Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерних наук_____
(повна назва)
Кафедра _____Інформаційних управляючих систем_____
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____другий (магістерський)_____
Дослідження моделей життєвого циклу артефактів в IT проектах процесного управління
(тема)

Виконав:
студент _2_ курсу, групи _УПГІТм-21-1____
_____Дмитро ШУБА_____
(власне ім'я, прізвище)

Спеціальність ____122 Комп'ютерні науки _
(код і повна назва спеціальності)
Тип програми ____освітньо-наукова_____
(освітньо-професійна або освітньо-наукова)
Освітня програма _Управління проектами в галузі інформаційних технологій_____
(повна назва освітньої програми)
Керівник професор .каф. ІУС Сергій ЧАЛИЙ
(посада, власне ім'я, прізвище)

Допускається до захисту

Зав. кафедри _____ ___Костянтин ПЕТРОВ__
(підпис) (власне ім'я, прізвище)

2023 р.

Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерних наук_____

Кафедра _____Інформаційних управляючих систем_____

Рівень вищої освіти _____другий (магістерський)_____

Спеціальність _____122 Комп'ютерні науки_____
(код і повна назва)

Тип програми _____освітньо-наукова_____
(освітньо-професійна або освітньо-наукова)

Освітня програма_Управління проектами в галузі інформаційних технологій
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

«_03_»__квітня__ 20 23_ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові_____Шубі Дмитру Івновичу_____
(прізвище, ім'я, по батькові)

1. Тема роботи _____Дослідження моделей життєвого циклу артефактів в IT проектах_____ процесного управління_____
затверджена наказом університету від _03_ __квітня__2023_ р. № _319Ст_
2. Термін подання студентом роботи до екзаменаційної комісії 18.05.2023 р.
3. Вихідні дані до роботи методична та технічна література, матеріали конференцій, дані інтернет-мережі з тематики кваліфікаційної роботи, програма Python._____

4. Перелік питань, що потрібно опрацювати в роботі Провести аналіз матеріалів та загальний огляд методів створення моделей життєвого циклу артефактів в IT проектах процесного управління. Провести збір та аналіз даних про самі IT проекти, про виявлення сутностей та артефактів, їх характеристики та зв'язки. Дослідити розроблення та застосування моделей. Провести аналіз та узагальнення результатів дослідження._____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання для кваліфікаційної роботи	3.04.2023	Виконано
2	Аналіз предметної області для кваліфікаційної роботи	4.04.2023 – 8.04.2023	Виконано
3	Формування плану роботи на кваліфікаційною роботою	9.04 – 10.04.2023	Виконано
4	Робота над теоретичними відомостями для кваліфікаційної роботи	11.04.2023 – 18.04.2023	Виконано
5	Аналіз моделей життєвого циклу артефактів ІТ-проектів процесного управління	19.04.2023 – 23.04.2023	Виконано
6	Аналіз методів для рекомендаційних систем на практиці	23.04.2023 – 25.04.2023	Виконано
7	Оформлення звіту та висновків	25.04.2023 – 30.04.2023	Виконано
8	Захист кваліфікаційної роботи	17.05.2023	Виконано

Дата видачі завдання 3 квітня 2023 р.

Студент _____
(підпис)

Керівник роботи _____ професор каф. ІУС Сергій ЧАЛИЙ
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи магістра: 66 стор., 14 рис., 7 табл., 5 формул, 20 джерел.

АРТЕФАКТИ, ВИЯВЛЕННЯ АРТЕФАКТІВ МОДЕЛІ ЖИТТЄВИХ ЦИКЛІВ, ЖИТТЄВИЙ ЦИКЛ, ПРОЕКТ ПРОЦЕСНОГО УПРАВЛІННЯ, ПРОЦЕС МАЙНІНГ.

Об’єкт дослідження – моделі життєвого циклу артефактів в ІТ проєетах процесного управління.

Мета дослідження – дослідити, порівняти та проаналізувати моделі життєвого циклу артефактів в ІТ проектах процесного управління.

Метод рішення – за дослідити існуючі методи побудови меделей життєвого циклу артефактів, порівняти різні методи їх організації, визначити структури та зв’язки артефактів, їх використання, типи та взаємодії.

