

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра інформаційних управляючих систем та технологій

"Допущений до захисту"

Завідувач кафедри ІУСТ

к.т.н, доц. Михелєв І.Л.

" ____ " _____ 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти "магістр"

на тему: Дослідницький цикл транзитної обробки багатовимірних даних

Виконав: студент групи 6145 м

(підпис) Вершинін В.В.

Керівник роботи:

доцент кафедри ІУСТ, д.т.н.,
(посада, науковий ступінь вчене звання)

(підпис) Казимиренко Ю.О.

м. Миколаїв – 2023 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова
Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра інформаційних управляючих систем та технологій
Спеціальність 124 "Системний аналіз"
Освітня програма "Системний аналіз"

"ЗАТВЕРДЖУЮ"

Гарант освітньої програми

д.ф-м.н, проф. Ушкац М.В.

"_____" _____ 2023 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ на здобуття ступеня вищої освіти "магістр"

Студенту Вершиніну Вадиму Володимировичу

(прізвище, ім'я, по батькові)

1. Тема роботи: Дослідницький цикл транзитної обробки багатовимірних даних

Керівник роботи Казимиренко Ю.О.

Затверджені наказом ректора № 1098-уч. від "18" вересня 2023 року.

2. Термін подання роботи: 15 грудня 2023 р.

3. Вихідні дані до проекту (роботи) ДСТУ щодо обробки інформації, літературні джерела, технічна документація на існуючі аналоги систем, матеріали наукового стажування

4. Перелік питань, що належать до розробки (найменування розділів)

Вступ. 1. Багатовимірні дані, прототипування і типи їх обробки

2. Модель системи та організація проекту

3. Аналіз циклу транзитної обробки багатовимірних даних

5. Охорона праці

6. Охорона навколишнього середовища. Висновки. Список

використаних джерел. Додатки

5. Перелік презентаційних матеріалів: схема порівняння моделей бізнес-процесів, схема транзитної обробки багатовимірних даних

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Доцент Казимиренко Ю.О.		
2	Доцент Казимиренко Ю.О.		
3	Доцент Казимиренко Ю.О.		
4	Доцент Казимиренко Ю.О.		
5	Доцент Казимиренко Ю.О.		
6	Доцент Казимиренко Ю.О.		

7. Дата видачі завдання 05 вересня 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів роботи	Примітка
1	Проектування інформаційної системи		
2	Розробка програмного забезпечення		
3	Техніко-економічне обґрунтування економічної ефективності		
4	Розробка заходів щодо охорони праці		
5	Розробка заходів щодо захисту навколишнього середовища		
6	Підготовка презентації та доповіді		
7	Оформлення пояснювальної записки		
8	Попередній захист		
9	Нормоконтроль, рецензування		
10	Допуск до захисту		
11	Занесення диплома в електронний архів		

Студент

(підпис)

Вершинін В.В.

(ПІБ)

Керівник роботи

(підпис)

Казимиренко Ю.О.

(ПІБ)

АНОТАЦІЯ

Магістерська робота «Дослідницький цикл транзитної обробки багатовимірних даних» за спеціальністю 124 – системний аналіз.

Дипломна робота “Дослідницький цикл транзитної обробки багатовимірних даних” присвячена вирішенню важливого науково-прикладного завдання побудови та аналізу моделі обробки багатовимірних даних. В роботі описується сучасні технології обробки даних людиною та Штучним Інтелектом, розробляється прийнятна модель обробки даних та аналізуються перспективи її використання на окремому бізнес-процесі.

Актуальність роботи полягає в створенні сучасної системи збору та інтерпретації великих даних з потенціалом масштабування від окремого бізнес-процесу до автономного сховища.

Дана робота містить інформацію про опис бізнес моделей та процесів, алгоритм побудови, інформацію про використане програмне забезпечення на підприємстві, економічне обґрунтування використання моделі на основі результатів імплементації в бізнес-процеси, результати роботи і систему агрегування та інтерпретації даних.

Робота написана на 91 сторінках машинописного тексту, містить 26 рисунків, 3 таблиці, 3 додатка і список використаних джерел з 19 найменувань.

Ключові слова: багатовимірні дані, транзитна обробка даних, системний аналіз, процес майнінг, штучний інтелект.

ABSTRACT

Master's thesis "Research cycle of transit processing of multidimensional data" in specialty 124 - system analysis.

The thesis "Research cycle of transit processing of multidimensional data" is dedicated to the solution of an important scientific and applied task of building and analyzing a model of multidimensional data processing. This paper describes modern human and artificial intelligence data processing technologies, develops an acceptable data processing model, and analyzes the prospects for its use in a separate business process. The relevance of the work lies in the creation of a modern system of collection and interpretation of big data with the potential of scaling from a separate business process to an autonomous storage. This work contains information about the description of business models and processes, the construction algorithm, information about the software used at the enterprise, the economic justification of the use of the model based on the results of implementation in business processes, the results of work and the data aggregation and interpretation system. The work is written on 91 pages of typewritten text, contains 26 figures, 3 tables, 3 appendices and a list of used sources from 19 names.

Keywords: multidimensional data, transit data processing, system analysis, process mining, artificial intelligence.

ЗМІСТ

Вступ	8
РОЗДІЛ 1 Багатовимірні дані, прототипування і типи їх обробки	
1.1 Багатовимірні дані	10
1.2 Використання Process Mining при транзитній обробці великих даних	
1.3 Аналіз невирішених питань та постановка завдань для дослідження	
Висновки до 1 розділу	
РОЗДІЛ 2 Модель системи та організація проекту	
2.1 Аналіз структури підприємства	
2.2 Опис спроектованої системи	
2.3 Алгоритм побудови сховища даних	
Висновки до 2 розділу.	
РОЗДІЛ 3 Аналіз циклу транзитної обробки багатовимірних даних	
3.1 Агрегація даних	
3.2 Приклад реалізації	
3.3 Практичні рекомендації на основі проведеного дослідження	
Висновки 3 розділу	
РОЗДІЛ 4 Економічний розділ	
РОЗДІЛ 5 Охорона праці	
РОЗДІЛ 6 Охорона навколишнього середовища	
Висновки	
Використані джерела інформації	
ДОДАТКИ	

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ПМ – процесний майнінг, process mining

ІІ - штучний інтелект, artificial intelligence

ВСТУП

Високотехнологічні компанії є рушієм світової економіки та науково технічного прогресу. Ці процеси відбуваються завдяки агрегації та аналізу даних, які є новим надважливим ресурсом людства. Рівень розуміння даних по продукту та клієнту формують рівень успішності самої компанії. Великі обсяги даних потребують великих затрат на їх агрегацію, обробку та аналіз, що призводить до необхідності впровадження складних багаторівневих систем зберігання та обробки. Підбір або розробка власної валідної моделі в довгостроковій перспективі дозволяє автоматизувати та оптимізувати системи керування підприємством та покращувати якість продукції.

Мета роботи - ґрунтовний аналіз та розробка дослідницького циклу транзитної обробки багатовимірних даних.

Завдання роботи:

1. Проаналізувати та визначити переваги і недоліки сучасних методів обробки даних в складних багаторівневих системах.
2. Обрати модель обробки даних для дослідження і можливої оптимізації.
3. Провести повний цикл дослідження на базі обраної моделі.
4. Запропонувати економічне обґрунтування, щодо впровадження розробленої моделі на підприємстві.
5. Розглянути заходи щодо охорони праці та навколишнього середовища.

Об'єктом дослідження є процеси транзитної обробки багатовимірних даних.

Предметом дослідження є методи агрегації, обробки та аналізу багатовимірних даних.

Апробацію результатів дослідження представлено на XXXIX-ої Міжнародної науково-практичної конференції “Сучасні аспекти модернізації науки: стан, проблеми, тенденції розвитку” (м. Гетеборг (Швеція), 2023 р.), Практичні результати роботи впроваджені на підприємстві, дослідження були використані

для прототипування системи обробки багатовимірних даних в компанії яка займається розробкою та видавництвом мобільних відеоігор. Прототип виявився вдалим, система була розроблена та інтегрована в бізнес-процеси розробки повного циклу.

Публікації за результатами магістерської роботи:

1 Перспективи та обґрунтування використання штучного інтелекту в process mining /В. В. Вершинін //Сучасні аспекти модернізації науки: стан, проблеми, тенденції розвитку: матеріали XXXIX Міжнародної науково-практичної конференції / за ред. І.В. Жукової, Є.О. Романенка. м. Гетеборг (Швеція): ГО «ВАДНД», 07 грудня 2023 р., С. 455-459

РОЗДІЛ 1

БАГАТОВИМІРНІ ДАНІ, ПРОТОТИПУВАННЯ І ТИПИ ЇХ ОБРОБКИ

1.1 Багатовимірні дані

Аналіз багатовимірних даних - це процес, який включає в себе роботу з наборами даних, що містять багато різних змінних або атрибутів. Цей тип аналізу є більш складним, оскільки потребує врахування взаємозв'язків та взаємодій між численними змінними. Розглянемо основні кроки аналізу багатовимірних даних:

1. Збір даних з різних джерел та їх очищення і структурування. Важливо визначити відповідні змінні та впорядкувати дані таким чином, щоб вони були придатні для аналізу.
2. Розуміння даних та визначення мети аналізу, цей етап включає первинний аналіз даних для визначення їх характеристик та визначення цілей аналізу.
3. Застосування методів зниження розмірності, в випадках, коли працювати з багатовимірними даними може бути складно, часто використовуються методи, такі як аналіз головних компонент (РСА) або факторний аналіз, для зменшення кількості змінних, що розглядаються, не втрачаючи при цьому важливої інформації.
4. Використання методів кластерного аналізу та класифікації, допомагає групувати дані на основі схожості або певних характеристик і можуть бути корисними для ідентифікації закономірностей або груп у багатовимірних даних.
5. Використання статистичних і машинних алгоритмів навчання для вивчення взаємозв'язків між змінними та для прогнозування або висновків.
6. Візуалізація даних - може бути складною задачею, але вона є ключовою для розуміння складних взаємозв'язків і закономірностей. Використовуються різні техніки, такі як багатовимірне шкалювання, теплові карти тощо.

7. Інтерпретація результатів дає глибоке розуміння взаємозв'язків та впливу різних змінних, а також здійснення обґрунтованих висновків на основі аналізу.

Агрегація багатовимірних даних - це процес комбінування та узагальнення даних з метою зменшення їх складності та забезпечення зручності їх аналізу. Цю систему можна візуалізувати за допомогою куба, в більшості випадків він має наступний схематичний вигляд (рисунок 1.1).

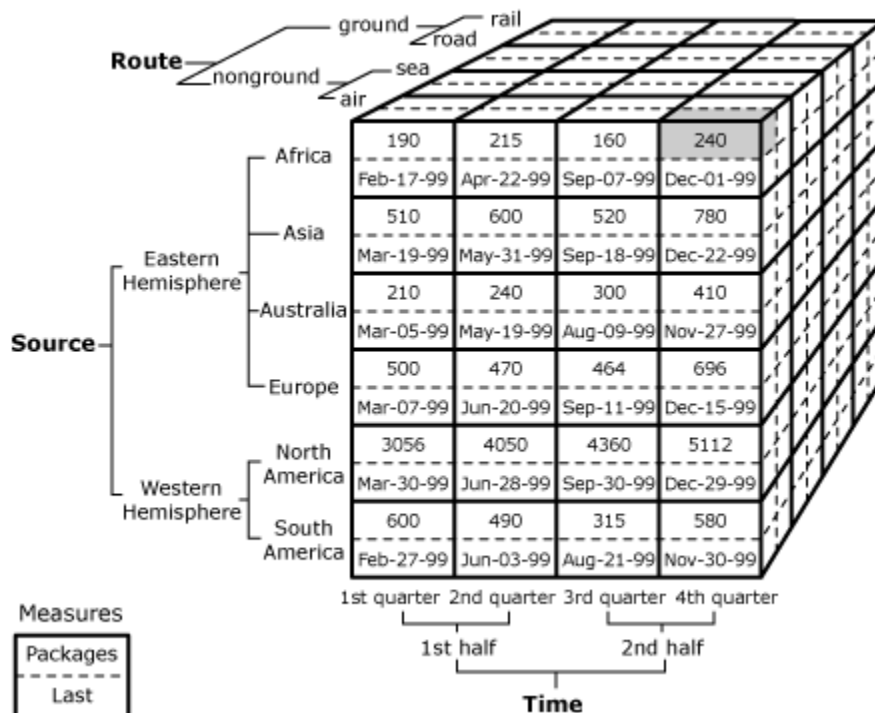


Рисунок 1.1 – Багатовимірний масив даних

Ось кілька ключових методів агрегації:

Один з найпростіших способів агрегації - сумування або обчислення середніх значень за певними змінними. Наприклад, у фінансовому аналізі можна підсумувати продажі по регіонах або обчислити середній дохід на користувача.

Зменшення розмірності - це методи, такі як аналіз головних компонент (РСА) або факторний аналіз, дозволяють зменшити кількість змінних, зберігаючи при цьому найбільш важливу інформацію. Це допомагає спростити аналіз та візуалізацію даних.

Групування за Атрибутами, це тип даних, які можуть бути агреговані за певними атрибутами або категоріями. Наприклад, в медичних дослідженнях дані можуть бути узагальнені за віковими групами або за діагнозами.

Використання Кластерного Аналізу дозволяє групувати дані на основі схожості або відмінностей у великому наборі змінних, що дозволяє узагальнити та вивчити дані на більш високому рівні.

Часова Агрегація, це тип даних, які збираються протягом часу, можна агрегувати значення за певні періоди часу, такі як дні, місяці або роки.

Просторова Агрегація може включати узагальнення даних по регіонах, місцевостях або інших просторових одиницях.

OLAP (Online Analytical Processing) куб, що дозволяють агрегувати багатовимірні дані з різних перспектив, наприклад, по продажам, часу та географічному розташуванню.

Process mining - це методика, що використовується для аналізу бізнес-процесів на основі цифрових слідів, залишених у системах інформаційних технологій. Цей метод дозволяє організаціям розуміти, як їхні бізнес-процеси функціонують на практиці, та виявляти можливості для оптимізації та покращення.

1.2 Використання Process Mining при транзитній обробці великих даних

Згідно з [1], “бізнес-процеси – це комплекс дій, спрямованих на досягнення встановлених у компанії цілей”. Вони включають в себе кроки, які необхідно виконати, і ресурси, які використовуються, для створення продукту чи послуги, які задовольняють потреби клієнтів. Бізнес-процеси можуть бути різноманітні: від дуже структурованих і до чітко визначених до більш гнучких та адаптивних. Вони є ключовим елементом у забезпеченні ефективності та продуктивності будь-якої організації, і слугують фундаментом для щоденної діяльності організацій, тому їх постійно намагаються оптимізувати та покращувати. У цьому процесі велику роль відіграють інформаційні системи, які фіксують

виконання кожного етапу бізнес-процесів у вигляді журналів подій. Ці журнали включають послідовності подій з точними часовими мітками, кожна з яких відображає окрему дію в процесі. ПМ – це група методів для аналізу цих записів, щоб зрозуміти та вдосконалити реальну роботу бізнес-процесів.[2]

Загалом, ПМ складається з трьох основних частин:

- Виявлення
- Конформація
- Покращення

Розглянемо кожний із них окремо.

1.3.1 Виявлення

Етап Виявлення (Discovery) є частиною управління бізнес процесами[3]. : Аналізується журнал подій для створення моделі бізнес-процесу, яка відображає те, як процеси насправді виконуються. Ця фаза фокусується на створенні моделі процесу, яка відображає реальні дії та послідовності подій, зафіксованих у системах. Використовуючи алгоритми майнінгу, такі як алгоритми виявлення шаблонів чи залежностей, цей етап дає можливість виявити стандартні та нетипові шляхи виконання процесів. Він допомагає ідентифікувати ключові етапи, взаємодії між різними активностями та можливі вузькі місця чи відхилення від стандартних процедур. Завдяки цьому етапу можна отримати глибоке розуміння функціонування бізнес-процесів, що є важливим для подальшої їх оптимізації та вдосконалення. Результатом етапу виявлення є процесна модель. Для визначення процесної моделі використовуються наступні техніки:

Альфа алгоритм[4, 5]. Алгоритм α переглядає журнал подій для виявлення специфічних зв'язків. Так, якщо дія a завжди передуює дії b , але дія b ніколи не відбувається після a , то робиться висновок про наявність причинно-наслідкового зв'язку між a та b . Ця залежність повинна бути представлена у відповідній мережі Петрі, де має бути місце з'єднання a з b . Ми розрізняємо чотири типи відносин

порядку, які базуються на журналах подій та призначені для виявлення ключових шаблонів у цих записах.

Нехай L буде журналом подій над A , тобто, $(L \in \mathbb{IB}(A^*))$. Нехай $(a, b \in A)$:

- $(a >_L b)$ тоді і тільки тоді, коли існує слід $(\sigma = ht_1, t_2, t_3, \dots, t_n i)$ та $(i \in \{1, \dots, n-1\})$ такі,

що $(\sigma \in L)$ та $t_i = a \quad t_{i+1} = b$,

- $(a \rightarrow_L b)$ тоді і тільки тоді, коли $(a >_L b)$ і $(b \not>_L a)$,

- $(a \#_L b)$ тоді і тільки тоді, коли $(a \not>_L b)$ і $(b \not>_L a)$, і

- $(a \leftrightarrow_L b)$ тоді і тільки тоді, коли $(a >_L b)$ і $(b >_L a)$.

Розглянемо, наприклад, $(L_1 = [ha, b, c, di^5, ha, c, b, di^8, ha, e, di^9])$. $(c >_{L_1} d)$ оскільки d безпосередньо слідує за c у сліді (ha, b, c, di) . Однак, $(d \not>_{L_1} c)$ оскільки c ніколи не слідує безпосередньо за d у жодному сліді в журналі.

