [Електронний ресурс] – Режим доступу до ресурсу: https://hazelcast.com/foundations/caching/caching/
43. How Uber Serves Over 40 Million Reads Per Second from Online Storage Using an Integrated Cache [Електронний ресурс] – Режим доступу до ресурсу: https://www.uber.com/en-UA/blog/how-uber-serves-over-40-million-reads-per-second-using-an-integrated-cache/
44. Вівек Басавеговда Раму. Optimizing Database Performance: Strategies for Efficient Query Execution and Resource Utilization / Вівек Басавеговда Раму // International Journal of Computer Trends and Technology. — 2023. — Т. 71, № 7. — С. 15—21.
45. Consistent Data Partitioning through Global Indexing for Large Apache Hadoop Tables at Uber [Електронний ресурс] – Режим доступу до ресурсу: https://www.uber.com/en-UA/blog/data-partitioning-global-indexing/
46. Оптимізація мережевих запитів у прогресивних веб-додатках [Електронний ресурс] – Режим доступу до ресурсу: https://peerdh.com/uk/blogs/programming-insights/optimizing-network-requests-in-progressive-web-apps
47. Протокол HTTP/2: що це, переваги та як ним користуватися [Електронний ресурс] – Режим доступу до ресурсу: https://vps.ua/blog/ukr/protocol-http-2-benefits/
48. Cloudflare [Електронний ресурс] – Режим доступу до ресурсу: https://www.cloudflare.com/application-services/products/website-optimization/
49. Абхішек Шукла. Introducing Multi-Threaded Programming in Parallel Programming Process for Optimal Performance Results / Абхішек Шукла // Journal of Computational and Applied Mathematics. — 2023. — Т.2, № 4. — С. 1—3.

50. Multithreaded recalculation in Excel [Електронний ресурс] – Режим доступу до ресурсу: https://learn.microsoft.com/en-us/office/client-developer/excel/multithreaded-recalculation-in-excel
51. Camunda Modeler [Електронний ресурс] – Режим доступу до ресурсу: https://camunda.com/platform/modeler/
52. WCAG Color Contrast Checker [Електронний ресурс] – Режим доступу до ресурсу: https://accessibleweb.com/color-contrast-checker/
53. Як кнопки зворотного зв`язку та віджети на сайті підвищують конверсію [Електронний ресурс] – Режим доступу до ресурсу: https://unitalk.cloud/uk/insajty-dlya-biznesu/yak-knopky-zvorotnogo-zvyazku-tai-vidzhety-na-sajti-pidvyshhuyut-konversiyu/
54. Gantt chart guide: How to create and use one [Електронний ресурс] – Режим доступу до ресурсу: https://www.atlassian.com/agile/project-management/gantt-chart