Дослідження, що проведені в роботі, базуються на використанні методів процес майнінгу проектів процесного управління за допомогою ILP Miner, Tree Miner; створення мереж Петрі (Petri nets) та Guard-Stage-Milestone моделей (GSM);

На основі результатів виконаних досліджень підведено висновок щодо доречності визначення та використання моделей життєвих циклів артефактів, їх практичне використання та методи формування в проектах процесного управління, що вже мають журнали подій, які описують перебіг подій, їх характеристики та властивості, а також задіяні артефакти чи інші елементи системи

ABSTRACT

As a part of mastership diploma, this thesis contains: 66 p. 14 fig. 7 tabl., 5 formul., 20 sources.

ARTIFACTS, ARTIFACT DISCOVERY PROCESS, LIFECYCLE, LIFECYCLE MODELS, PROCESS MANAGAMENT PROJECT, PROCESS MINING.

The object of the study are lifecycle models of artifacts for a process management IT project.

The purpose of the study is to research, compare and analyse different lifecycle models for process management IT projects.

The solution method is to explore existing, research and investigate the methods of building lifecycle models of artifacts, compare different approches of their processing, define the structure and the relations between artifacts, their usage and interactions.

The research is based on the application of such methods as process mining of process managament projects using ILP Miner, Miner Tree; generation of Petri nets and Guard-Stage-Milestone (GSM) models.

As a result of the research, the work concludes if the life cycle models of artifacts are relavant for discovery and application, their practical usage and approaches of forming in projects of process managament using existing raw logs of events, which describe how procedures, processes and other activities undergo, their characteristics and properties, and how artifacts are involved.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ІАС – інформаційно-аналітична система;

ІУС – Інформаційна управляюча система;

ХНУРЕ – Харківський національний університет радіоелектроніки;

ІТ – інформаційні технології – information technologies;

РС – рекомендаційна система;

ЦІСТ – центр інформаційних систем та технологій;

ІС – інформаційна система;

МПД – модель потоку даних;

МС – модель сценаріїв;

МП – модель послідовності;

PMP – Process Mining Process

ПУ – процесне управління;

ACL – Artifact-Centric Logs.

ВСТУП

Будь-яка діяльність у сучасному світі завжди супроводжується відповідними інформаційними системами та технологіями, які допомагають у прийнятті рішень, аналізі даних та обробці інформації. В результаті цього невід'ємною частиною таких систем стала здатність ефективно зберігати, створювати, автоматично генерувати та обробляти великі обсяги інформації, знань, даних.

Дана робота в першу чергу включає в себе вивчення, аналіз та дослідження таких пунктів, як:

– моделей артефактів;

– методів виявлення артефактів в IT-проектах;

– моделей життєвого циклу артефактів.

Моделі життєвих циклів артефактів є важливим інструментом для керування та організації процесу розробки артефактів у проектах. Вони надають структуру, методологію та способи описування різних станів, переходів та етапів, через які проходять артефакти від початку до завершення проекту.

Також моделі життєвих циклів допомагають структурувати процес розробки артефактів, визначаючи послідовність етапів та кроків. Вони надають команді чіткий план дій та вказівки, які допомагають уникнути хаосу та зберегти час та ресурси.

Поміж цього, такі моделі дозволяють відстежувати прогрес розробки артефактів і виявляти можливі затримки або проблеми; надають спільну мову та зрозумілість між всіма учасниками проекту та дозволяють краще розподілити ресурси, визначити потреби у персоналі, обладнанні та інших ресурсах на різних етапах розробки артефактів.

1 АНАЛІЗ МОДЕЛЕЙ АРТЕФАКТІВ У ПРОЦЕСНОМУ УПРАВЛІННІ

1.1 Аналіз особливостей процесного управління

ІТ проекти можуть бути різноманітними за своїми характеристиками, особливостями та учасниками. В наслідок цього вимоги до них є теж унікальними і тому для кожного проекту можуть використовуватися різні методи управління. Для цього частіше за все виділяють та використовують всього 5 основних підходів: функціональний, системний, ситуаційний, проектний та процесний [1].

Для функціонального методу основний фокус в проекті стоїть на функціях та ролях. Відповідно, основна задача управління стоїть в тому, щоб надати правильну структуру та координацію взаємодії та дій учасників проекту. В свою чергу системний підхід визначає та проводить аналіз для всіх елементів та їх взаємодій системи. В такому форматі детально розглядається вся система цілком.