$(>_{L_1} = \{(a, b), (a, c), (a, e), (b, c), (c, b), (b, d), (c, d), (e, d)\})$ містить усі пари діяльностей у відношенні "безпосередньо слідує". $(c \rightarrow_{L_1} d)$ оскільки іноді d безпосередньо слідує за c і ніколи навпаки $((c >_{L_1} d) \text{ і } (d \not>_{L_1} c))$.

$(\rightarrow_{L_1} = \{(a, b), (a, c), (a, e), (b, d), (c, d), (e, d)\})$ містить усі пари діяльностей у відношенні "причинності". $(b \leftrightarrow_{L_1} c)$ оскільки $(b >_{L_1} c)$ і $(c >_{L_1} b)$, тобто іноді c слідує за b і іноді навпаки. $(\leftrightarrow_{L_1} = \{(b, c), (c, b)\})$. $(b \#_{L_1} e)$

оскільки $(b \not>_{L_1} e)$ і $(e \not>_{L_1} b)$. $(\#_{L_1} = \{(a, a), (a, d), (b, b), (b, e), (c, c), (c, e), (d, a), (d, d), (e, b), (e, c), (e, e)\})$. Зверніть увагу, що для будь-якого журналу подій L над A та $x, y \in A$:

$(x \rightarrow_L y, y \rightarrow_L x, x \#_L y,)$ або $(x \leftrightarrow_L y)$.

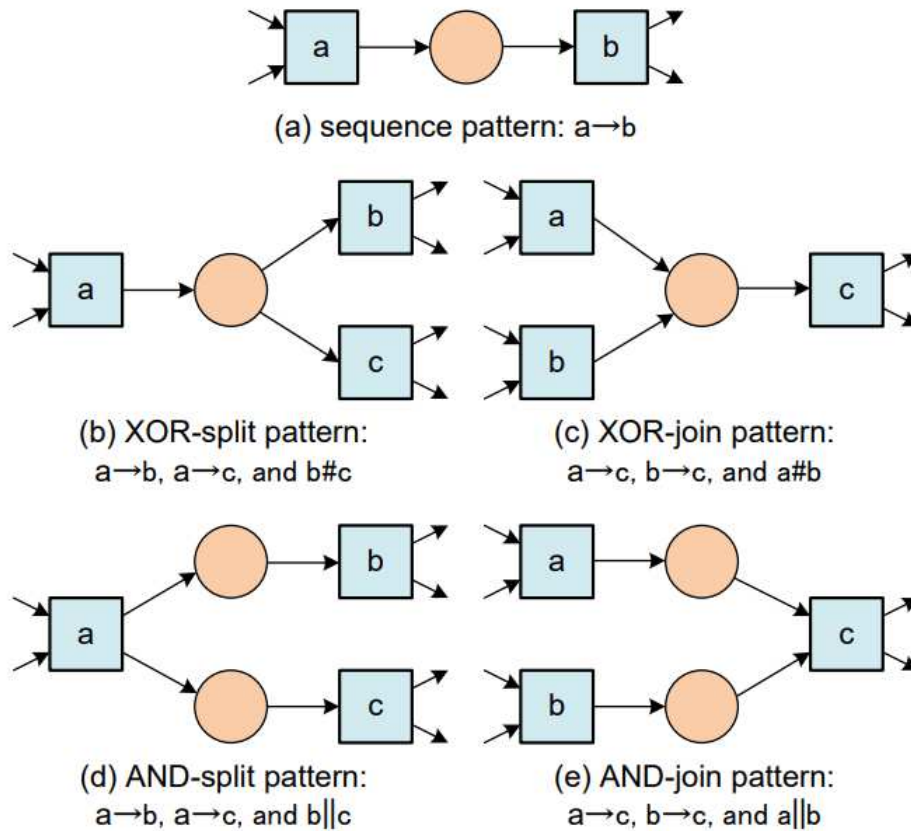


Рисунок 1.2 - Типові патерни і шляхи, якими вони проходять через журнал подій

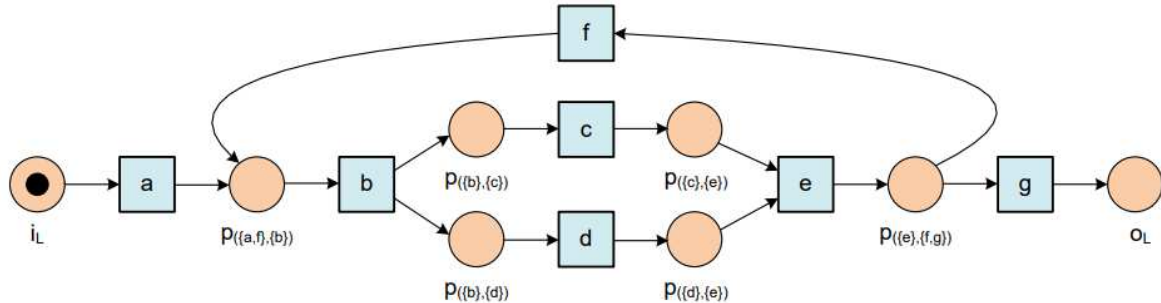


Рисунок 1.3 -Мережа N2, побудована на основі журналу подій L2

Розгляньмо, наприклад, мережу N2, яка зображена на рис.2 та побудована на основі журналу подій L2 = $([ha, b, c, d, e, f, b, d, c, e, gi, ha, b, d, c, e, gi, ha, b, c, d, e, f, b, c, d, e, f, b, d, c, e, gi])$.

Алгоритм α створює WF-net N2, використовуючи L2. Зауважте, що виявлені шаблони у моделі дійсно відповідають відношенням порядку, отриманим з журналу подій. Розглянемо фрагмент процесу, що включає b, c, d та e. Цей фрагмент може бути побудований на основі $(b \rightarrow_{L2} c)$, $(b \rightarrow_{L2} d)$, $(c \leftrightarrow_{L2} d)$, $(c \rightarrow_{L2} e)$, та $(d \rightarrow_{L2} e)$. Вибір, що йде після e, виявляється через $(e \rightarrow_{L2} f)$, $(e \rightarrow_{L2} g)$, та $(f \#_{L2} g)$. І т.д.

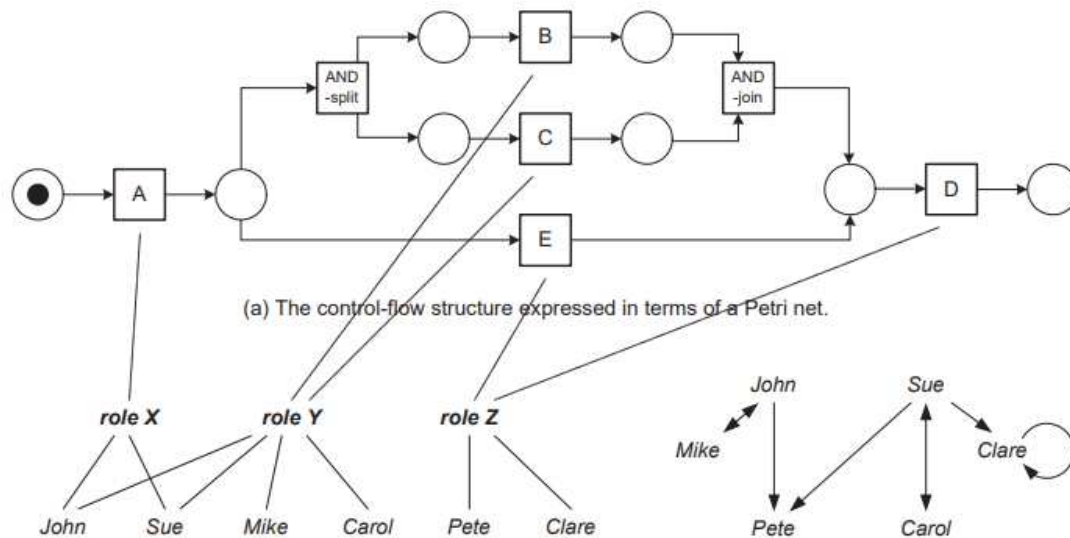


Рисунок 1.4 Організаційна структура виражена в діаграмі
Плагін HeuristicsMiner аналізує перспективу контролю потоків процесу моделі.
Для цього він враховує лише порядок подій в межах окремого випадку. Іншими
словами, порядок подій між випадками не має значення. Наприклад, для журналу
в Tabletablog під час аналізу враховуються лише поля ідентифікатора випадку,
часової мітки та активності. Часова мітка активності використовується для
визначення цього порядку.

case id	activity id	originator	time stamp
case 1	activity A	John	9-3-2004:15.01
case 2	activity A	John	9-3-2004:15.12
case 3	activity A	Sue	9-3-2004:16.03
case 3	activity B	Carol	9-3-2004:16.07
case 1	activity B	Mike	9-3-2004:18.25
case 1	activity C	John	10-3-2004:9.23
case 2	activity C	Mike	10-3-2004:10.34
case 4	activity A	Sue	10-3-2004:10.35
case 2	activity B	John	10-3-2004:12.34
case 2	activity D	Pete	10-3-2004:12.50
case 5	activity A	Sue	10-3-2004:13.05
case 4	activity C	Carol	11-3-2004:10.12
case 1	activity D	Pete	11-3-2004:10.14
case 3	activity C	Sue	11-3-2004:10.44
case 3	activity D	Pete	11-3-2004:11.03
case 4	activity B	Sue	11-3-2004:11.18
case 5	activity E	Clare	11-3-2004:12.22
case 5	activity D	Clare	11-3-2004:14.34
case 4	activity D	Pete	11-3-2004:15.56

Таблиця 1. Журнал подій

У Таблиці 1 важливо, що для випадку 1 активність А слідує за активністю В у контексті цього випадку, а не те, що активність А випадку 1 слідує за активністю А випадку 2. Тому ми визначаємо журнал подій наступним чином. Нехай T буде набором активностей. $(\sigma \in T^*)$ - це слід події, тобто довільна послідовність ідентифікаторів активностей. $(W \subseteq T^*)$ - це журнал подій, тобто мультипослідовність. Зауважте, що оскільки W є мультимножиною, кожен слід подій може з'являтися у журналі більше ніж один раз. У практичних інструментах майнінгу важливими стають частоти. Якщо ми використовуємо цю нотацію для опису журналу, показаного в Таблиці 1, ми отримуємо мультимножину $W = ABCD, ABCD, ACBD, ACBD, AED$.

Для знаходження моделі процесу на основі журналу подій, потрібно проаналізувати журнал на наявність причинно-наслідкових залежностей, наприклад, якщо одна активність завжди слідує за іншою, існує велика ймовірність, що між цими активностями є залежність. Для аналізу цих зв'язків ми вводимо наступні позначення.

Нехай W це журнал подій по T , тобто WT^* . Нехай $a, b \in T$:

1. $a > Wbiff$ має зв'язок $= t_1 t_2 t_3 \dots t_n$ і $i_1, \dots, n-1$ вигляду $W_i t_i = a$ і $t_{i+1} = b$,
2. $a Wbiff a > Wb_i b > W a$,
3. $a Wbiff a > Wb_i b > W a_i$,
4. $a Wbiff a > Wb_i b > W a$,
5. $a >> Wbiff$ має зв'язок $= t_1 t_2 t_3 \dots t_n$ і $i_1, \dots, n-2$ вигляду $W_i t_i = a$ і $t_{i+1} = b$ і $t_{i+2} = a$,
6. $a >>> Wbiff$ має зв'язок $= t_1 t_2 t_3 \dots t_n$ і $i < j$ і $i, j_1, \dots, n$ вигляду $W_i t_i = a$ і $t_j = b$.

Індуктивний майнер[7]

ІМ намагається знайти розрізи в порядку $(\vee), (\wedge)$ та $(\Rightarrow)$. Як тільки зустрічається нетривіальний розріз, цей розріз повертається функцією *findCutIM*. Спочатку ми представляємо псевдокод цих розрізів, після чого доводимо їхню правильність та обговорюємо їхні гарантії. Ми доведемо правильність для журналів подій без порожніх слідів, це припущення буде виконано за допомогою *emptyTraces*. У випадку, якщо журнал подій не містить сліду взаємодії, то всі активності будуть об'єднані, і розподіл складатиметься з одного набору. Остаточна функція *findCutIM* застосовує наступні функції виявлення сліду взаємодії, поки одна з них не знайде розріз з розподілом, що складається з декількох наборів. Виключний Вибір. Щоб виявити розріз виключного вибору, виконайте наступні кроки: функція *xorCut()*($\Sigma_1 \dots \Sigma_k$) - вузли з'єднаних компонентів повертають $(p, \Sigma_1 \dots \Sigma_k)$.

Лог L, який задовольняє умову R L, має властивість таку, що якщо функція *xorCutppLqq* повертає відрізок, це відповідає максимальному -відрізку в L. В результаті побудови зв'язаних компонентів, з'єднання між частинами, що відповідають розділу активностей відрізка, відсутні.

Мережі Петрі

Статистичні мережі Петрі (SPN) представляють просту графічну мову для опису стохастичних процесів[8]. Ми вибрали SPN для побудови наших моделей через їхню відносну простоту, що робить SPN ідеальним формалізмом для створення складних моделей з використанням менших як будівельних блоків.

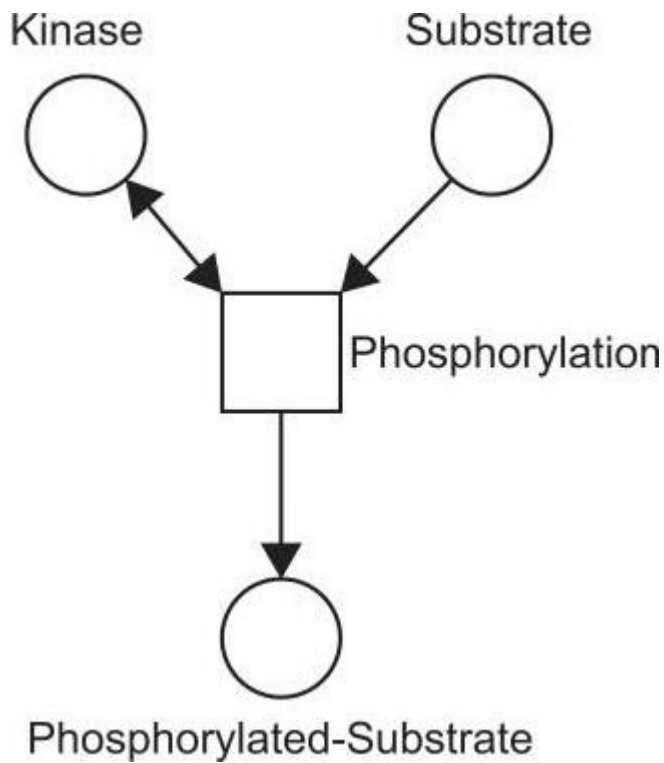


Рисунок 1.5 - Мережа Петрі

На відміну від графів, Петрієві мережі детальніше зображують реакції, і тому не завжди очевидно, як відобразити мережі взаємодій білків на моделі Петрієвої мережі. Моделі Петрієвих мереж представляють математичну модель динамічного процесу, тоді як графи - це статичне зображення взаємодій. Петрієві мережі вимагають більшої параметризації, ніж графи (наприклад, кількість кожного виду і стехіометрія реакцій), що часто є значною перешкодою для їхнього конструювання. Взаємодії, які формують графи, регулярно вимірюються на великому масштабі для багатьох організмів, тоді як те саме не можливо для Петрієвих мереж. Важливо, що Петрієві мережі можуть бути перетворені в серію матриць, які потім можуть бути використані для симуляцій, в яких ряд переходів "запускаються", переміщуючи жетони між місцями. Для симуляцій, моделі SPN можуть бути перетворені в матриці та завантажені у форматі JSON.

Діаграми діяльності

[9]

Моделювання діяльності фокусується на організації нижчих рівнів завдань, а не на визначенні того, хто володіє ними. Ці моделі, відомі як моделі

контролю потоку та об'єктного потоку, координують дії на основі завершених завдань, доступних ресурсів або зовнішніх подій.

Дії та активності

Виконання дії означає здійснення певного завдання, тоді як виконання активності включає в себе кілька завдань. Кожне завдання в межах активності може відбуватися кілька разів за кожне виконання активності. Завдання вимагають доступу до даних, їхньої трансформації, тестування та послідовності. Моделі вищого рівня активності дозволяють кілька контрольних потоків та синхронізацію для забезпечення впорядкованого виконання. Ці моделі можуть бути адаптовані для розподілених систем, але це не означає обов'язкову розподілену структуру програмного забезпечення. Валідним також є послідовне виконання, що відповідає обмеженням послідовності моделі.

Різні реалізації можуть втілювати одну й ту ж модель, і будь-яка, яка зберігає інформацію та поведінку моделі, є прийнятною. Реалізації можуть відрізнятися структурно від моделі, що призводить до різних відображень між ними. Ці відображення не обов'язково є один-до-одного і можуть навіть не використовувати об'єктно-орієнтований підхід або первісний набір класів.

Відображення реалізації може бути ручним, з використанням фізичних моделей комп'ютерів та завдань для цілей реалізації, або автоматичним. Хоча ця специфікація не забезпечує безпосередньо цих накладок або генерації коду, вона дозволяє обидва методи.

Основні Активності

На фундаментальному рівні діяльності визначаються як включаючі вузли, до яких належать дії. Цей рівень є спільним для потокових та структурованих форм діяльності.

Базові Активності

Цей рівень включає послідовне управління та потоки даних між діями, але явні розділення та з'єднання контролю, а також рішення та об'єднання, не підтримуються. Базовий та структурований рівні є ортогональними. Можна використовувати або один, або обидва для підтримки моделювання, яке включає потоки та структуровані контрольні конструкції.

Проміжні Активності

Проміжний рівень підтримує моделювання діаграм діяльності, що включають паралельний контроль та потоки даних, а також рішення. Він підтримує моделювання, подібне до традиційних Петрієвих мереж з чергуванням. Для цього потрібен базовий рівень.

Повні Активності

Повний рівень додає конструкції, які покращують моделі нижчих рівнів, такі як вага ребер та потоковість.