Для ситуаційного підходу головним чинником є визначення найбільш важливих, критичних та ефективних дій в чинних обставинах. Тому дуже важливою частиною такого управління є розпізнавання унікальних умов і процедурне виконання відповідних дій у відповідь.

В проектному управлінні головним фокусом визначається відповідно планування та управління проектом, а точніше його ресурсами, тощо. Це здійснюється з ціллю доведення виконання поставлених задач до фінального етапу в межах заданих спочатку часових термінів та бюджету.

Для процесного управління визначає головною ціллю організацію управління довкола бізнес-процесів підприємства та проектів. Відповідно, головною особливістю ІТ проектів процесного управління є їх фокус на бізнес-процесах, що насамперед означає, що такі проекти не фокусуються на певних

продуктах чи технологіях, а натомість виділяють основними елементами саме процеси та їх взаємодії між собою, які відбуваються при виконанні роботи в компанії.

Отже, можна сказати, що процесний підхід розглядає управління проектом чи підприємством через певну серію пов’язаних між собою функцій чи підфунцій управління і саме вони є складовими процесу управління.

Впровадження процесного бачення в управлінні проекту може значно покращити його ефективність та прозорість оскільки дає можливість розглянути операційну діяльність з точки зору саме перебігу визначених активностей та взаємодій, які необхідні для реалізації проекту. Саме це виступає основним фокусом, а не організаційна структура чи учасники проекту, хоча всі інші елементи не ігноруються і теж можуть враховуватись при визначенні процесів і їх перебігу.

Переваги процесного управління для проекту можуть також включати такі покращення, як:

– прозоре делегування повноважень та відповідальності виконавцям, що сприяє покращенню якості продукції та послуг;

– кожен учасник залучений до інноваційного процесу та несе відповідальність за якість кінцевого продукту;

– визначена відповідність до потреб споживачів є головним критерієм якості та цінності продукції;

– скорочення та спрощення кількості рівнів прийняття рішень. В свою чергу це забезпечує підвищення оперативності та адаптивності проекту;

– виявлення та усунення невиправданих витрат ресурсів проекту, які не впливають на створення цінності продукту чи послуг;

– покращення комунікації шляхом здійснення обміну інформацією всередині процесу, що уникає її спотворення та значно скорочує терміни передачі інформації між суб’єктами інноваційного процесу;

– створення умов для автоматизації технологій виконання процедур, активностей та процесів.

Окрім того, можна зазначити, що процесне управління також створює умови для покращення та формування, насамперед, горизонтальних (пласких) зв’язків в організації чи команді. Завдяки цьому різні команди та учасники можуть повністю самостійно та автономно корегувати та координувати відповідну роботу згідно з визначеного процесу, а також оброблювати проблеми, що можуть виникати під час роботи, без необхідності в допомозі від керівництва чи зовнішніх діячів.

Отже, можна визначити, що процесне управління базується на певних принципах, серед яких можна виділити:

– організація визначається як мережа процесів, де кожен процес є будь-якою діяльністю, що включає виконання робіт, та пов'язаний з іншими процесами в організації;

– кожен процес має мету та результати, які є необхідними для свого внутрішнього або зовнішнього споживача;

– діяльність, що пов'язана з процесом, документується, що стандартизує процес та надає базу для його зміни та подальшого вдосконалення;

– кожен процес має чітко визначені межі, які визначають початок та кінець процесу, та показники, які характеризують його результати;

– чітко визначений працівник відповідає за виконання процесу та його результати, незважаючи на те, що до виконання процесу можуть бути задіяні різні фахівці та співробітники.

Таблиця 1.1 – Порівняння функціонального та процесного управління

	Функціональний	Процесний
Сутність	Проект розглядається як механізм, що володіє набором функцій, що розподіляються серед учасниками.	Проект представляється набором процесів.
Об’єкт	Організаційна структура	Бізнес-процес
Недоліки	Ускладнений обмін інформацією між командами, підрозділами, учасниками, що призводить до додаткових витрат, тривалих термінів, тощо; Вимагає багато часу для реалізації управлінського впливу на виробничий процес, має авторитарний стиль управління.	Взаємозалежність осіб, що рішення; Впровадження процесного управління в організації може бути складним і вимагати значних зусиль та ресурсів
Переваги	Безконфліктний процес прийняття рішень, що виключає взаємозалежність осіб та рішень.	Орієнтація на кінцевий результат; Гнучкість реагування на зовнішні і внутрішні зміни; Контроль якості процесу, а не кінцевого продукту.

Проекти процесного управління мають багато переваг, серед яких можна зазначити стандартизацію та однорідність, керованість та контроль, адаптацію до змін, а також загальне покращення ефективності та якості процесів і їх результатів.

1.2 Визначення властивостей артефактів як об’єктів процесного управління

В процесному управлінні артефакти (наприклад, документи, моделі процесів, процедури тощо) визначають для того, щоб забезпечити стандартизацію процесів, їх ефективну управління та вдосконалення. Артефакти допомагають узгоджувати та координувати дії різних підрозділів підприємства, які задіяні в процесах. Крім того, артефакти забезпечують можливість аналізу та оцінки ефективності процесів, їх вдосконалення та оптимізації.

Наприклад, моделі процесів дозволяють візуалізувати послідовність робіт, ресурси, які задіяні в процесі, а також забезпечують можливість виявлення слабких місць та вдосконалення процесів. Документи та процедури допомагають забезпечити однаковий підхід до виконання робіт та зменшити ризик помилок. Всі ці артефакти є важливим інструментом для забезпечення ефективного процесного управління та досягнення поставлених цілей підприємства.

Тож артефакти є ключовими компонентами операційної діяльності проекту чи підприємства і їх життєвий цикл визначає загальні межі та формат бізнес процесів.

Артефакти, як об'єкти процесного управління, можуть мати різні властивості і призначення, але можна виділити що вони є результатом процесу (або можуть бути використані для управління процесом); можуть бути документами, електронними файлами, базами даних, таблицями, звітами тощо; є важливими з точки зору управління якістю та ефективністю процесу; можуть бути використані для відстеження прогресу процесу, контролю за якістю та оцінки результатів; можуть бути використані для визначення проблем та можливостей для покращення процесу; можуть бути зв'язані між собою та іншими елементами процесу, що дозволяє зрозуміти взаємозв'язки між різними

етапами та складовими процесу; можуть бути документовані та стандартизовані, що дозволяє забезпечити їхню консистентність та використання в майбутньому.

Частіше за все до основних артефактів проекту виділяють елементі, які зазначені на рисунку 1.1.