Структуровані Активності

Структурований рівень підтримує моделювання традиційних структурованих програмних конструкцій, таких як послідовності, цикли та умовні вирази, як доповнення до основних вузлів діяльності. Він вимагає фундаментального рівня та сумісний з проміжним та повним рівнями.

Повні Структуровані Активності

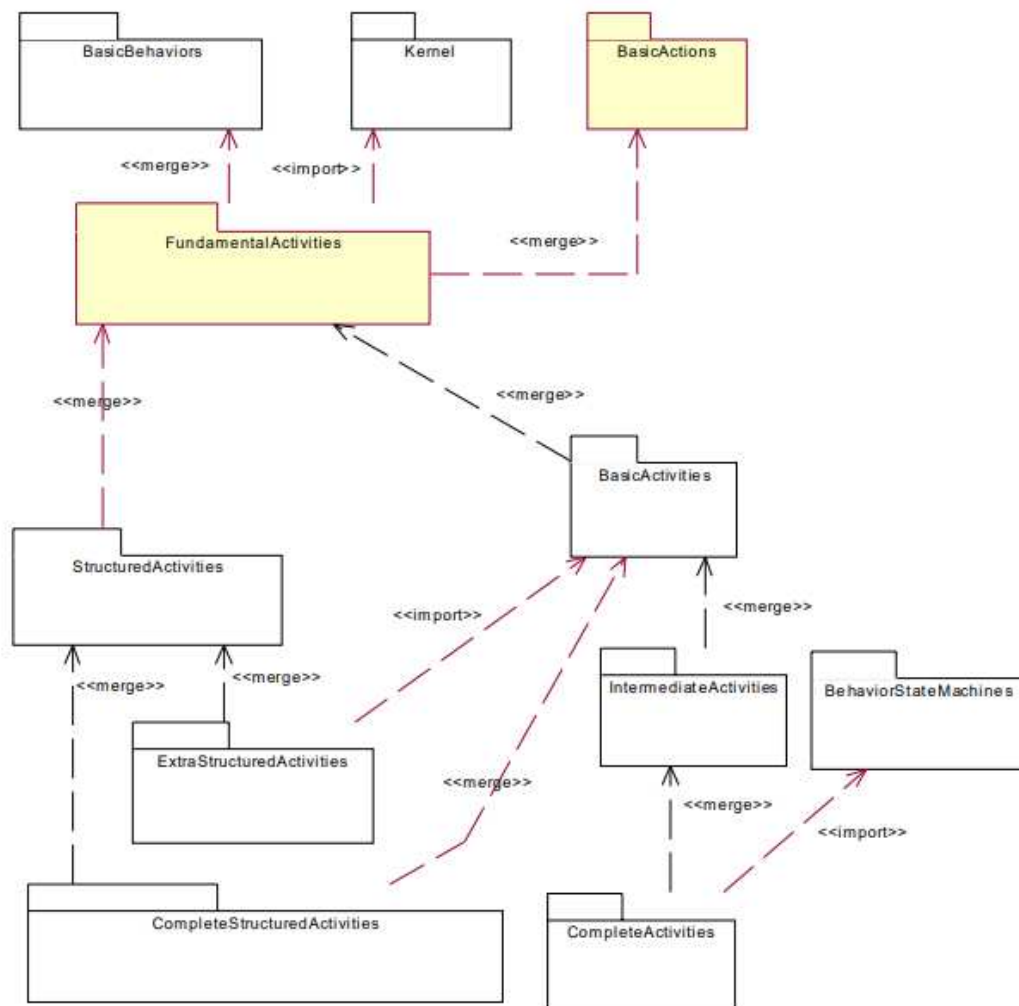
Цей рівень додає підтримку вихідних потоків даних для послідовностей, умовних виразів та циклів. Він залежить від базового рівня для потоків.

Додаткові Структуровані Активності

Додатковий структурований рівень підтримує обробку винятків, як у традиційних мовах програмування, та виклик поведінок на наборах значень. Він вимагає структурованого рівня.

Рисунок 1.6 - Залежності пакетів активності

Діаграми станів



Діаграма переходів станів (state-transition diagram) – це візуальне представлення поведінки об'єкта в різних станах та переходів між цими станами відповідно до певних подій. Кожен стан відображається як круг або прямокутник, а переходи між станами зображаються стрілками. Ця діаграма є ключовим елементом в моделюванні поведінки системи в області розробки програмного забезпечення та системного аналізу, особливо в контексті об'єктно-орієнтованого

модельовання. Вона допомагає у визначенні реакції системи на зовнішні впливи та в управлінні складними умовами переходу[10].

Побудова діаграми переходів станів включає декілька ключових кроків:

Визначення станів: Ідентифікуємо всі можливі стани, в яких може перебувати система або об'єкт. Стани часто описуються іменниками та відображають різні фази життєвого циклу об'єкта.

Визначення подій та переходів: Визначимо події, що спричиняють перехід об'єкта з одного стану в інший. Події можуть бути зовнішніми (наприклад, дії користувача) або внутрішніми (наприклад, досягнення певної умови).

Створення діаграми: Намалюймо діаграму, використовуючи кола або прямокутники для представлення станів і стрілки для представлення переходів між станами. Кожна стрілка позначена подією, яка спричиняє перехід.

Визначення дій або активностей: Для кожного стану можуть бути визначені дії або активності, які виконуються при вході, перебуванні в стані або виході з нього.

Перевірка та оптимізація: Перевіримо діаграму на наявність неповних або непотрібних переходів і внесіть необхідні корективи. Також переконаймося, що діаграма точно відображає бізнес-процеси або логіку системи.

Валідація з зацікавленими сторонами: Представимо діаграму зацікавленим сторонам для перевірки, що вона відповідає їхнім вимогам і очікуванням.

Ітерації та оновлення: Діаграма може бути змінена або доповнена з часом, особливо у відповідь на зміни в процесах або вимогах.

Цей процес є ітеративним і може вимагати повторних переглядів, особливо при розробці складних систем.

Діаграма станів (або діаграма переходів станів) може складатися з наступних елементів:

Стани: Представляють різні фази або умови, в яких може перебувати система або об'єкт. Кожен стан часто зображується у вигляді круга або прямокутника з назвою стану всередині.

Переходи: Це стрілки, які з'єднують стани, показуючи, як система переходить від одного стану до іншого. На стрілках може бути нанесена мітка, що описує подію або умову, яка спричиняє перехід.

Початковий стан: Це спеціальний символ (часто представлений як чорна точка), що вказує, з якого стану починається виконання або активність.

Кінцевий стан: Це символ, який вказує на завершення всіх процесів або активностей в контексті діаграми. Часто зображується як коло з закружкованим контуром.

Дії/Активності: Вказують на операції або активності, які виконуються в певному стані або під час переходу між станами.

Охоронні умови: Це умови, що вказують, чи можливий перехід між станами. Вони зазвичай розміщуються біля стрілки переходу.

Події: Події, які спричиняють перехід від одного стану до іншого.

Вкладені стани (Суб-стани): Дозволяють деталізувати поведінку всередині одного стану, розбиваючи його на декілька внутрішніх станів.

Паралельні стани: У діаграмах станів можуть бути представлені стани, які виконуються одночасно в рамках паралельних процесів.

Ланцюг послідовності процесів

Ланцюг подій (Event-driven Process Chain, EPC) є методологією моделювання бізнес-процесів, яка використовується для документування, аналізу та оптимізації бізнес-процесів. Основна ідея EPC полягає у відображенні процесів як серії подій, які визначають, коли та як виконуються різні завдання чи функції у межах організації. Ланцюгом подій може контролювати автономний екземпляр робочого потоку у спільній роботі[11].

Елементи, які використовуються у діаграмах ланцюгів подій:

Подія: Події є пасивними елементами у ланцюгах подій. Вони описують обставини, за яких працює функція або процес, або стан, у який вони призводять.

Приклади подій - "вимога зафіксована", "матеріал на складі". У графіку EPC подія представлена як шестикутник.

Функція: Функції є активними елементами в ЕРС. Вони моделюють завдання або діяльність у компанії. Функції описують перетворення від початкового стану до кінцевого. Якщо можливі різні кінцеві стани, вибір відповідного стану може бути явно модельований як функція рішення з використанням логічних з'єднувачів. Функції можуть бути деталізовані в іншій ЕРС. Приклади функцій - "фіксація вимог", "перевірка матеріалу на складі". У графіку ланцюгів подій функція представлена як округлений прямокутник.

Власник процесу: Відповідальний за функцію (наприклад, клерк бронювання відповідає за бронювання подорожей). Зазвичай є частиною організаційного підрозділу (наприклад, клерк бронювання належить до відділу бронювання). Представлений як квадрат з вертикальною лінією.

Організаційний підрозділ: Визначає, яка організація в структурі підприємства відповідає за певну функцію. Приклади - "відділ продажів", "відділ закупівель" тощо. Представлений як еліпс з вертикальною лінією.

Інформація, матеріал або ресурсний об'єкт: У ланцюгах подій вони зображують об'єкти в реальному світі, наприклад бізнес-об'єкти, ентитети тощо, які можуть бути вхідними даними, що слугують основою для функції, або вихідними даними, створеними функцією. Приклади - "матеріал", "замовлення". У графіку ЕРС такий об'єкт представлений як прямокутник.

Логічний з'єднувач: У ланцюгах подій описують логічні зв'язки між елементами в потоці управління, тобто подіями та функціями. З їх допомогою можна розділити потік управління з одного на два або більше потоків та синхронізувати потік управління з двох або більше потоків в один.

Логічні зв'язки:

Відгалуження/Злиття: Відповідає за вибір шляху серед кількох потоків управління. У графіку ЕРС відгалуження представлене як відкриваючий XOR, а злиття - як закриваючий XOR.

Розгалуження/З'єднання: Відповідає за активацію всіх шляхів у потоці управління одночасно. У графіку ЕРС розгалуження представлене як відкриваючий 'AND', а з'єднання - як закриваючий 'AND'.

OR: Відповідає за активацію одного або декількох шляхів серед потоків управління. У графіку EPC відкриваючий OR може активувати один або декілька потоків.

Потік управління: З'єднує події з функціями, шляхами процесів або логічними з'єднувачами, створюючи хронологічну послідовність та логічні залежності між ними. Представлений у вигляді пунктирної стрілки.

Потік інформації: Показує зв'язок між функціями та вхідними або вихідними даними, на які функція читає, змінює або пише.

Призначення організаційного підрозділу: Показує зв'язок між організаційним підрозділом та функцією, за яку він відповідає.

Шлях процесу: Служить для навігації в EPC, показуючи зв'язок з іншими процесами. Представлений як складений символ, який складається з символу функції, накладеного на символ події.

1.3.2 Конформація

Конформація (Conformance): Порівняння існуючої моделі процесу з журналом подій для визначення відхилень, помилок або не послідовностей між моделлю та реальними даними. Цей аналіз допомагає виявити розбіжності між теоретичними або запланованими процесами та їх реальним виконанням. Використовуючи конформаційний аналіз, можна ідентифікувати невідповідності, такі як пропущені або додаткові кроки, а також визначити частини процесу, які вимагають покращення або адаптації. Цей етап важливий для забезпечення того, щоб реальні процеси відповідали встановленим стандартам і поліпшення загальної продуктивності та ефективності організаційних операцій. Етап конформації складається з наступних частин:

Збір та Підготовка Даних: Це початковий крок, де збираються реальні дані про виконання процесів (логи подій). Дані можуть включати часові мітки, ідентифікатори виконавців, типи подій та інші відомості, які є критичними для аналізу.

Моделювання Процесу: Перед конформацією необхідно мати модель процесу, з якою будуть порівнюватися реальні дані. Ця модель може бути створена вручну або отримана з попереднього етапу ПМ, наприклад, з використанням дискавері майнінгу.

Вирівнювання (Alignment): На цьому етапі відбувається порівняння реальних даних з моделлю процесу. Вирівнювання допомагає виявити відхилення, конфлікти та невідповідності між тим, як процеси повинні виконуватися згідно з моделлю, і тим, як вони виконуються на практиці.

Аналіз Відхилень: Після вирівнювання аналізуються конкретні відхилення. Це може включати виявлення пропущених дій, дій, що виконуються занадто пізно або занадто рано, або дій, що виконуються в неправильному порядку.

Діагностика та Звітність: Після ідентифікації відхилень проводиться детальний аналіз причин цих відхилень. Результати можуть бути представлені у вигляді звітів або візуалізацій, які допомагають краще зрозуміти, де та чому відбуваються проблеми в процесах.

Вдосконалення Процесів: На основі отриманих результатів можуть бути запропоновані рекомендації щодо вдосконалення бізнес-процесів, що може включати зміни у процедурах, поліпшення контролю якості або навіть перепроєктування певних аспектів процесу.

1.3.3 Покращення

Покращення (Enhancement): Використання інформації, отриманої з журналів подій, для оптимізації та вдосконалення бізнес-процесів. Під час цього етапу, виявлені недоліки або відхилення в процесах використовуються для планування конкретних заходів щодо їх покращення. Це може включати модифікацію існуючих процесів, впровадження нових методів або технологій, а також розробку стратегій для підвищення ефективності та продуктивності. Основна мета цього етапу - забезпечити, щоб бізнес-процеси були більш відповідними до потреб і цілей організації.

Типові елементи етапу покращення:

Аналіз Результатів ПМ: Перед початком етапу покращення необхідно ретельно проаналізувати результати, отримані на попередніх етапах ПМ, таких як виявлення (discovery) і конформація (conformance). Це включає аналіз відхилень, проблемних зон, неефективних елементів та потенційних можливостей для покращення.

Розробка Сценаріїв Покращення: На основі аналізу розробляються сценарії покращення, які можуть включати зміни в процесах, автоматизацію певних завдань, оптимізацію послідовності дій або виправлення виявлених проблем.

Моделювання та Оцінка Змін: Застосування моделювання процесів для оцінки потенційного впливу запропонованих змін. Це може включати використання симуляції для перевірки, як зміни впливають на ефективність та продуктивність процесів.

Впровадження Змін: Реалізація обраних сценаріїв покращення в реальні бізнес-процеси. Це може включати зміну робочих інструкцій, внесення змін в ІТ-системи або навчання персоналу новим процедурам.

Моніторинг та Оцінка Ефективності: Після впровадження змін необхідно моніторити та оцінювати їх ефективність, порівнюючи нові результати з попередніми даними для визначення, чи досягнуто бажаних покращень.

Ітеративне Удосконалення: Покращення процесів часто є ітеративним процесом, де після впровадження перших змін проводиться додатковий аналіз та вносяться подальші корективи для досягнення оптимальних результатів.

1.3 Вирішення проблем діяльності організації шляхом ПМ

ПМ може вирішити ряд проблем організації, особливо тих, що пов'язані з ефективністю, прозорістю та оптимізацією бізнес-процесів. Розглянемо наступні приклади:

Виявлення та усунення затримок у процесах: ПМ може допомогти ідентифікувати кроки в процесі, де відбуваються непотрібні затримки.

Наприклад, якщо в процесі обробки замовлень існують невинновдані затримки між поданням замовлення та його підтвердженням, ПМ може виявити та вказати на конкретні точки, де ці затримки відбуваються, дозволяючи менеджменту вжити відповідних заходів.

Затримки в процесах можуть виникати з різних причин, серед яких:
Неефективне Розподілення Ресурсів: часто затримки виникають через недостатнє або неправильне розподілення ресурсів, як-от персоналу чи обладнання.

Погане Планування: неправильне планування або недостатня координація між відділами може призвести до непередбачених затримок.

Технічні Проблеми: проблеми з ІТ-системами або обладнанням також можуть бути причиною затримок.

Недоліки в Процесах: часто затримки виникають через недосконалі або застарілі процеси.

Оптимізація шляхів обробки: через аналіз потоків даних ПМ, компанії можуть виявити найбільш та найменш ефективні шляхи обробки завдань. Наприклад, виявлення того, що деякі запити клієнтів обробляються швидше за інші, може спонукати до перегляду та стандартизації процедур, щоб забезпечити більш послідовну якість обслуговування. Типові причини появи проблем з оптимізацією шляхів обробки:

Нестандартизовані Процеси: в організаціях, де процеси не стандартизовані, різні співробітники можуть виконувати однакові завдання різними способами, що призводить до непослідовності та неефективності.

Наявність Застарілих Процедур: іноді шляхи обробки залишаються незмінними протягом тривалого часу, незважаючи на зміни в технологіях та бізнес-середовищі.

Недостатнє Розуміння Процесів: відсутність глибокого розуміння процесів може призвести до використання неефективних шляхів обробки.

Виявлення відхилень від стандартних процедур: ПМ може виявити випадки, коли дії працівників або процеси відхиляються від затверджених процедур. Це може бути корисним для виявлення невідповідностей у процесах, що може вказувати на необхідність додаткового навчання, змін у процедурах або навіть виявлення шахрайства. Відхилення від стандартних процедур може виникати з наступних причин:

Недостатня Обізнаність або Навчання Персоналу: співробітники можуть не бути повністю обізнані зі стандартними процедурами або можуть не отримати належного навчання.

Звичка або Зручність: співробітники можуть віддавати перевагу власним звичним методам роботи замість стандартних процедур.

Відсутність Ясності у Процедурах: нечіткі або складні для розуміння стандартні процедури можуть призвести до їх неправильного виконання.

Загальний патерн використання ПМ в зазначених випадках матиме наступний вигляд:

Аналіз Логів Діяльності: ПМ використовує логи діяльності для аналізу фактичних шляхів робочих процесів. Це дозволяє виявити відхилення від стандартних процедур шляхом порівняння реальних даних зі стандартними моделями процесів.

Ідентифікація Причин Відхилень: аналіз даних допомагає зрозуміти, чому відбуваються відхилення, визначивши, наприклад, чи є проблеми з навчанням, комунікацією або зрозумілістю процедур.

Розробка Механізмів Виявлення та Попередження: за допомогою ПМ можна розробити системи, які будуть автоматично виявляти відхилення від стандартних процедур і повідомляти про них відповідальних осіб або керівників.

Внесення Змін у Процедури та Навчання: на основі отриманих даних можуть бути внесені зміни у стандартні процедури для їх спрощення та

поліпшення зрозумілості, а також розроблені нові програми навчання для персоналу.

Моніторинг Після Внесення Змін: ПМ дозволяє продовжувати моніторинг процесів після внесення будь-яких змін, щоб переконатися, що відхилення від стандартних процедур зменшуються або усуваються.

В [2] запропоновано фреймворк для визначення параметрів процесу і питань, за допомогою яких їх можна коректно визначити. Вони зазначені в Таблиці 2.

Бізнес-орієнтовані Питання	Застосування ПМ
Прозорість	
Відкриття моделі процесу	Як виконуються випадки процедурного процесу?
	Як виконуються випадки декларативного процесу?
	Як виконуються випадки гібридного процесу?
Покращення моделі процесу	Як можна відремонтувати модель процесу, щоб краще відображати реальне виконання процесу?
	Як можна розширити модель процесу додатковою інформацією, записаною в журналах подій?
	Як можна покращити зрозумілість видобутих моделей процесів?

Майнінг соціальних мереж	Які взаємовідносини між ресурсами, що беруть участь у процесі?
Майнінг цілей	Які цілі процесу?
Майнінг правил	Які правила прийняття рішень визначають, яким шляхом йде випадок?
Ефективність	
Продуктивність процесу	Яка продуктивність випадку за часом?
	Яка продуктивність випадку за ресурсами?
	Яка продуктивність випадку за якістю?
	Як на продуктивність випадку впливають інші фактори?
	Яка продуктивність кількох пов'язаних процесів?
	Як можна оптимізувати продуктивність випадку?
	Як змінюється продуктивність випадку з часом?
Якість	
Аналіз варіантів процесу	Які основні варіанти процесу?
	Які схожості між двома або більше варіантами процесу?

	Які відмінності між двома або більше варіантами процесу?
Майнінг відхилень процесу	Чому деякі випадки відхиляються від нормального потоку?
Відповідність	
Перевірка відповідності процесу	Де випадок відрізняється від моделі процесу?
	У разі невідповідного випадку, який найближчий відповідний випадок?
Моніторинг відповідності процесу	Чи відповідає поточний випадок деяким попередньо визначеним правилам та обмеженням?
	Які причини невідповідності у поточному випадку?
Гнучкість	
Передбачуваний моніторинг	Який прогнозований залишковий час поточного випадку?
	Які прогнозовані затримки поточного випадку?
	Яка наступна дія/дії поточного випадку?
	Які прогнозовані часи очікування поточного випадку?

	Які прогнозовані наслідки поточного випадку?
	Які прогнозовані ризики поточного випадку?
	Які прогнозовані витрати поточного випадку?
	Які прогнозовані показники продуктивності поточного випадку?
Концептуальна дрейф	Як змінилося поведінка процесу з часом?
Прескриптивний моніторинг	Який рекомендований шлях виконання поточного випадку?
	Яке рекомендоване розподілення ресурсів?
	Хто рекомендований виконавець процесу?
	Коли слід втручатися, щоб збільшити ймовірність позитивного результату?

Таблиця 2. - Визначення фреймворку [складено автором]

Висновок до 1 розділу

В даному розділі було розглянуто концепцію багатовимірного куба та типи багатовимірних даних. Розглянуто архітектуру побудови систем на основі ПМ. Був проведений аналіз метода, виявлені сильні та слабкі сторони методології, практичні кейси використання на підприємстві. Дослідження було вирішено проводити на основі ПМ. Метод, ресурси та нагальні потреби дозволяють розробити прототип та імплементувати таку систему.

РОЗДІЛ 2

МОДЕЛЬ СИСТЕМИ ТА ОРГАНІЗАЦІЯ ПРОЕКТА

2.1 Аналіз структури підприємства

Компанія, в якій ключові ідеї кваліфікаційної роботи пройшли апробацію, займається розробкою мобільних відеоігор та видавництвом їх на всіх ключових цифрових маркетплейс. Продуктова ІТ-компанія, яка спеціалізується на розробці відеоігор, виконує ряд ключових завдань та процесів, щоб створити та випустити гру.

Для розуміння типових процесів розробки програмного забезпечення розглянемо The Software Development Life Cycle (SDLC) - стандарт в плануванні розробки систем зі сталим життєвим циклом. SDLC - це метод, що використовується в системній інженерії, інформаційних системах та програмній інженерії для опису методу планування, створення, тестування та розгортання інформаційної системи. Він застосовний до різних конфігурацій апаратного та програмного забезпечення, оскільки система може складатися лише з апаратного забезпечення, лише з програмного забезпечення або з їх комбінації.

Основні етапи SDLC включають:

1. Планування та аналіз. На цьому етапі збираються бізнес-вимоги від клієнтів або зацікавлених сторін, оцінюється можливість створення продукту, потенційний дохід, вартість виробництва та потреби кінцевих користувачів.
2. Визначення вимог Тут інформація, зібрана під час етапу планування та аналізу, перетворюється на чіткі вимоги для команди розробників. Розробляються важливі документи, такі як специфікація вимог до програмного забезпечення (SRS), документ використання випадків та матриця відстеження вимог.
3. Дизайн. На цьому етапі план та візія розробляються у документ дизайну програмного забезпечення (SDD), який включає системний дизайн, мову

програмування, шаблони, платформу використання та заходи безпеки програмного забезпечення. Часто включає створення прототипу моделі.

4. Розробка. Команда розробників ділить проект на модулі програмного забезпечення та перетворює вимоги програмного забезпечення у код. Цей етап може бути тривалим і часто поєднується з етапом тестування.

5. Тестування. Команда забезпечення якості проводить валідаційне тестування, щоб переконатися, що програмне забезпечення функціонує належним чином і відповідає своїй меті. Типи тестування включають тестування продуктивності, функціональне тестування, тестування безпеки, модульне тестування, тестування зручності користування та приймальне тестування.

7. Розгортання. На цьому етапі кінцевий продукт поставляється призначеним користувачам. Цей процес можна автоматизувати та запланувати в залежності від типу розгортання, як-от оновлення функцій або створення абсолютно нового програмного забезпечення.

8. Технічне обслуговування. Останній етап передбачає виправлення помилок та проблем, які могли бути пропущені раніше. Цей етап може призвести до вдосконалень і іноді повернення до першого кроку SDLC для нових функцій або оновлень.

SDLC PHASES

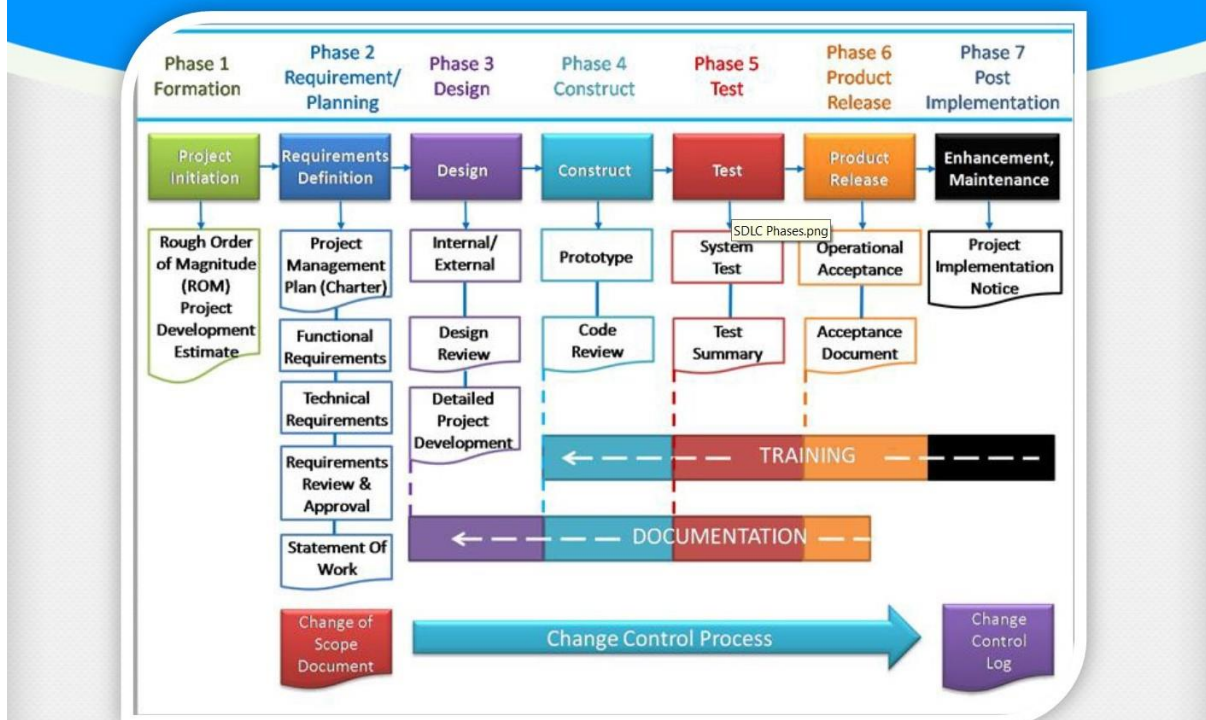


Рисунок 2.1 - Візуалізація концепції SDLC

Якщо ІТ-компанія займається не тільки розробкою, але й видавництвом відеоігор, як є в нашому випадку, до її діяльності додаються наступні ключові функції:

9. Ліцензування та Партнерства. Видавництво може займатися встановленням партнерських відносин з іншими розробниками та отриманням прав на видання їх ігор. Це може включати переговори, підписання контрактів та управління правами на інтелектуальну власність.

10. Фінансування Розробки: Видавництво часто виступає в ролі інвестора, забезпечуючи фінансування розробки ігор, які вони планують видати. Це включає бюджетування, управління витратами та контроль за фінансовими потоками проектів.

11. Розповсюдження ігор. Видавництво відповідає за розміщення гри на різних платформах та фізичних носіях. Це включає ведення переговорів з платформами, такими як Steam, PlayStation, Xbox, а також виробництво та дистрибуцію фізичних копій гри.

12. Маркетинг та Промоція на Зовнішніх Ринках. Видавництво також відповідає за маркетинг та промоцію гри на різних ринках, включаючи міжнародні. Це може включати створення рекламних кампаній, участь у виставках та ігрових конференціях, а також роботу з пресою та інфлюенсерами.

13. Клієнтська Підтримка та Спільнота. Видавництво також може займатися підтримкою спільноти гравців, управлінням форумів, відповідями на запити користувачів та збором зворотного зв'язку для подальшого вдосконалення ігор.

14. Управління Правами на Контент. Видавництво займається захистом прав на інтелектуальну власність та управлінням ліцензіями, а також контролює розповсюдження контенту для запобігання порушень авторських прав.

Всі ці аспекти разом формують повний цикл діяльності компанії, яка не тільки розробляє, але й видавничко займається відеоіграми, забезпечуючи їх виход на ринок та підтримку після релізу.

Перелічені в першому розділі роботи плюси використання Process Mining серед складних інформаційних систем багатовимірних даних налічуючих багато різнотипових сховищ обумовлюють вибір саме цієї моделі для дослідження в цій роботі. Дослідження проводилось на базі великого підприємства виробляючого цифрову продукцію. Підприємство підходить нам за рахунок своєї складної ієрархічної структури, великої кількості внутрішніх бізнес процесів працюючих з різними типам сховищ даних. Виклики в вигляді циклічних аудитів покликаних до оптимізації та автоматизації процесів потребують надійної унітарної моделі зберігання даних для аналізу цієї складної системи. Декомпозиція процесів описана в наступних підпунктах допомагає нам обрати тестовий бізнес-процес використовуючи кілька різних джерел даних, на ньому і буде досліджуватися наша модель. За наявності метрик вказуючих на успіх та доцільність моделі, підприємство може масштабувати використання моделі від окремих бізнес-процесів до побудови архітектури зберігання даних спільної для всього підприємства.

2.2 Опис спроектованої системи

Предметом нашого дослідження є інфраструктура сховища даних. З нього і почнемо. Компанія притримується парадигми використання ETL в обробці даних. ETL" в контексті обробки даних є аббревіатурою, яка означає "Extract, Transform, Load" (Витягування, Трансформація, Завантаження). Це процес, який використовується для збору даних з різних джерел, їх перетворення в структурований формат і завантаження в сховище даних для подальшого аналізу або звітності. Ось основні етапи ETL:

Витягування (Extract) - на цьому етапі дані збираються з різних джерел. Це можуть бути бази даних, файли, хмарні сховища тощо. Ціль полягає в тому, щоб ефективно зібрати всі необхідні дані для подальшої обробки.

Трансформація (Transform) - після витягування дані трансформуються у формат, необхідний для аналізу або звітності. Цей етап може включати очищення даних (наприклад, видалення дублікатів або коригування помилок), нормалізацію, зміну формату, об'єднання даних з різних джерел та інші перетворення.

Завантаження (Load) - нарешті, трансформовані дані завантажуються в цільову систему або сховище даних. Це може бути база даних, система для великих даних, хмарне сховище тощо.

ETL є ключовим компонентом в бізнес-аналітиці, управлінні даними та інтелектуальному аналізі даних, оскільки він дозволяє організаціям ефективно обробляти великі об'єми даних з різноманітних джерел.

Різноріччя БД спирається на хмарне "сховище" Snowflake. Snowflake - це хмарна платформа для зберігання та аналізу даних, яка надає широкі можливості для обробки великих обсягів даних. Ось основні аспекти того, як працює Snowflake:

Роздільне зберігання та обчислення: snowflake використовує унікальну архітектуру, де зберігання даних і обчислювальні ресурси (відомі як

"warehouses") віддалені одне від одного. Це дозволяє масштабувати зберігання та обчислення незалежно, оптимізуючі вартість та продуктивність.

Сховище даних у хмарі: дані зберігаються в хмарі у вигляді снігопаду (часто у форматі стовпцевого сховища), що забезпечує високу ефективність читання та оптимізацію для аналітичних запитів.

Підтримка різних форматів даних: snowflake може зберігати різноманітні формати даних, включаючи JSON, Avro, XML, Parquet та інші.

Віртуальні склади: обчислювальні ресурси у Snowflake називаються "віртуальними складами" і вони відповідають за виконання запитів до даних. Користувачі можуть масштабувати ці ресурси вгору або вниз залежно від потреби, що забезпечує гнучкість та ефективність використання.

Ізоляція ресурсів: кожен віртуальний склад працює незалежно, що усуває конкуренцію за ресурси та забезпечує стабільну продуктивність.

Автоматичне та ручне масштабування: snowflake дозволяє автоматично масштабувати обчислювальні ресурси вгору або вниз залежно від навантаження, а також дозволяє користувачам ручно налаштовувати ресурси.

Зашифроване зберігання: всі дані у Snowflake автоматично зашифровані.

Контроль доступу: snowflake пропонує розширені можливості для управління доступом та безпекою даних.

Підключення до інструментів аналізу даних: snowflake легко інтегрується з різними інструментами для аналізу.

Впровадження використання Process Mining на підприємстві потребує великих ресурсних вкладень та пов'язано з великими ризиками для безпеки даних та життєдіяльності виробничих процесів в цілому. Тому для дослідження обирається бізнес-процес, зміни в якому суттєво не загрожують роботі бізнесу, і одночасно має суттєві вади та потребує покращення в загальній перспективі.

Для дослідження моделі використовуємо один з бізнес-процесів, який давно потребує переробки. Бізнес-процес описує систему обробки зворотнього зв'язку, або Second Level Support. Другий рівень підтримки (Second Level Support), також відомий як рівень 2 або L2, є важливою частиною структури ІТ-підтримки клієнтів. Цей рівень займається складнішими проблемами, які не можна вирішити на першому рівні підтримки. Характеристикою Другого рівня підтримки є більш глибокі технічні знання та досвід, необхідні для роботи з більш складними технічними проблемами. Другий рівень підтримки відповідає за більш глибоку технічну підтримку. Це включає в себе виправлення неочевидних програмних помилок, вирішення невеликих мережевих проблем та розгляд складніших запитів, пов'язаних з обліковими записами, які можуть вимагати більш глибокого доступу до системи. Техніки, або PAU менеджери, на Другому рівні підтримки зазвичай розвивають навички, необхідні на першому рівні. Їм потрібні міцні технічні знання щодо продуктів та послуг компанії, глибоке технічне розуміння стандартів і практик галузі, а також сильні навички вирішення проблем та аналітичні здібності. Ефективні комунікаційні навички та вміння технічного письма також є важливими для цієї ролі.

Завдання на цьому рівні включають пошук рішень зі старих випадків за допомогою бази знань, ведення життєвого циклу інциденту на допоміжному столі, збір інформації про проблему для передачі її в інші департаменти наприклад в відділ розробки або відділ електронної комерції, якщо це необхідно, керівництво клієнтів у вирішенні проблем, моніторинг доступних інструментів для виявлення проблем (наприклад, мережі або сервера)

Для розуміння поточної моделі бізнес-процесу, за потреби її декомпозиції та розробки нової моделі потрібно обрати прості, але універсальні інструменти прототипування, адже є ймовірність, що на практиці модель може давати погіршність та потребувати доробки.