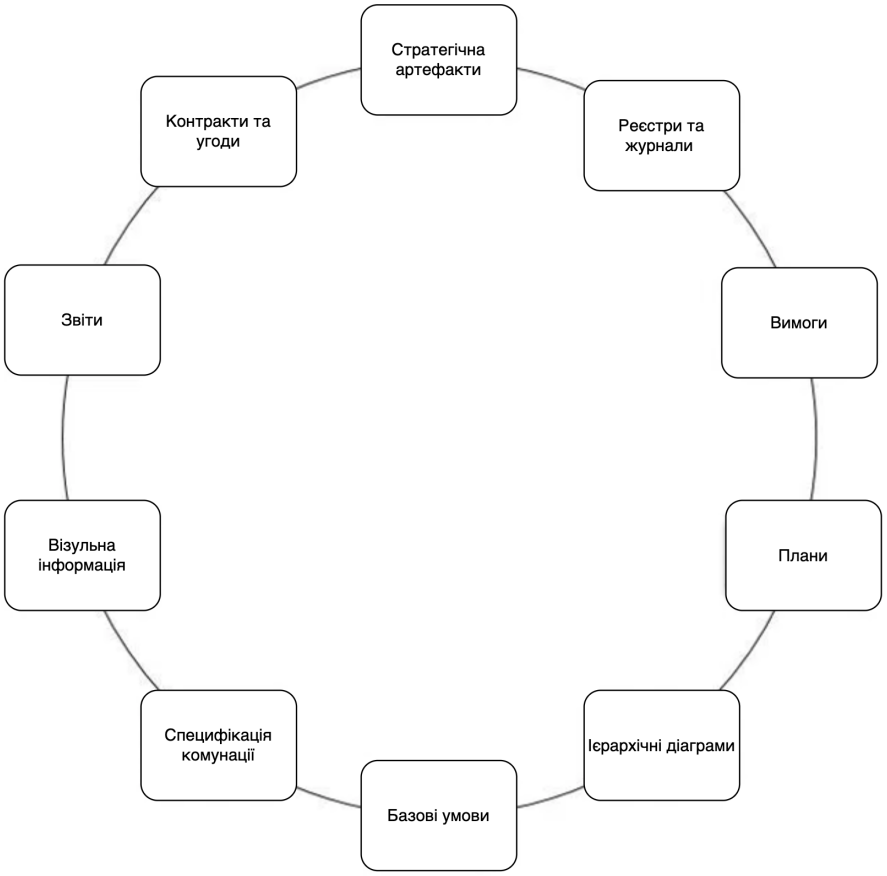

Рисунок 1.1 – Приклад артефактів проекту процесного управління

Відповідно цими артефактами є:

– реєстри та журнали. До цієї категорії входять реєстри зацікавлених сторін, списки завдань, журнал проблем та реєстр ризиків, тощо. За допомогою цих артефактів керівник проекту може оцінити, на якому саме етапі знаходиться

проект та визначити завдання, які потрібно виконати. Зазвичай протягом життєвого циклу проекту журнали та реєстри зазнають численних змін;

– вимоги. Тестування прототипів, розробка одиниць та готових продуктів спрощується завдяки чітко визначиними вимогам до релізу. Ці вимоги часто складаються з оцінок(естійметів), умов, угоди між зацікавленими сторонами, тощо. Критерії поставлених оцінок допомагають визначити чи успішно виконаний проект та його цілі, що були сформовані на початку;

– плани є важливою категорією артефактів в IT проекті. Керівники використовують їх для здійснення роботи, контролю за прогресом та успішного завершення проекту. Цей документ може включати в собі також один або декілька артефактів, які поєднують текстові та візуальні компоненти. План для IT проекту може включати план управління ризиками, план управління комунікаціями, план управління ресурсами, план управління обсягом робіт, план управління закупівлями, тощо. Керівник проекту та зацікавлені сторони, маючи чинні плани, можуть легко будуть орієнтуватися в проекті. Поміж того, часто письмові документи та діаграми теж вважаються типом планів;

– ієрархічні діаграми. Ці артефакти визначають зв'язок між різними компонентами проекту, такими як структура розбиття ризиків, структура розбиття організації, продукт та структура розбиття робіт і інших документів, що візуально описуються за допомогою ієрархічних діаграм;

– специфікація комунікації. Комунікаційну стратегію розробляє команда проекту під час роботи над проектом для визначення найбільш ефективних способів передачі інформації. Наприклад, вони можуть розробляти стратегії ефективної комунікації щодо конференцій, питань, переглядів документів, доступу до результатів роботи та поточного стану проекту;

– звіти. Управління проектами передбачає роботу з великою кількістю звітів. Цей тип артефактів може включати опис стану проекту, ризики, якість та офіційну документацію для зацікавлених сторін;

– контракти та угоди. У проекті може бути декілька контрактів та угод щодо закупівлі. Наприклад, це можуть бути замовлення на придбання ресурсів, контракти на придбання послуг чи підпідрядкування робіт тощо. При наймі персоналу або експертів вам можуть знадобитися контракти.

1.3 Дослідження підходів для побудови життєвого циклу артефактів

В проектах процесного управління частіше за все головною компонентою системи часто визначається саме певна діяльність чи процес. Тим не менш, при визначенні артефактів та їх життєвого циклу основна увага припадає на них, оскільки вони стають основними елементами проекту.

У дослідженні життєвого циклу артефактів насамперед необхідно визначити які саме артефакти існують в проекті, яка їх структура, властивості, тощо. Для цього можуть використовуватись різні методи і підходи, але головним способом створення артефактів та визначенням їх життєвого циклу є виявлення їх з первинних логів (асоціація подій з певними артефактами), пошук їх зв’язків з іншими елементами проекту, генерація первинних діаграм, що опишуть властивості та взаємозв’язки цих артефактів, та формування фінального відображення життєвого циклу. Цей процес візуально описаний на рисунку 1.2.