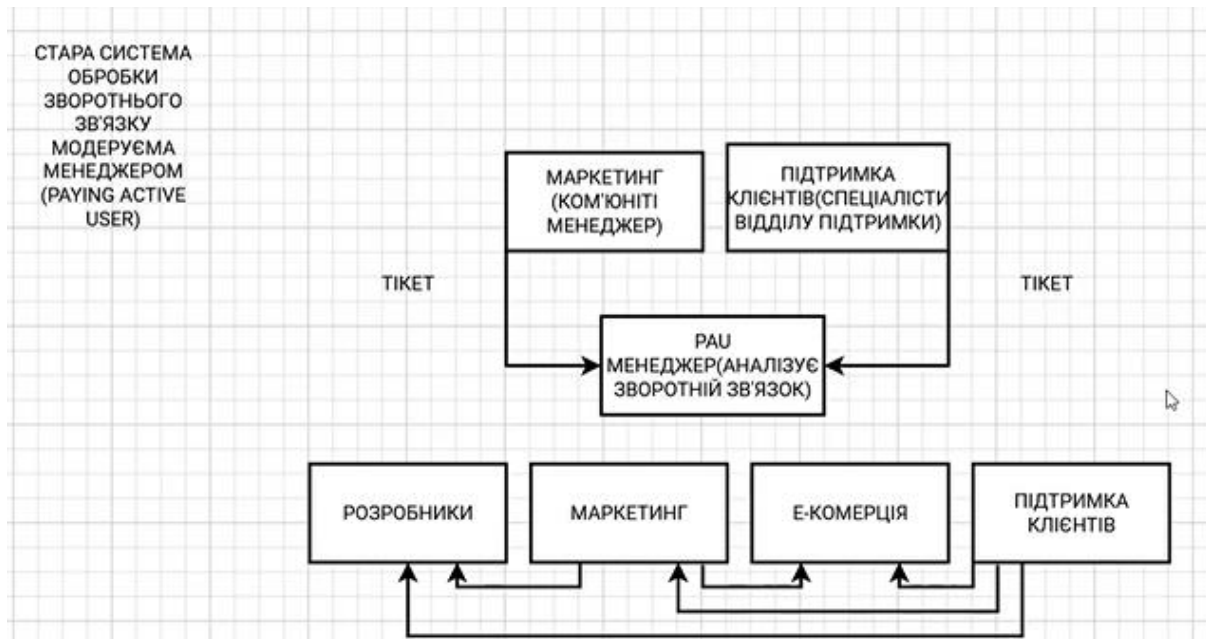


Рисунок 2.2 - модель “Як є”

Проблема процесу наразі виглядає наступним чином:

1. Це мануальний процес, що потребує фахівця-селектора (РАУ менеджер), який повинен досить швидко знаходити рішення, або відповідного фахівця з інших департаментів.
2. Бізнес-процес не інтегрований в хмарне сховище даних Snowflake. Це зумовлює проблеми при аналізі та ретроспектив проблем. Також страждає комунікація відділів, через відсутність можливості швидко передивитися дані, без портування інформації з локальних баз даних до корпоративного хмарного сховища.
3. Аналітична інформація зберігається в різних типах сховищ. Наприклад, робота з логами пристроїв клієнтів не синхронізується з даними трекінгу ігрової активності клієнта, бо зберігається в окремих сховищах, не маючих синхронізації в одній системі.
4. Задачі мають своє ПО для трекінгу, де документуються тільки проблеми від підтримки та рішення від РАУ Менеджера.
5. Фактор кадрів та legacy архів знань. При зміні фахівця на позиції РАУ Менеджера, новим кадрам важко розібратися та кластеризувати проблеми. Наявність бібліотеки з переліком проблем не гарантує знаходження

рішення, і працює за принципом legacy практик - дані можуть бути вже не актуальні, або складними до сприйняття.

6. Ведення бібліотеки з кейсами звернень займає багато часу, часто це одноразові рішення, які більше не дублюються, що формує високий поріг входження.

Загалом цей список проблем призводять до проблем в комунікації відділів та періодичному «простою» звернень клієнтів, що негативно впливає на якість обслуговування та залученість клієнтів.

Виділимо кілька з них:

1. Невірна інтерпретація менеджером важливої інформації з приводу стану гри без моніторингу профільних фахівців може сприяти потенційним проблемам зросту кількості критичних помилок та загальній якості продукту
2. Довга відповідь клієнтам, є неприпустимою помилкою в системі обслуговування будь-якої комерційної установи.
3. Падіння KPI відділа підтримки та ком'юніті-модераторів, що негативно впливає на загальну вмотивованість та задоволеність своїм місцем роботи в співробітників цих департаментів.

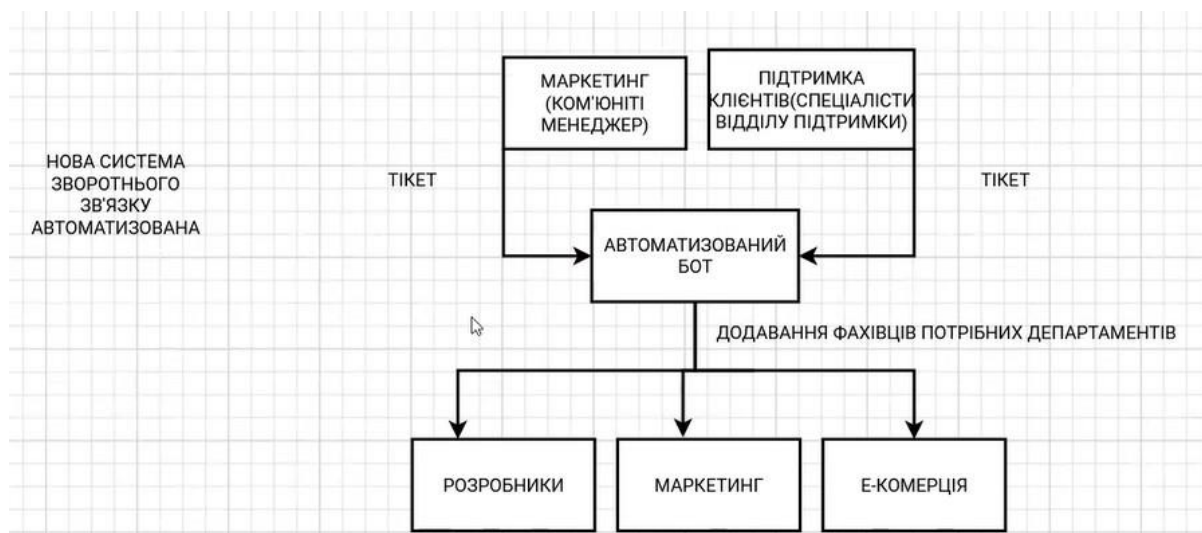


Рисунок 2.3 - модель як повинно бути

Модель потребує перебудови та за автоматизації. В якості покращення пропонується залучити в систему автоматизованого бота, який оброблятиме

скарги клієнтів в якості 2L. Людина та бот будуть працювати паралельно, що дасть змогу поступово підняти рівень якості цього бізнес процеса.

Фінкція бота полягає в семантичному аналізі оформлених заявок-тікетів від відділу підтримки та ком'юніті-модераторів. Оброблені заявки отримують Виконавців в вигляді конкретних спеціалістів, яких додає бот в системі трекінгу тікеті Jira.

Для коректної роботи бота потрібна попередня обробка великого масиву даних, задля збору семантичного ядра, яке буде валідно індексувати тікети та додавати потрібних фахівців. В цьому нам може допомогти методологія обробки великих масивів даних ПМ.

2.3 Алгоритм побудови транзитної обробки даних

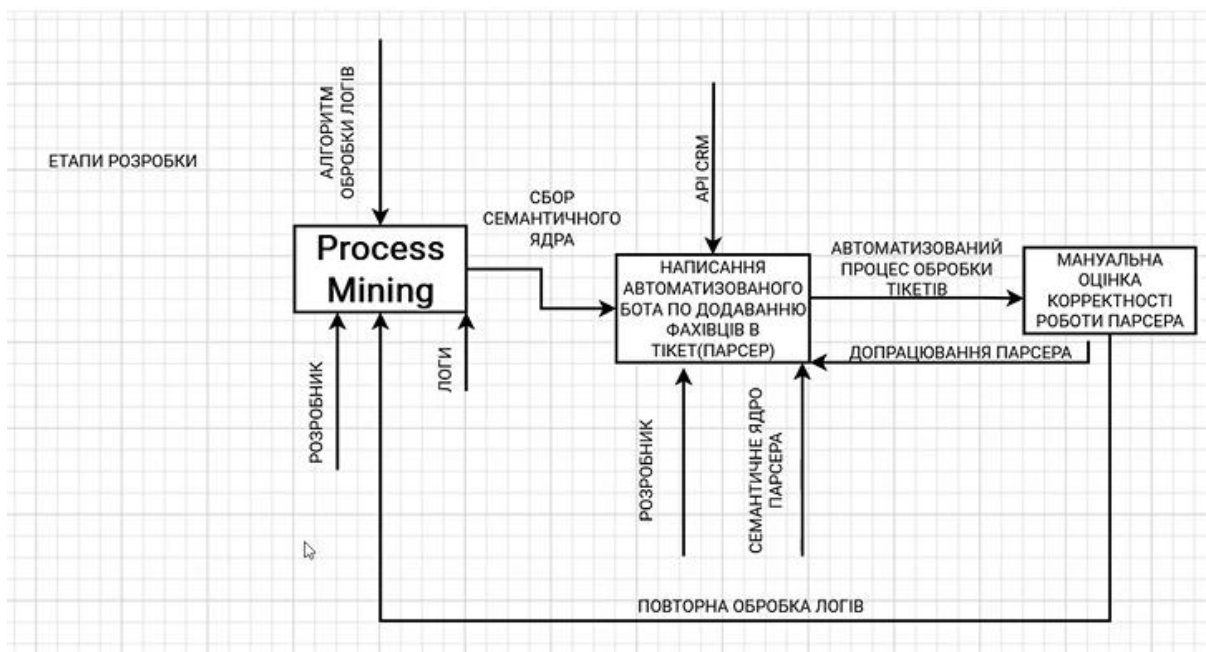


рис 2.4 транзитна обробка даних

Для обробки великого масиву даних зі скарг клієнтів, використовуючи Process Mining і модель ІІІ на базі scikit-learn, можна використати наступний бізнес-процес:

1. Збір даних:
 - a. Збір усіх доступних Jira тікетів та листувань з клієнтами.
2. Попередня обробка даних:
 - a. Очистка даних: видалення несуттєвої інформації, коригування помилок, стандартизація форматів.
 - b. Сегментація тексту: поділ даних на логічні частини (наприклад, за датами, категоріями проблем).
3. Аналіз даних за допомогою Process Mining:
 - a. Використання Process Mining для виявлення, моніторингу та оптимізації реальних процесів на основі зібраних даних.
 - b. Виявлення шаблонів і тенденцій у скаргах клієнтів.
4. Розробка та навчання моделі ІІІ:
 - a. Використання scikit-learn для створення моделі машинного навчання.
 - b. Навчання моделі на оброблених даних для класифікації, прогнозування або інших завдань.
5. Екстракція семантичного ядра:
 - a. Аналіз тексту скарг для виявлення ключових термінів та концепцій.
 - b. Створення семантичного ядра, яке відображає основні питання та проблеми клієнтів.
6. Інтеграція та використання отриманих даних:
 - a. Інтеграція висновків у бізнес-процеси для покращення обслуговування клієнтів.
 - b. Розробка стратегій відповіді на найбільш поширені та критичні скарги.
7. Моніторинг та оцінка:

- a. Регулярний моніторинг ефективності впроваджених змін.
 - b. Оцінка впливу на задоволеність клієнтів та оптимізація процесу на основі зворотного зв'язку.
- 8. Наступний етап розробка парсера для автоматичного призначення відповідальних фахівців у тикет в системі трекінгу Jira.
- 9. Інтеграція семантичного ядра з Jira:
 - a. Впровадження семантичного ядра в систему Jira.
 - b. Розробка інтерфейсу для зв'язку семантичного ядра з базою даних Jira.
- 10. Розробка парсера для аналізу скарг:
 - a. Створення парсера, який аналізує текст скарг на наявність ключових слів із семантичного ядра.
 - b. Реалізація алгоритмів для розпізнавання контексту та значення ключових слів.
- 11. Класифікація скарг:
 - a. Використання парсера для визначення категорії кожної скарги.
 - b. Автоматичне присвоєння скарг до відповідних категорій на основі семантичного аналізу.
- 12. Визначення відповідальних працівників:
 - a. Встановлення відповідності між категоріями проблем і відповідними спеціалістами або відділами в IT-компанії.
 - b. Створення бази даних зі спеціалізаціями та відповідальностями працівників.
- 13. Автоматичне призначення скарг:
 - a. Розробка системи, яка автоматично призначає скарги відповідним працівникам у Jira на основі аналізу ключових слів.
 - b. Інтеграція цієї системи з процесом створення та управління тикетами в Jira.
- 14. Моніторинг та оптимізація процесу:

- a. Впровадження моніторингу ефективності системи автоматичного призначення.
- b. Збір зворотного зв'язку від працівників та коректування алгоритмів для поліпшення точності призначень.

15. Навчання та адаптація:

- a. Регулярне оновлення семантичного ядра та алгоритмів парсингу для врахування нових термінів та змін у ІТ-галузі.
- b. Навчання працівників, як ефективно працювати з оновленою системою.

Цей процес передбачає тісну взаємодію між розробниками, аналітиками даних та відділом технічної підтримки, а також постійне оновлення та адаптацію системи під потреби бізнесу.

Висновки до 3 розділу.

Було розглянуто концепції роботи студії розробки відеоігор. Підібрано валідний для модернізації бізнес-процесу. Розглянута застаріла модель та запропонована нова, на базі якої ми можемо розглянути транзит багатовимірних даних. Розроблено систему збору, обробки та інтерпретації даних. Система передбачає збір та аналіз даних з Jira тикетів та листувань з клієнтами за допомогою Process Mining і моделі ІІІ, створеної на базі scikit-learn. Метою цього процесу було створення семантичного ядра з оброблених скарг для подальшого використання в бізнес-процесі, спрямований на автоматизацію призначення скарг клієнтів відповідним працівникам ІТ-компанії у системі Jira. Цей процес включав інтеграцію семантичного ядра з Jira, розробку парсера для аналізу ключових слів у скаргах, автоматичне призначення скарг, моніторинг і оптимізацію системи.

РОЗДІЛ 3

Аналіз циклу транзитної обробки багатовимірних даних

3.1 Агрегація даних

Агрегація даних це обов'язковий етап перед ПМ, адже для цього необхідно зібрати валідний датасет даних. В нашому випадку, дані - це тікети в системі Jira і звернення гравців на електронну пошту.

Jira – це популярна платформа для управління проектами та відстеження помилок, розроблена компанією Atlassian[12]. Вона широко використовується в сфері розробки програмного забезпечення для відслідковування завдань, помилок і запитів на функціонал. Jira дозволяє командам організовувати роботу, відстежувати прогрес і співпрацювати над проектами. Система пропонує гнучкість у налаштуванні робочих процесів, інтеграції з різними зовнішніми сервісами та забезпечує потужні інструменти для створення звітів та аналітики, що робить її вибором для багатьох компаній різного масштабу. В нашому випадку, через систему Jira також проводиться процес обробки користувацьких запитів, пов'язаних з проблемами синхронізації в грі. Так як продукт мультиплатформений, обробка таких запитів є розповсюдженим бізнес-процесом, який потребує значних ресурсів. Зазвичай дані по подібним зверненням заносяться із сторонніх систем. Друге джерело даних для цього бізнес-процесу - це електронна пошта, куди потрапляють прямі запити від гравців. Іноді для певного звернення на пошту вже є згенерований тікет, іноді його доводиться заводити в ручному або напівавтоматичному режимі.

Для цього етапу валідним буде наступний порядок дій:

Крок 1: Збір Даних

Інтеграція з Джерелами Даних:

Налаштувати інтеграцію з електронною поштою для автоматичного збору всіх звернень гравців.

Налаштувати інтеграцію з Jira для збору даних про всі створені тікети.

Фільтрація Даних:

Визначити ключові слова та фрази, що стосуються синхронізації в грі (наприклад, "проблема синхронізації", "синхронізація в грі" тощо).

Використати ці ключові слова для фільтрації звернень, щоб відокремити ті, що стосуються синхронізації.

Крок 2: Попередня Обробка Даних

Структурування Даних:

Витягти з електронних листів та тікетів важливу інформацію, таку як дата звернення, електронна адреса або ім'я користувача, опис проблеми, статус тікету тощо.

Чищення Даних:

Видалити дублікати звернень.

Виправити помилки у даних (наприклад, неправильні часові мітки).

Об'єднання Даних:

Об'єднати дані з різних джерел в один централізований набір даних для подальшого аналізу.

Дані із джерел необхідно витягти і відфільтрувати. В даному випадку нас цікавлять виключно звернення, пов'язані з синхронізацією, тому фільтрацію можна виконати одразу після налаштування інтеграції, сформувавши вибірку коректних ключових слів, за яким відбувається пошук.

Сформуємо критерії полів даних тікетів в джири і інформації з електронних листів - в даному випадку це буде ідентифікатор тікета, ідентифікатор листа(якщо є), заголовок, опис, час звернення, хто із працівників був доданий до тікета і з якого він департаменту, а також часовий відбиток події. Так як в одному тікеті може відбуватись більше ніж одна подія(в нашому випадку подія це додавання працівника до тікета або ж його відписки від тікета, що є свідченням не релевантного додавання працівника), то тікет джири будемо вважати за один бізнес-процес, підписку або відписку будемо вважати за подію, в кожній з яких має бути максимально можлива кількість заповнених полів.

Налаштуємо доступ до даних. Jira пропонує REST API, який дозволяє програмно отримувати дані з системи. Правильно сформований запит дозволяє сформувати релевантне вивантаження даних. Також, так-як Jira це корпоративне програмне забезпечення, для доступу до його API необхідно скористатись аутентифікацією, наприклад за API ключем або логіном і паролем.

Схожий набір дій проводиться і з електронною поштою. В даному випадку валідно використання налаштованого поштового агента і парсинг даних, реалізований мовою пайтон є оптимальним рішенням.

Створимо запити до API з урахуванням комплексності запитів, доцільно скористатись комбінацією із REST API і Jira Query Language (JQL) для створення складних запитів, щоб фільтрувати тікети за різними критеріями.

Аналогічно робиться функціонал збору даних з електронної пошти. В даному випадку необхідно підключити поштовий агент і налаштувати вивантаження даних за допомогою скрипта мовою Python.

1. Вибір поштового агента, залежно від поштового сервісу (наприклад, Gmail, Outlook тощо).
2. Для налаштування доступу до пошти, підключається IMAP у налаштуваннях поштового акаунта та генеруються спеціальні паролі для програм, якщо це необхідно (наприклад, для Gmail).

3. У Python розпаковуються необхідні бібліотеки, такі як imaplib та email.
4. Автентифікація та підключення потребує скрипт для підключення до поштового сервера за допомогою імені користувача та пароля.
5. Вибір поштової скриньки та фільтрація повідомлень.
6. Зчитування та обробка повідомлень. Тепер можемо отримувати текстовий вміст, вкладення тощо.
7. Вивантаження даних в сховище snowflake.

Після інтеграції і опису запитів необхідно відфільтрувати отримані дані за ключовими значеннями, і відправити отримані данні в базу даних, Snowflake. Вже звіди будуть опрацьовувати відфільтровані данні. Окрім цього, необхідний крок перевірки нових записів на дублювання в системі, а також інтегрувати різні за джерелом походження дані між собою у випадку, якщо така можливість існує.

В даному випадку вищезазначені кроки підходять під класифікацію як елементи ETL, принципа, що розшифровується як Витяг, Перетворення та Завантаження

3.2 Приклад реалізації

Для реалізації системи процес майнінгу прийнято рішення в якості основного інструменту використати мову програмування Python, з ключовими бібліотеками pm4py і scikit-learn. Для розділу дата-агрегації розроблено скрипт мовою python для реалізації процесу ETL.

Розглянемо детальніше реалізацію скрипта для ETL.

Спочатку імпортуємо необхідні бібліотеки

```
import snowflake.connector
from jira import JIRA
import imaplib
import email
```

Рисунок 3.1 - Необхідні імпорти для скрипту ETL

підключимось до Jira

```
# Параметри підключення до Jira
jira_options = {'server': 'https://domain.atlassian.net'}
jira = JIRA(options=jira_options, basic_auth=('email', 'token'))

# Зчитування тикетів
issues = jira.search_issues('project=PROJECT_KEY')
```

Рисунок 3.2 - Підключення до Jira

Далі нам необхідно зчитати тикети. Для цього загрегуємо необхідні нам дані

```
for issue in issues:
    jira_data.append({
        'case_id': issue.key,
        'header': issue.fields.summary,
        'description': issue.fields.description,
        'assignee': issue.fields.assignee,
        'added': issue.fields.added,
        'timestamp': issue.fields.timestamp
    })
```

Рисунок 3.3 - Агрегація даних з Jira

Тепер підключимось до поштового сервера


```

# Підключення до поштового сервера
mail = imaplib.IMAP4_SSL('imap.mailserver.com')
mail.login('email', 'password')
mail.select('inbox')

# Зчитування листів
type, data = mail.search(None, 'ALL')
mail_ids = data[0]

```

Рисунок 3.4 - Підключення до поштового сервера

На даному етапі нам необхідно агрегувати дані з пошти

```

email_data = []
for num in mail_ids.split():
    typ, data = mail.fetch(num, '(RFC822)')
    raw_email = data[0][1]
    # Розбір листа
    msg = email.message_from_string(raw_email.decode('utf-8'))
    email_data.append({
        'subject': msg['subject'],
        'body': msg.get_payload(decode=True),
    })

```

Рисунок.3.5 - Агрегація даних з пошти

Тепер необхідно підключитись до бази даних і генерувати запит на вставку, за

```
conn = snowflake.connector.connect(  
    user='USER',  
    password='PASSWORD',  
    account='ACCOUNT',  
    warehouse='WAREHOUSE',  
    database='DATABASE',  
    schema='SCHEMA'  
)  
cur = conn.cursor()  
insert_query = """  
INSERT INTO tickets (case_id, header, description, timestamp)  
VALUES (%s, %s, %s, %s)  
"""
```

Рисунок 3.6 - Підключення до БД

На даному етапі необхідно пройти циклом по агрегованим даним і вставити їх у таблицю в БД за допомогою запиту на вставку, згенерованого на попередньому кроці[14].

```
for record in jira_data:  
    cur.execute(insert_query, (record['case_id'], record['header'], record['description'], record['timestamp']))  
  
for record in email_data:  
    cur.execute(insert_query, (record['case_id'], record['subject'], record['body'], record['timestamp']))  
conn.commit()  
cur.close()  
conn.close()
```

Рисунок 3.7 - Запис агрегованих даних в БД

Пройшовши цими кроками, ми описуємо скрипт для агрегації даних в БД snowflake, і тепер можемо перейти до процес майнінгу

Розглянемо детальніше реалізацію моделі ПМ

Спочатку організуємо імпорти і підключення до таблиці БД Snowflake

```

import os
import pandas as pd
import snowflake.connector
from sklearn.preprocessing import MinMaxScaler
from pm4py.objects.conversion.log import converter as log_converter
from pm4py.algo.discovery.inductive import algorithm as inductive_miner
from pm4py.visualization.petri_net import visualizer as pn_visualizer

# Налаштування з'єднання до Snowflake
conn = snowflake.connector.connect(
    user='USERNAME',
    password='PASSWORD',
    account='ACCOUNT',
    warehouse='WAREHOUSE',
    database='DATABASE',
    schema='SCHEMA'
)

```

Рисунок 3.8 - Іморти і підключення до БД

Завантадимо данні з БД Snowflake в дата фрейм, проведемо попередню обробку даних, перейменуємо колонки, переведемо часові значення у відповідний формат, за допомогою класу MinMaxScaler з бібліотеки scikit-learn проведемо нормалізацію часових даних[13].

```

# Завантаження даних з Snowflake
query = "SELECT * FROM TABLE" # Оновіть запит відповідно до вашої таблиці
df = pd.read_sql(query, conn)
conn.close()

# Попередня обробка даних
df.rename(columns={'case_id': 'case:concept:name'}, inplace=True)
df['concept:name'] = df['header']
df['timestamp'] = pd.to_datetime(df['timestamp'], format='%Y-%m-%d %H:%M:%S', errors='coerce')
start_time = df['timestamp'].min()
df['timestamp_seconds'] = (df['timestamp'] - start_time).dt.total_seconds()
scaler = MinMaxScaler()
df['timestamp_normalized'] = scaler.fit_transform(df[['timestamp_seconds']])

```

Рисунок 3.9 попередня обробка і нормалізація даних

Наступним кроком буде перетворення датафрейму на журнал подій, виявлення моделі процесу мереж Петрі за допомогою алгоритму індуктивного майнінгу.

```
# Перетворення DataFrame на Event Log
log = log_converter.apply(df)

# Виявлення моделі процесу (Petri Net) за допомогою алгоритму індуктивного майнера
proc_tree_example = inductive_miner.apply(log)
net, initial_marking, final_marking = pm4py.convert_to_petri_net(proc_tree_example)

# Візуалізація моделі процесу (Petri Net)
gviz = pn_visualizer.apply(net, initial_marking, final_marking)
pn_visualizer.matplotlib_view(gviz)
```

Рисунок 3.10 Визначення моделі

Тепер модель сформовано і доступна графічна репрезентація даних. Отриманий об'єм потребує додаткової обробки.

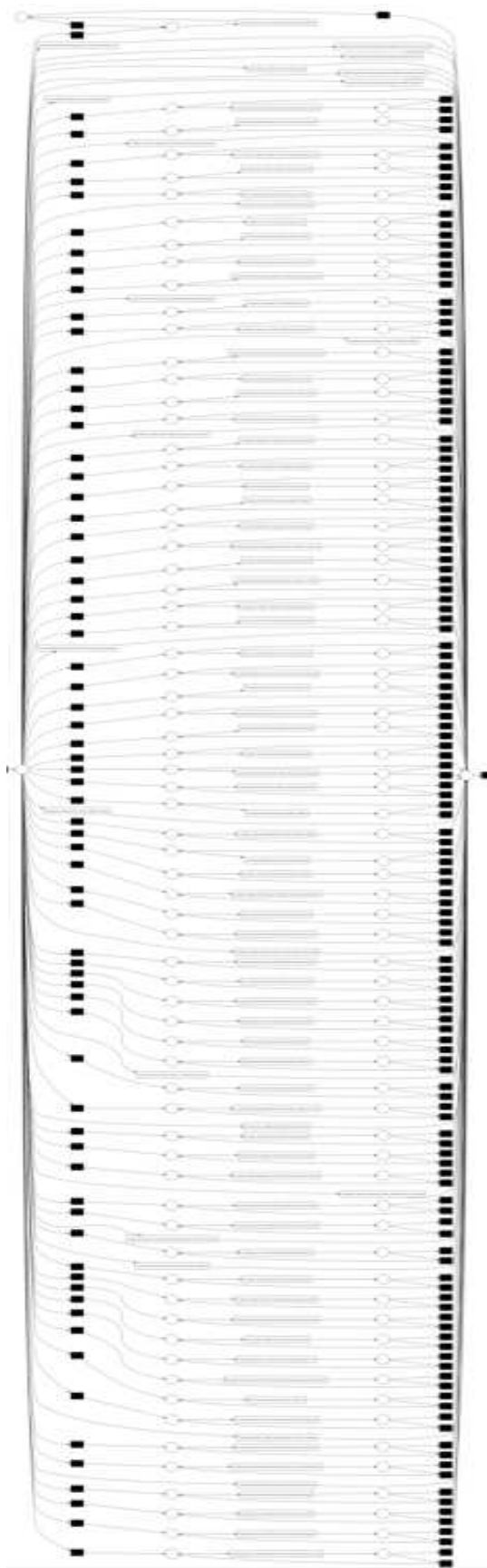


Рисунок 3.11 - Згенеровані мережі Петрі

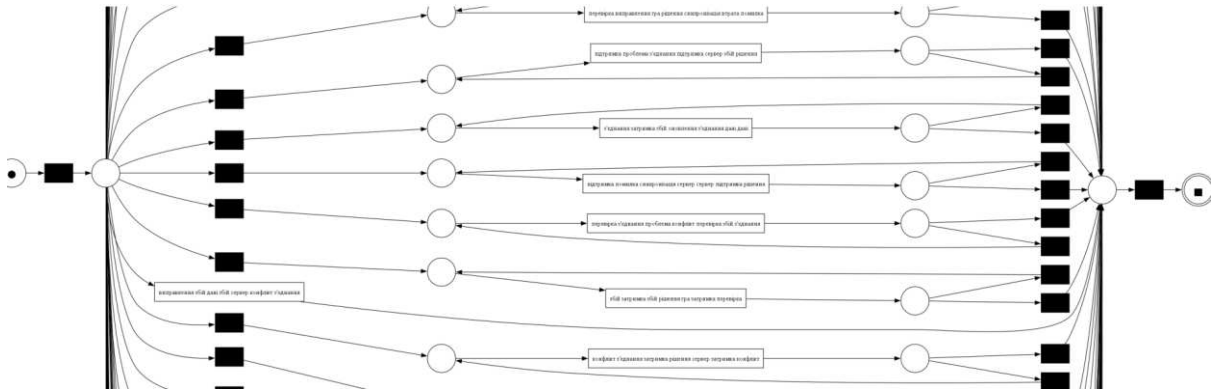


Рисунок 3.12 - Згенеровані мережі Петрі в наближенні.

Таким чином, в результаті виконання скрипту згенеровано процесну модель.

Розглянемо детальніше реалізацію інтеграції ІІІ в модель ПМ

Спочатку імпортуємо необхідні бібліотеки і налаштуємо підключення до таблиці. В даному випадку скористаємось SCHEMA

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.feature_extraction.text import TfidfVectorizer
import snowflake.connector

# Налаштування з'єднання до Snowflake
conn = snowflake.connector.connect(
    user='USERNAME',
    password='PASSWORD',
    account='ACCOUNT',
    warehouse='WAREHOUSE',
    database='DATABASE',
    schema='SCHEMA'
)
```

Рисунок 3.13 - Підключення до таблиці

Тепер вивантажимо дані, векторизуємо їх і додамо ознаки

```

# Завантаження даних з Snowflake
query = "SELECT * FROM TABLE" |
df = pd.read_sql(query, conn)
conn.close()

# Перетворення текстових даних в числові вектори за допомогою TF-IDF
tfidf = TfidfVectorizer(max_features=100)
X_tfidf = tfidf.fit_transform(df['header']).toarray()

# Додавання інших числових/категорійних ознак, якщо потрібно
# Припустимо, assignee вже перетворено в числовий формат
X = pd.concat([pd.DataFrame(X_tfidf), df[['assignee_numeric']]], axis=1)
y = df['added']

```

Рисунок 3.14 - Попередня обробка даних

Розділимо отримані дані на тренувальні і тестувальні набори, скористаємось випадковим лісом в якості класифікатора, спрогнозуємо та оцінимо модель [15-17].

```

# Розділення даних на тренувальні та тестувальні набори
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Використання випадкового лісу як класифікатора
clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(X_train, y_train)

# Прогнозування та оцінка моделі
y_pred = clf.predict(X_test)
print("Точність моделі:", accuracy_score(y_test, y_pred))

```

Рисунок 3.15 - Розділення, класифікація і прогнозування моделі

Підключимось до іншої таблиці, де розміщені вже реальні дані, і виберемо останній запис.

```
# Використання моделі для генерації рекомендації для нового запису
conn = snowflake.connector.connect(
    user='USERNAME',
    password='PASSWORD',
    account='ACCOUNT',
    warehouse='WAREHOUSE',
    database='DATABASE',
    schema='SCHEMA1'
)
query = "SELECT * FROM TABLE ORDER BY timestamp DESC LIMIT 1;"
new_record = pd.read_sql(query, conn)
conn.close()
```

Рисунок 3.16 - Підключення до другої таблиці і вибір останнього запису

Скористаємось отриманою моделлю для генерації нового запису