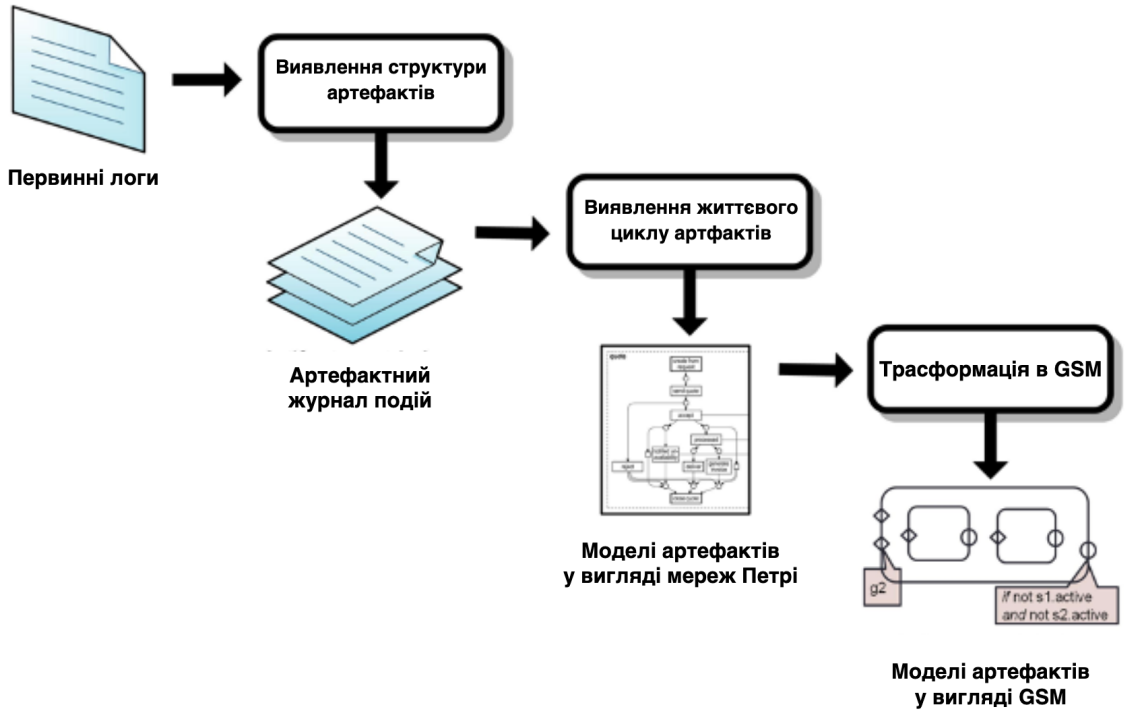

Рисунок 1.2 – Створення життєвого циклу артефакту

В результаті маємо, що для побудови життєвого циклу артефактів використовуються зазвичай два основних представлення: мережі Петрі та GSM (Guard-Stage-Milestone) моделі.

Мережі Петрі – це математична модель для моделювання систем. Вона складається з графічної частини, що складається з вузлів і стрілок, і математичної частини, яка описує розвиток системи з часом.

У мережах Петрі вузли представляють стани системи, а стрілки - переходи між станами. Модель може бути використана для аналізу систем з багатьма паралельними процесами та рівнем невизначеності. Мережі Петрі використовуються для моделювання різних процесів, таких як виробництво, маршрутизація даних, контроль систем, дизайн програмного забезпечення та

інші. Вони також можуть допомогти в проектуванні та аналізі складних систем та в управлінні проектами. Перевагами використання мереж Петрі на етапі формування первинного життєвого циклу артефакту є:

– графічна нотація. Мережі Петрі представляють зв'язки між подіями та ресурсами у вигляді графів, що можна легко зрозуміти і проаналізувати. Графічне зображення процесу допомагає простіше виявляти його недоліки та покращувати ефективність;

– моделювання поведінки. Ці мережі дозволяють моделювати поведінку артефактів в процесі їх життєвого циклу. Це дозволяє проектним менеджерам та іншим учасникам проекту зрозуміти, як артефакти будуть використовуватися в процесі, та які дії необхідні для їх успішної реалізації;

– аналіз процесу. Мережі Петрі дозволяють проаналізувати процес та виявити можливі проблеми та затримки. З їх допомогою можна виявити зв'язки між подіями чи ресурсами та визначити оптимальні шляхи виконання проекту;

– мінімізація помилок: Мережі Петрі дозволяють побудувати вірогідний сценарій життєвого циклу артефакту, що допомагає зменшити кількість помилок, які можуть приймати участь в побудові.