```
new_record_transformed = tfidf.transform([new_record['header']]).toarray()
new_record_features = pd.concat([pd.DataFrame(new_record_transformed),
                                pd.Series([new_record['assignee_numeric']]), axis=1)
recommended_added = clf.predict(new_record_features)[0]
print("Рекомендоване значення для 'added':", recommended_added)
```

Рисунок 3.17 - Отримання рекомендацій

Запропонована система опрацьовує дані з двох джерел за допомогою процесу ETL, формує модель ПМ, а також за допомогою алгоритмів машинного навчання розробляє рекомендації щодо визначення кращої поведінки щодо додавання нових користувачів в тікет. Переваги запропонованої моделі полягають в широких можливостях отримання і інтерпретації висновків на основі алгоритмів ПМ і машинного навчання. До недоліків варто віднести складність побудови, складність категоризації бізнес процесів і їх параметрів, а також необхідність залучення великих обчислювальних потужностей у випадку масштабізації рішення.

3.3 Практичні рекомендації на основі проведеного дослідження

На основі отриманих результатів можна резюмувати, що дослідження мало практичний сенс[18].

Інтеграція ШІ з ПМ: Використання алгоритмів машинного навчання для глибшого аналізу даних з журналів подій. Це може включати прогнозування майбутніх трендів процесів або ідентифікацію потенційних вузьких місць.

Автоматизація й Оптимізація: Використання ШІ для автоматизації рутинних завдань та покращення взаємодії між різними елементами процесу, що підвищить ефективність.

Підвищення Адаптивності: Розробка адаптивних бізнес-процесів з допомогою ШІ, які можуть самостійно адаптуватися до змін у даних чи умовах ринку.

Інтеграція штучного інтелекту (ШІ) з ПМ забезпечує більш глибокий аналіз і покращення бізнес-процесів. ШІ може використовуватися для аналізу великих обсягів даних з журналів подій, що дозволяє отримати докладніші інсайти про роботу процесів. Це включає виявлення неочікуваних закономірностей, прогнозування майбутніх трендів та виявлення потенційних проблемних зон. Такий підхід може допомогти в автоматизації та оптимізації процесів, підвищуючи ефективність і зменшуючи витрати.

Автоматизація та оптимізація за допомогою штучного інтелекту (ШІ) в контексті ПМ передбачає використання ШІ для покращення бізнес-процесів. Це включає автоматизацію повторюваних завдань, ефективне розподілення ресурсів та виявлення неефективних кроків у процесах. Завдяки аналізу даних з журналів подій, ШІ допомагає виявляти шляхи покращення процесів, що сприяє зниженню витрат та підвищенню продуктивності.

Підвищення адаптивності бізнес-процесів через інтеграцію штучного інтелекту (ШІ) означає впровадження систем, які можуть самостійно реагувати на зміни у даних чи умовах ринку. Це дозволяє бізнес-процесам

швидко адаптуватися до нових вимог, оптимізувати свої дії в реальному часі та забезпечити більш гнучке управління. Адаптивність за допомогою ШІ забезпечує більшу ефективність у вирішенні проблем, прогнозуванні потреб клієнтів та оптимізації ресурсів.

Висновки до 3 розділу

Для реалізації системи процес майнінгу в якості основного інструменту використано мову програмування Python, з ключовими бібліотеками `rm4py` і `scikit-learn`. Для розділу дата-агрегації розроблено скрипт мовою `python` для реалізації процесу ETL.

Запропонована система опрацьовує дані з двох джерел за допомогою процесу ETL, формує модель ПМ, а також за допомогою алгоритмів машинного навчання розробляє рекомендації щодо визначення кращої поведінки щодо додавання нових користувачів в тікет. Переваги запропонованої моделі полягають в широких можливостях отримання і інтерпретації висновків на основі алгоритмів ПМ і машинного навчання. До недоліків варто віднести складність побудови, складність категоризації бізнес процесів і їх параметрів, а також необхідність залучення великих обчислювальних потужностей у випадку масштабізації рішення.

РОЗДІЛ 4 ЕКОНОМІЧНИЙ РОЗДІЛ

4.1 Економічний розрахунок витрат

Відразу потрібно розуміти, що витрати матимуть як одноразовий характер(розробка), так і витрати на постійній основі(оренда серверів, ПО, підтримка)

Для створення смети на проектування та розробку ІТ-проекту на базі ПМ, де залучено бекенд розробника, дата-інженера та дата-аналітика.

План Розробки

Тиждень 1: Аналіз та Планування

1. Зустріч з Замовником (4 години): Збір вимог, обговорення цілей проекту.
2. Аналіз Вимог (16 годин): Бізнес аналітик аналізує вимоги, формулює функціональні та нефункціональні вимоги.
3. Планування Проекту (8 годин): Визначення ресурсів, часових рамок, ризиків.

Тиждень 2-3: Розробка та Тестування

1. Розробка (80 годин): Бекенд розробник працює над створенням коду згідно з вимогами.
2. Аналіз Даних (40 годин): Дата аналітик працює над обробкою та аналізом даних, необхідних для програми.

Тиждень 4: Перевірка, Впровадження та Документація

1. Тестування та Виправлення Помилки (12 годин): Перевірка програми на помилки, їх виправлення.
2. Створення Документації (8 годин): Підготовка технічної та користувацької документації.
3. Підсумкова Зустріч з Замовником (4 години): Демонстрація програми, збір відгуків.

Кошторис

Витрати на Персонал

- Бізнес Аналітик: 40 годин x \$50/год = \$2,000

- Бекенд Розробник: 80 годин x \$60/год = \$4,800
- Дата Аналітик: 40 годин x \$55/год = \$2,200

Інші Витрати

- Ліцензії та Інструменти: \$500
- Зустрічі та Комунікації: \$300
- Непередбачувані Витрати: 10% від загальної суми = \$980

Загальний Кошторис

- Всього: \$2,000 + \$4,800 + \$2,200 + \$500 + \$300 + \$980 = \$10,780

4.2 Економічний розрахунок прибутку

Для розуміння кінцевого прибутку потрібно розуміти, як працює ринок F2P ігор. Приділимо цьому трохи уваги. Фритуплей ігри (free-to-play, F2P) – це відеоігри, які користувачі можуть грати безкоштовно, без необхідності сплачувати за саму гру. Ці ігри зазвичай мають вбудовані механізми монетизації, які дозволяють розробникам заробляти гроші, не зважаючи на безкоштовний доступ до основної гри.

Основні методи монетизації фритуплей ігор включають:

1. Мікротранзакції: гравці можуть купувати віртуальні товари або валюту у грі. Це можуть бути косметичні предмети, які змінюють зовнішній вигляд персонажів, або функціональні предмети, які поліпшують геймплей або дають переваги в грі.
2. Сезонні пропуски та підписки: гравці платять за регулярний доступ до ексклюзивного вмісту, оновлень або бонусів.
3. Реклама: ігри можуть включати рекламу у вигляді банерів, відео реклам, або інтерактивних рекламних вставок. Деякі ігри також пропонують нагороди за перегляд рекламних роликів.
4. Продаж даних: деякі розробники заробляють, продавши аналітичні дані про поведінку гравців рекламодавцям або стороннім компаніям.

Фактори, що впливають на підвищення конверсії

1. Якість геймплею: основою успіху є залучення гравців. Якщо гра цікава, збалансована і надає глибокий і змістовний досвід, гравці більше схильні витратити гроші.
2. Постійне оновлення контенту: регулярні оновлення, додавання нових рівнів, персонажів або викликів зберігають інтерес гравців та спонукають до повернення.
3. Баланс між безкоштовним і платним вмістом: важливо, щоб гравці, які не платять, теж мали доступ до задовільного досвіду гри, в той час як платний вміст повинен додавати значну цінність.
4. Соціальні елементи: можливість гри з друзями або змагання в рейтингових таблицях створює додатковий стимул для покупок у грі.
5. Оптимізація цінності пропозиції: Цінова політика має бути привабливою для гравців, з пакетами пропозицій, які відповідають різним бюджетам і потребам.
6. Психологічні стимули: використання технік, таких як обмежені пропозиції, івенти або системи винагород, стимулює гравців до покупок.
7. Зручність платежів: легкість здійснення покупок у грі та надання різноманітних опцій оплати поліпшує конверсію.
8. Аналітика поведінки гравців: вивчення даних про поведінку гравців дозволяє розробникам оптимізувати елементи гри та монетизації.
9. Персоналізація пропозицій: Пропозиції, які відповідають інтересам та історії покупок гравця, збільшують ймовірність покупки.
10. Використання метрик і KPI: відстеження ключових показників ефективності (KPI), таких як частота гри, середній дохід на користувача (ARPU), та тривалість сесії, дозволяє оцінити ефективність стратегій монетизації.

Для розуміння формування прибутку в f2p мобільних іграх потрібно мати уявлення про основні метрики прибутку. Це метрики **MAU** (Monthly Active Users), **DAU** (Daily Active Users), **ARPU** (Average Revenue Per User) та Customer

lifecycle. Ці метрики надають нам важливу інформацію про те, як користувачі взаємодіють з додатком, його виручку та загальну активність.

MAU або щомісячні активні користувачі — це метрика, яка вимірює кількість унікальних користувачів, які взаємодіяли з вашим мобільним додатком протягом конкретного місяця. Основна перевага MAU полягає в тому, що вона дозволяє вам відстежувати динаміку зростання аудиторії та визначати, чи вдається вам зберігати користувачів на тривалий термін.

DAU або щоденні активні користувачі вимірюють кількість унікальних користувачів, які взаємодіяли з додатком протягом одного дня. Ця метрика надає інформацію про щоденну активність користувачів.

ARPU або середній дохід на користувача визначається як загальний дохід, отриманий від користувачів, поділений на кількість активних користувачів. Ця метрика допомагає розробникам зрозуміти, наскільки прибутковим є їхні користувачі.

Customer lifecycle - це процес, який описує всі етапи взаємодії клієнта з компанією або брендом, починаючи від моменту, коли клієнт вперше дізнається про продукт чи послугу, і закінчуючи моментом, коли він стає лояльним клієнтом та активним прихильником бренду. Цей термін охоплює всі кроки, які клієнт проходить у своєму взаємодії з компанією, включаючи виявлення, оцінку, покупку, використання продукту чи послуги, а також розвиток відносин із брендом. Цикл клієнта є ключовим для розуміння та вдосконалення взаємодії з клієнтами, що сприяє збільшенню задоволення клієнтів та, відповідно, доходу компанії.

За рахунок пришвидшення швидкості відповіді клієнтам та вирішення проблемних кейсів ми підвищуємо показники метрики MAU (Monthly Active Users), DAU (Daily Active Users), ARPU (Average Revenue Per User) та Customer lifecycle. Задля виконання DNA показники на графіках умисно мають погрішність від реальних даних. Попередні показники за 4 квартал 2024 року

вказали, що MAU виріс на 2.5%, DAU на 2.1%, ARPU на 2.6%, Customer lifecycle 2.5%,

Щоб розрахувати повернення інвестицій (ROI) від проекту, який планується покращити Customer Lifetime Value (CLV) на 2.5%, нам потрібно знати суму поточного CLV. Повернення інвестицій розраховується за формулою:

$$\text{ROI} = \frac{\text{Фінансовий вигаш від інвестиції} - \text{Вартість інвестиції}}{\text{Вартість інвестиції}} \times 100\%$$
$$\text{ROI} = \frac{\text{Фінансовий вигаш від інвестиції} - \text{Вартість інвестиції}}{\text{Вартість інвестиції}} \times 100\%$$

Фінансовий вигаш від інвестиції в даному випадку буде дорівнювати приросту CLV, тобто 2.5% від поточного CLV.

Вхідні дані

expenses = 10780

net_income_initial = 37

net_income_final = 41

total_players = 4001830

Розрахунок зміни чистого доходу

change_in_income_per_player = net_income_final - net_income_initial

total_income = change_in_income_per_player * total_players

Розрахунок ROI

roi = (total_income - expenses) / expenses * 100

Очікуваний річний ROI для компанії, враховуючи вказані умови, становить приблизно 148,391%. Це означає, що компанія отримала значне збільшення доходу в порівнянні з первинними витратами

Висновки до 4 розділу

Free-to-play ігри залучають користувачів безкоштовною грою, але монетизуються через внутрішньоігрові покупки, рекламу та інші методи. Впровадження нової бізнес-моделі може вплинути на зростання компанії, особливо в контексті free-to-play ігор. Нова бізнес-модель, яка зосереджується на додатковому утриманні користувачів та оптимізації монетизації, може суттєво вплинути на показники, такі як MAU, DAU, ARPU та Customer Lifecycle. Розрахунок ROI вказує на економічну доцільність інтеграції нашого бізнес-процесу в інфраструктуру компанії.

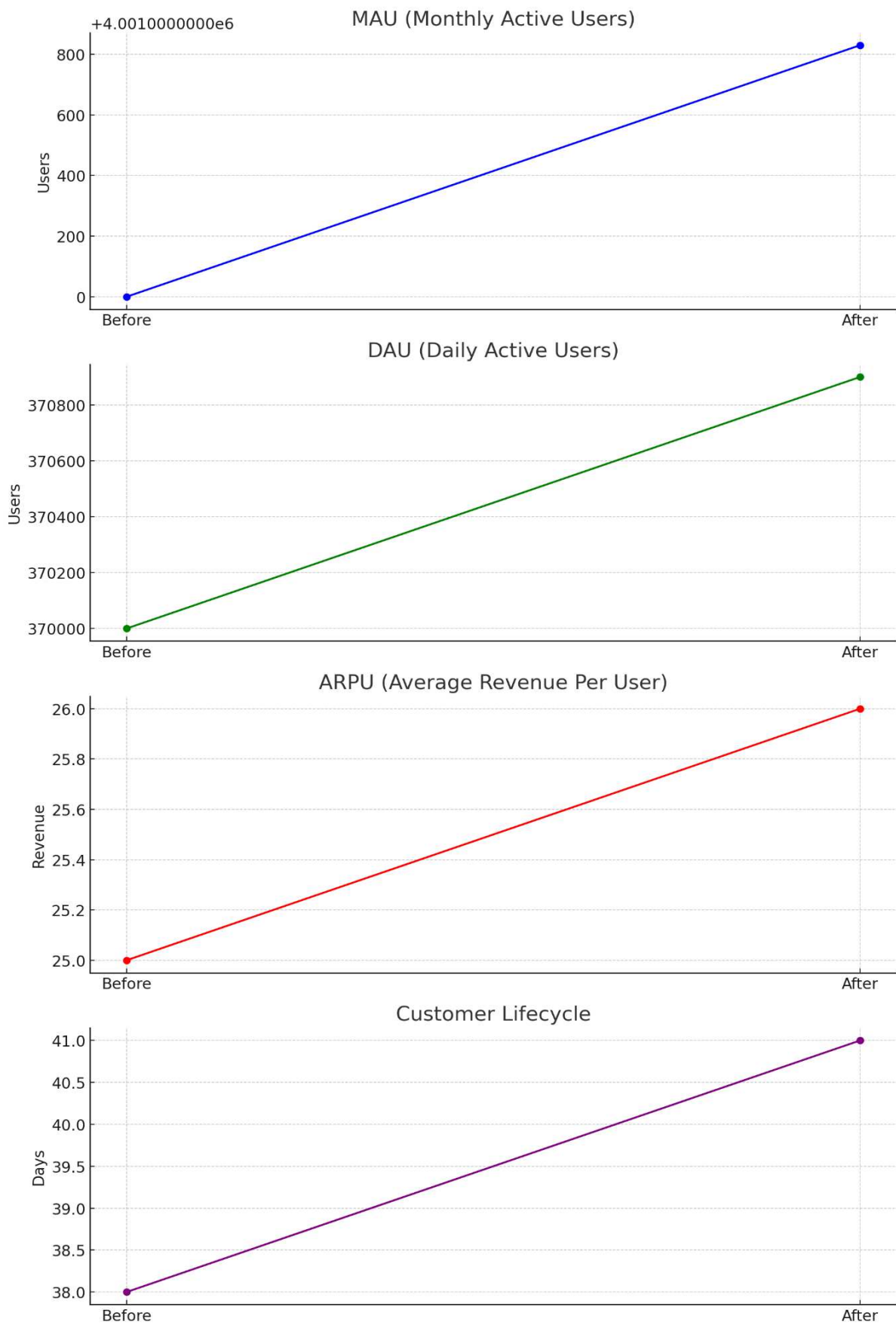


Рисунок 4.1- Якісні зміни метрик після впровадження процесу.

РОЗДІЛ 6 ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

5.1 Законодавча система охорони довкілля в Україні

Основна мета українського екологічного законодавства полягає в регламентації відносин у сфері захисту, використання та відновлення природних ресурсів. Воно забезпечує екологічну безпеку, спрямоване на запобігання та усунення шкідливого впливу промислової та іншої діяльності на довкілля. Законодавство має на меті збереження природних ресурсів, генетичного різноманіття флори та фауни, ландшафтів, інших природних комплексів, а також унікальних територій та об'єктів, які мають історико-культурне значення.

Регулювання відносин у цій галузі в Україні здійснюється на основі Закону про охорону довкілля, а також інших законодавчих актів у сферах землекористування, водопостачання, лісового господарства, користування надрами, охорони атмосферного повітря, а також охорони і використання флори та фауни.

Екологічне право включає систему правових норм, що керують суспільними відносинами у сфері охорони довкілля і раціонального використання природних ресурсів. До цієї системи входять специфічні підгалузі, такі як земельне, водне, лісове та гірниче законодавство.

Серед ключових законодавчих актів в цій галузі - закони "Про охорону довкілля", "Про охорону атмосферного повітря", "Про природно-заповідний фонд України", "Про тваринний світ", "Про карантин рослин" та інші. Додатково, деякі аспекти використання та охорони навколишнього середовища регулюються через земельний, водний, лісовий кодекси та інші закони, такі як "Про плату за землю" та "Про ветеринарну медицину". Особливу роль відіграє постанова Верховної Ради України "Про обмеження, тимчасову заборону чи припинення діяльності підприємств, установ, організацій у разі порушення законодавства про

охорону довкілля". Все це формує правову основу для охорони навколишнього природного середовища в Україні [14].

5.2 Відповідальність за порушення законодавства про охорону навколишнього середовища

Українське законодавство в галузі охорони навколишнього природного середовища встановлює строгі правила, порушення яких може призвести до різноманітних форм відповідальності. Ця відповідальність може варіюватися від дисциплінарної до кримінальної, в залежності від ступеня і серйозності порушення. Люди або організації, визнані винними у таких порушеннях, можуть бути покарані за несанкціоновані дії, які шкодять природному середовищу.

Порушення можуть включати, але не обмежуються, такими діями: порушення правил екологічної безпеки, подання недостовірної інформації під час екологічної експертизи, невиконання вимог державної екологічної експертизи, фінансування або впровадження нових технологій без екологічного схвалення, порушення екологічних стандартів під час проектування, будівництва, реконструкції чи експлуатації об'єктів, а також викиди і скиди забруднюючих речовин, що перевищують дозволені ліміти.

Важливим аспектом є також незаконне використання природних ресурсів, невиконання обов'язкових платежів за їх використання, а також ігнорування необхідних заходів для запобігання або усунення екологічних наслідків аварій чи інших негативних впливів на навколишнє середовище. Крім того, відповідальність настає за невиконання розпоряджень органів державного контролю, приниження честі чи заподіяння шкоди їхнім представникам.

Підприємства та окремі громадяни, чії дії призводять до екологічних порушень, зобов'язані відшкодовувати збитки, нанесені довкіллю, у порядку, визначеному законодавством. Це стосується також випадків, коли застосовуються інші форми відповідальності, такі як адміністративні чи кримінальні покарання. Незаконно добуті природні ресурси підлягають конфіскації, а вироблена з них продукція

вилучається без компенсації. Доходи від їх реалізації направляються до республіканських або місцевих фондів охорони навколишнього природного середовища.

Важливо відзначити, що посадові особи та спеціалісти, визнані винними в порушенні екологічних норм, можуть бути позбавлені премій або інших форм матеріальної винагороди. Це рішення приймається управлінськими органами на підставі подань державних органів охорони довкілля. Такий підхід забезпечує додатковий механізм стримування та запобігання порушенням екологічного законодавства.

РОЗДІЛ 5 ОХОРОНА ПРАЦІ

Охорона праці є комплексною системою, що включає правові, соціально-економічні, технічні та медичні заходи та методи, які мають на меті забезпечити безпеку, здоров'я та працездатність працівників під час виконання їхніх обов'язків. Згідно зі статтею 4 Закону "Про охорону праці", визначені основні принципи державної політики у цій сфері. Ці принципи слугують як керівні правила для інституту охорони праці, застосовуються в ситуаціях, де не існує специфічних правових норм, вказують на шляхи розвитку цього інституту та визначають зобов'язання держави, які повинні дотримуватись у майбутньому для забезпечення безпеки праці[19].

5.1 Організація охорони праці на підприємстві

Організація охорони праці на підприємстві Відповідальність за організацію охорони праці на підприємстві покладається на роботодавця. Роботодавець зобов'язаний створити умови праці в кожному структурному підрозділі, що відповідають нормативно-правовим актам, та забезпечити дотримання законодавчих вимог з охорони прав працівників.

У цьому контексті, відповідно до статті 13 Закону України "Про охорону праці" №2694-ХІІ від 14 жовтня 1992 року з подальшими змінами (далі – Закон №2694), роботодавець забезпечує функціонування системи управління охороною праці, до яких входять:

- Створення відповідних служб та призначення посадових осіб, відповідальних за вирішення конкретних питань охорони праці, затвердження інструкцій про їх обов'язки, права та відповідальність, а також контроль за їх дотриманням;
- Розробка та реалізація комплексних заходів для досягнення встановлених нормативів та підвищення рівня охорони праці за участю сторін колективного договору;

- Забезпечення виконання необхідних профілактичних заходів відповідно до змінюваних обставин, організація проведення аудиту охорони праці, лабораторних досліджень умов праці, оцінка технічного стану виробничого обладнання та устаткування, атестація робочих місць на відповідність нормативно-правовим актам з охорони праці;
- Розробка та затвердження положень, інструкцій та інших актів з охорони праці, що діють на підприємстві, встановлення правил виконання робіт та поведінки працівників на території підприємства, у виробничих приміщеннях, на будівельних майданчиках та робочих місцях;
- Контроль за дотриманням працівниками технологічних процесів, правил поводження з машинами, механізмами, устаткуванням та іншими засобами виробництва, використанням засобів колективного та індивідуального захисту, виконанням робіт відповідно до вимог з охорони праці;
- Організація пропаганди безпечних методів праці та співпраці з працівниками у галузі охорони праці;
- Вживання термінових заходів для допомоги потерпілим, залучення за необхідності професійних аварійно-рятувальних формувань у разі виникнення на підприємстві аварій та нещасних випадків.

Роботодавець безпосередньо відповідає за будь-які порушення умов охорони праці. Ключ до запобігання більшості аварій та нещасних випадків на виробництві, а також забезпечення збереження життя, здоров'я та здатності до праці працівників, полягає в строгому дотриманні ними вимог охорони праці. Згідно зі статтею 14 Закону №2694, працівник зобов'язаний:

- Піклуватися про власну безпеку та здоров'я, а також про безпеку та здоров'я інших осіб під час виконання будь-яких робіт або перебування на території підприємства;
- Володіти знаннями та дотримуватися вимог законодавчих актів з охорони праці, правил поводження з машинами, механізмами, обладнанням та

іншими засобами виробництва, користуватися засобами колективного та індивідуального захисту;

- Регулярно проходити медичні огляди відповідно до законодавства.

За порушення законодавства та інших нормативних актів щодо охорони праці, перешкоджання діяльності представників державного нагляду та профспілкових організацій, винні особи несуть дисциплінарну, адміністративну, матеріальну чи кримінальну відповідальність згідно із законом.

5.2 Аналіз мікрокліматичних умов

Мікроклімат впливає на самопочуття. Мікроклімат в приміщеннях – це внутрішній клімат, який формується за допомогою комбінації температури, вологості, швидкості руху повітря та впливає на організм людини.

Параметри мікроклімату визначаються відповідно до сезону та виду виконуваної роботи. У таблиці 5.1 представлені оптимальні параметри мікроклімату для роботи за комп'ютером.

Таблиця 5.1 – Параметри мікроклімату в кімнаті

Таблиця 5.1 – Параметри мікроклімату в кімнаті

Період року	Параметр	Оптимальний	Фактичний
Теплий	Температура	23-25 °C	24-26 °C
	Вологість	40 – 60%	40%
Холодний	Температура	22-24 °C	21-23 °C
	Вологість	40 – 60%	50%

Ці показники задовольняють вимогам для роботи за комп'ютером і є задовільними для здоров'я людини.

5.3 Аналіз освітлення

Для приміщень, в яких місця обладнані комп'ютером, важливо організувати правильні умови освітлення. Основна структура підприємства складає це робота за комп'ютерами. В кожного відділу буде своя кімната або вони можуть кооперуватися один з одним.

Для розрахунку візьмемо кімнату одного відділу. Освітлення приміщення здійснюється за допомогою штучного та природного освітлення. Згідно плану приміщення, кімната, яку ми аналізуємо має 3 вікна з розмірами: ширина – 2.2, висота – 2.5, відповідно площа кожного вікна.

5.5 квадратних метри. Природне освітлення здійснюється саме цими вікнами і на таку кімнату, яку зображено нижче на рисунку, його вистачить. Вікно має регульований пристрій для відкривання та обладнане жалюзіями з можливістю захисту від прямого попадання сонячних променів. Стіни пофарбовані в зелений колір і стеля білого кольору. В досліджуваному приміщенні мається 2 енергозберігаючих ламп які знаходиться в різних частинах кімнати. Їх технічні характеристики:

- Максимальна потужність – до 10 Вт.
- Потужність – 4.6 Вт.
- Напруга – від мережі 220 В.
- Світловий потік (Яскравість) – 355лм.
- Кут розсіювання світла(для LED лампи) - 36°.
- Кольорова температура: 6500К (холодний білий).

Рисунок 5.1 – Схема кімнати

Довжина кімнати складає 12 метрів, її ширина – 8 метрів, а висота стелі 2.8 метрів.

При таких параметрах кімнати яку досліджуємо можна розрахувати освітленість, якщо порахувати наступні параметри:

- S: Площа – $12 \cdot 8 = 96\text{м}.$
- K: Коефіцієнт для стелі в 2.8м – 1.2.
- E – освітленість в ЛК – $96 \cdot 1.2 = 115.2 \text{ ЛК}.$

З такими параметрами можна розрахувати скільки ламп потрібно для такої кімнати:

$$\text{Кількість ламп} = 115.2 / 355 = 0.3$$

Виходячи з такого розрахунку, то тих двох ламп які встановлені в кімнаті вистачить навіть з запасом.

5.4 Шум

Шум в приміщенні може негативно впливати на користувача. Основними параметрами звуку, що нормуються є інтенсивність, звуковий тиск і частота коливань.

Можливий список джерел шуму у кімнаті:

- Система охолодження комп'ютером.
- Принтер під час роботи (якщо він є).
- Шум за вікном.
- Кондиціонер(якщо він є).

Хоча в сучасних комп'ютерах усі комплектуючі мають дуже низький рівень шуму і цей фактор можна не дуже враховувати. Згідно з технічною документацією шум, зумовлений кулером в блоці живлення складає 25 дБ, кулером процесора - 30 дБ, загальний, - 34 дБ. Враховуючи незначний рівень шуму від персонального комп'ютера і незначний рівень фонового шуму від іншого устаткування, можна стверджувати, що сумарний рівень шумового забруднення приміщення не перевищує максимально допустимий рівень коригованої звукової потужності і складає не більше 50 дБ.

При роботі з персональним комп'ютером в робочому приміщенні значення характеристик вібрації на робочих місцях не повинна перевищувати допустимих значень.

5.5 Випромінювання

У приміщенні відсутні джерела інфрачервоного, ультрафіолетового і електромагнітного випромінювання. Сучасні монітори вироблені на основі рідкокристалічної матриці, що не має сильного випромінювання і вони сертифіковані в Україні.

Під час роботи за комп'ютером з монітором в 27 дюймів, для забезпечення очей відстань від них до монітора повинна бути 1200 мм. Робота за комп'ютером

повинна бути не більше 4-х годин на день і потрібно робити перерви на 10-15 хвилин, кожні 2 години. Також періодично можна виконувати вправи для очей.

Рекомендований комплекс вправ для очей:

Міцно примружитися на 3-5 секунд. Відкрити очі і не блимати 3-5 секунд.

Повторити 7-8 разів.

Закрити очі і масажувати їх круговими рухами пальців, не натискаючи на очні яблука, протягом 1-2 хвилин. Поставити вказівні пальці під кожною бровою і злегка потягнути ними шкіру вгору, щоб піднялися очі. Потримати протягом 1-2 секунд, потім відпустити. Повторити 5 разів.

ВИСНОВКИ

Проаналізовано основні види та типи багатовимірних даних, на підставі чого для досліджень обрано методикау Process Mining.

Запропоновано новий опис моделі з подальшим аналізом та систематизацією інформації, що розвиває можливості автоматизованої обробки процесів на підприємстві.

На базі обраної моделі Process Mining досліджено цикл транзитної обробки багатовимірних даних з залученням ІІІ. Використання ІІІ виявилось виправданим, якість обробки даних значно підвищилась, що можна було побачити в розділі економічного обґрунтування.

Економічно обґрунтовано в перспективі впровадження розробленої моделі на підприємстві та масштабування її в інфраструктурі .

Розглянуто заходи щодо охорони праці та навколишнього середовища.

Наукове значення роботи. Одержані результати спрямовані на вирішення науково-прикладного завдання з дослідження циклу транзитної обробки багатовимірних даних.

Наукова новизна роботи полягає в удосконаленні моделі транзиту даних в системі ПМ, що масштабує в перспективі спектр розв'язань практичних бізнес задач.

Практична цінність роботи полягає в можливості інтеграції системи транзитної обробки багатовимірних даних в різні бізнес-процеси.

Результати роботи стали публікаціями у наступних наукових працях:

1.Перспективи та обґрунтування використання штучного інтелекту в process mining /В. В. Вершинін //Сучасні аспекти модернізації науки: стан, проблеми, тенденції розвитку: матеріали XXXIX Міжнародної науково-практичної конференції

ВИКОРИСТАНІ ДЖЕРЕЛА ІНФОРМАЦІЇ

1. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management, 2nd edn. Springer, Berlin (2018).
<https://doi.org/10.1007/978-3-662-56509-4>
2. Process Mining: A Guide for Practitioners Fredrik Milani^{1(B)}, Katsiaryna Lashkevich¹, Fabrizio Maria Maggi², and Chiara Di Francescomarino³¹University of Tartu, Tartu, Estonia {milani,katsiaryna.lashkevich}@ut.ee²Free University of Bozen-Bolzano, Bolzano Italymaggi@inf.unibz.it ³FBK-IRST, Trento, Italydfmchiara@fbk.eu
3. Jeston, John; Nelis, Johan (21 January 2014). Business Process Management. Routledge. ISBN 9781136172984.
4. Van der Aalst, W M P and Weijters, A J M M and Maruster, L (2004). "Workflow Mining: Discovering process models from event logs", IEEE Transactions on Knowledge and Data Engineering, vol 16
5. Wil Van der Aalst, Boudewijn F. van Dongen: Discovering Petri Nets from Event Logs, January 2013 DOI:10.1007/978-3-642-38143-0_10
6. Process Mining with the Heuristics Miner-algorithm January 2006 A. J. M. M. Weijters, Wil Van der Aalst
7. Leemans, S. J. J. (2017-05-09). Robust process mining with guarantees - SIKS Dissertation Series No. 2017-12 (PDF). TU/e - Eindhoven University of Technology. ISBN 978-90-386-4257-4.
8. Favrin, Bean (2014-09-02). "esyN: Network Building, Sharing and Publishing". PLOS ONE. 9 (9): e106035.
9. OMG Unified Modeling Language Superstructure Specification, version 2.1.1. Document formal/2007-02-05, Object Management Group, February 2007

10. John Hopcroft and Jeffrey Ullman (1979) Introduction to Automata Theory, Languages, and Computation, Addison-Wesley Publishing Company, Reading Mass
11. Bart-Jan Hommes (2004). The Evaluation of Business Process Modeling Techniques
12. HAVAZÍK, Ondřej; PAVLÍČKOVÁ, Petra. How to design Agile game for education purposes in JIRA. In: *2020 7th International Conference on Control, Decision and Information Technologies (CoDIT)*. IEEE, 2020. p. 331-334.
13. L'Esteve, R. (2022). Snowflake. In: The Azure Data Lakehouse Toolkit. Apress, Berkeley, CA., pp. 45–82
14. Graziano, M. 2017. The attention schema theory: A foundation for engineering artificial consciousness. In: *Frontiers in Robotics and AI* 4, art. 60, pp. 1-9.
15. Палагін О.В., Кургаєв О.П., Шевченко А.І. Ноосферна парадигма розвитку науки та штучний інтелект. *Кібернетика і системний аналіз*. 2017. Т. 53. № 4. С. 12-21.
16. Шевченко А.І. До питання щодо створення штучного інтелекту. *Штучний інтелект*, №1, 2016. С. 7–15.
17. С.К. Рамазанов, А.І. Шевченко, Є.О. Купцова. Штучний інтелект і проблеми інтелектуалізації: стратегія розвитку, структура, методологія, принципи і проблеми. *Штучний інтелект*. 2020. № 4 (90). С. 14-23.
18. Перспективи та обґрунтування використання штучного інтелекту в process mining /В. В. Вершинін //Сучасні аспекти модернізації науки: стан, проблеми, тенденції розвитку: матеріали XXXIX Міжнародної науково-практичної конференції / за ред. І.В. Жукової, Є.О. Романенка. м. Гетеборг (Швеція): ГО «ВАДНД», 07 грудня 2023 р., С. 455-459
19. Закон України “Про охорону праці”, від 14.10.1999 р № 2694-XII.

ДОДАТОК 1

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.feature_extraction.text import TfidfVectorizer
import snowflake.connector

# Налаштування з'єднання до Snowflake
conn = snowflake.connector.connect(
    user='USERNAME',
    password='PASSWORD',
    account='ACCOUNT',
    warehouse='WAREHOUSE',
    database='DATABASE',
    schema='SCHEMA'
)

# Завантаження даних з Snowflake
query = "SELECT * FROM TABLE"
df = pd.read_sql(query, conn)
conn.close()

# Перетворення текстових даних в числові вектори за допомогою TF-IDF
tfidf = TfidfVectorizer(max_features=100)
X_tfidf = tfidf.fit_transform(df['header']).toarray()

# Додавання інших числових/категорійних ознак, якщо потрібно
# Припустимо, assignee вже перетворено в числовий формат
X = pd.concat([pd.DataFrame(X_tfidf), df[['assignee_numeric']]], axis=1)
```

```

y = df['added']

# Розділення даних на тренувальні та тестувальні набори
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)

# Використання випадкового лісу як класифікатора
clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(X_train, y_train)

# Прогнозування та оцінка моделі
y_pred = clf.predict(X_test)
print("Точність моделі:", accuracy_score(y_test, y_pred))

# Використання моделі для генерації рекомендації для нового запису
conn = snowflake.connector.connect(
user='USERNAME',
password='PASSWORD',
account='ACCOUNT',
warehouse='WAREHOUSE',
database='DATABASE',
schema='SCHEMA1'
)
query = "SELECT * FROM TABLE ORDER BY timestamp DESC LIMIT 1;"
new_record = pd.read_sql(query, conn)
conn.close()
new_record_transformed = tfidf.transform([new_record['header']]).toarray()
new_record_features = pd.concat([pd.DataFrame(new_record_transformed),
pd.Series([new_record['assignee_numeric']]), axis=1)
recommended_added = clf.predict(new_record_features)[0]