Отже, для побудови життєвого циклу артефакту в процесному управлінні за допомогою мереж Петрі, спочатку потрібно ідентифікувати артефакти, які існують в системі. Після цього можна побудувати життєвий цикл для кожного артефакту. Життєвий цикл артефакту можна подати у вигляді мережі Петрі, де стани моделі відображають різні етапи життєвого циклу артефакту, а переходи між станами відображають операції або події, які змінюють стан артефакту. Спочатку необхідно визначити стани моделі та переходи між станами. Стани моделі можуть відображати будь-який етап життєвого циклу артефакту, наприклад, ініціалізація, редагування, схвалення та завершення. Переходи між станами можуть відображати дії, які виконуються з артефактом (наприклад,

створення або зміна артефакту), або події, які стаються в процесі (наприклад, схвалення або відхилення артефакту).

Після визначення станів і переходів потрібно побудувати мережу Петрі, використовуючи матрицю інцидентності. Матриця інцидентності містить інформацію про залежності між станами та переходами, яка використовується для побудови мережі Петрі.

Мережа Петрі життєвого циклу артефакту може бути використана для визначення послідовності етапів, що потрібні для обробки артефакту, і контролює процес його життєвого циклу, забезпечуючи, що артефакт переходить від одного етапу до наступного. Результат створення мереж Петрі на рисунку 1.3.

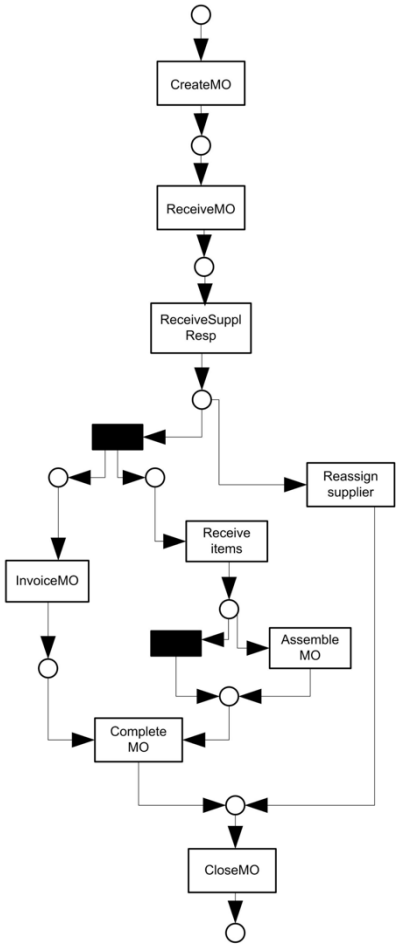

Рисунок 1.3 – Виявлення життєвого циклу артефакту за допомогою мереж Петрі

Для моделювання взаємодій між артефактами в артефакт-орієнтованих системах ми можемо використовувати нотацію Proclets [27], де кожен proclet представляє один тип артефакту у вигляді мережі Петрі, а такі конструкції, як порти та канали, можуть використовуватись для представлення різних типів взаємодій між артефактами. У цій роботі ми зосереджуємося на життєвих циклах артефактів. Відмінність від класичного виявлення процесів полягає в тому, що класичне виявлення розглядає лише одну монолітну модель процесу (мережу Петрі), тоді як ми розглядаємо набори пов'язаних мереж Петрі.

Це дозволяє нам враховувати взаємодію між різними типами артефактів та їхніми життєвими циклами. Замість того, щоб мати одну загальну модель процесу, ми можемо мати набір пов'язаних моделей артефактів, які краще відображають реальну структуру та взаємодію між артефактами.

Цей підхід до процесного управління дозволяє нам більш точно відтворити реальні процеси та взаємодії в артефакт-орієнтованих системах. Він дозволяє нам зосередитись на кожному типі артефакту окремо, розглядаючи його життєвий цикл, взаємодію з іншими артефактами та специфічні вимоги, пов'язані з цим типом артефакту.

Такий підхід виявляється особливо корисним у сферах, де взаємодія та обмін даними між різними типами артефактів є важливою частиною процесу. Він дозволяє нам аналізувати та оптимізовувати взаємодію між артефактами, виявляти можливі проблеми та покращувати ефективність процесу управління артефактами.