```

```
print("Рекомендоване значення для 'added':", recommended_added)
```


ДОДАТОК 2

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.feature_extraction.text import TfidfVectorizer
import snowflake.connector

# Налаштування з'єднання до Snowflake
conn = snowflake.connector.connect(
    user='USERNAME',
    password='PASSWORD',
    account='ACCOUNT',
    warehouse='WAREHOUSE',
    database='DATABASE',
    schema='SCHEMA'
)

# Завантаження даних з Snowflake
query = "SELECT * FROM TABLE"
df = pd.read_sql(query, conn)
conn.close()

# Перетворення текстових даних в числові вектори за допомогою TF-IDF
tfidf = TfidfVectorizer(max_features=100)
X_tfidf = tfidf.fit_transform(df['header']).toarray()

# Додавання інших числових/категорійних ознак, якщо потрібно
# Припустимо, assignee вже перетворено в числовий формат
X = pd.concat([pd.DataFrame(X_tfidf), df[['assignee_numeric']]], axis=1)
```

```

y = df['added']

# Розділення даних на тренувальні та тестувальні набори
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)

# Використання випадкового лісу як класифікатора
clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(X_train, y_train)

# Прогнозування та оцінка моделі
y_pred = clf.predict(X_test)
print("Точність моделі:", accuracy_score(y_test, y_pred))

# Використання моделі для генерації рекомендації для нового запису
conn = snowflake.connector.connect(
    user='USERNAME',
    password='PASSWORD',
    account='ACCOUNT',
    warehouse='WAREHOUSE',
    database='DATABASE',
    schema='SCHEMA1'
)
query = "SELECT * FROM TABLE ORDER BY timestamp DESC LIMIT 1;"
new_record = pd.read_sql(query, conn)
conn.close()
new_record_transformed = tfidf.transform([new_record['header']]).toarray()
new_record_features = pd.concat([pd.DataFrame(new_record_transformed),
                                pd.Series([new_record['assignee_numeric']]), axis=1)
recommended_added = clf.predict(new_record_features)[0]

```

```
print("Рекомендоване значення для 'added':", recommended_added)
```

ДОДОТОК 3

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.feature_extraction.text import TfidfVectorizer
import snowflake.connector

# Налаштування з'єднання до Snowflake
conn = snowflake.connector.connect(
    user='USERNAME',
    password='PASSWORD',
    account='ACCOUNT',
    warehouse='WAREHOUSE',
    database='DATABASE',
    schema='SCHEMA'
)

# Завантаження даних з Snowflake
query = "SELECT * FROM TABLE"
df = pd.read_sql(query, conn)
conn.close()

# Перетворення текстових даних в числові вектори за допомогою TF-IDF
tfidf = TfidfVectorizer(max_features=100)
X_tfidf = tfidf.fit_transform(df['header']).toarray()

# Додавання інших числових/категорійних ознак, якщо потрібно
# Припустимо, assignee вже перетворено в числовий формат
X = pd.concat([pd.DataFrame(X_tfidf), df[['assignee_numeric']]], axis=1)
```

```

y = df['added']

# Розділення даних на тренувальні та тестувальні набори
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)

# Використання випадкового лісу як класифікатора
clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(X_train, y_train)

# Прогнозування та оцінка моделі
y_pred = clf.predict(X_test)
print("Точність моделі:", accuracy_score(y_test, y_pred))

# Використання моделі для генерації рекомендації для нового запису
conn = snowflake.connector.connect(
user='USERNAME',
password='PASSWORD',
account='ACCOUNT',
warehouse='WAREHOUSE',
database='DATABASE',
schema='SCHEMA1'
)
query = "SELECT * FROM TABLE ORDER BY timestamp DESC LIMIT 1;"
new_record = pd.read_sql(query, conn)
conn.close()
new_record_transformed = tfidf.transform([new_record['header']]).toarray()
new_record_features = pd.concat([pd.DataFrame(new_record_transformed),
pd.Series([new_record['assignee_numeric']]), axis=1)
recommended_added = clf.predict(new_record_features)[0]

```

```
print("Рекомендоване значення для 'added':", recommended_added)
```