Однією з ключових особливостей Проклетів є їх здатність взаємодіяти один з одним через чітко визначені інтерфейси. Це дозволяє координацію та комунікацію між різними Проклетами у межах робочого процесу. Взаємодії можуть включати обмін даними, потік керування або механізми синхронізації, залежно від вимог процесу.

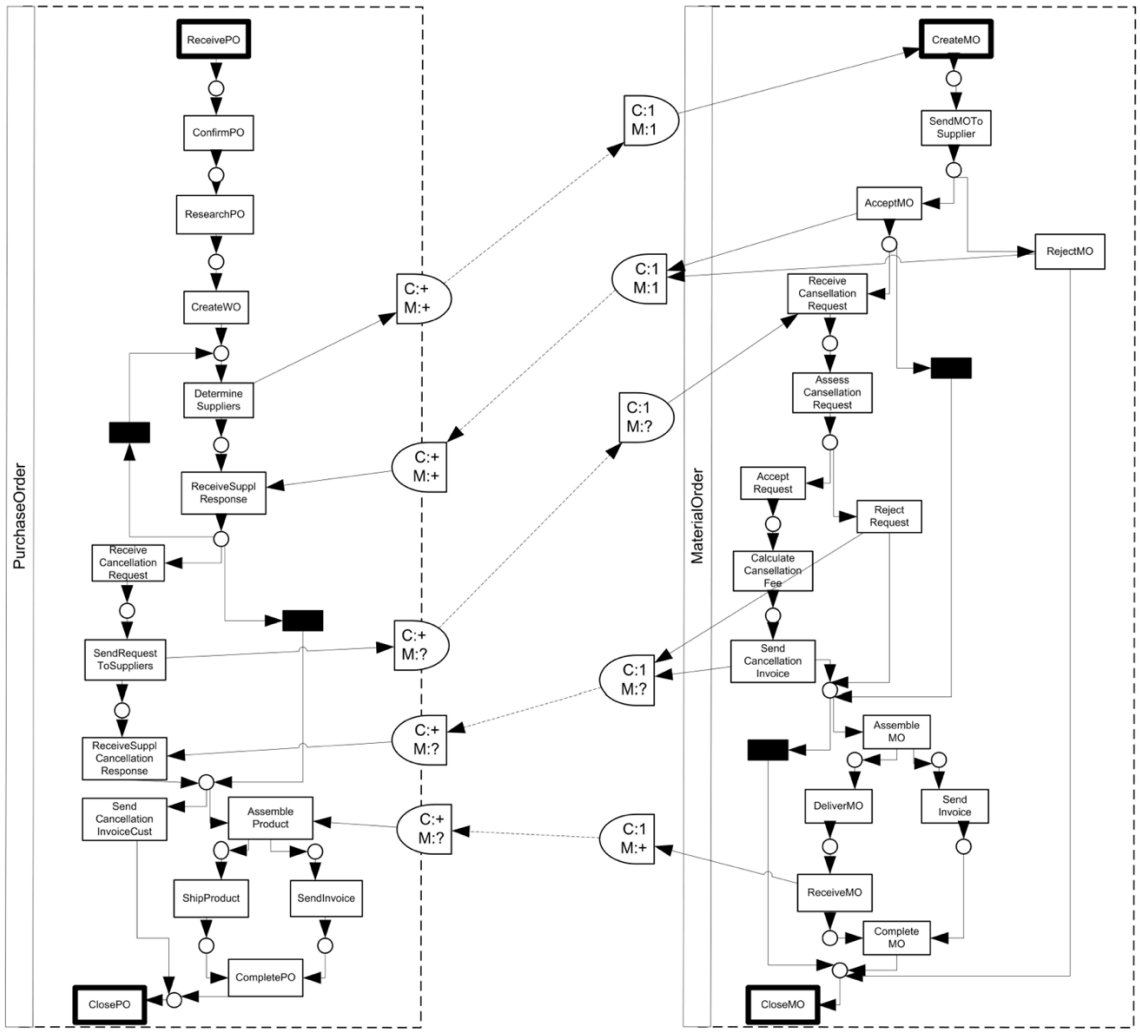

Рисунок 1.3.1 – Використання Проклетів для відображення взаємодій різних артефактів між собою в мережах Петрі

Наступним кроком після формування первинного життєвого циклу артефакту є нормалізація цих даних. Це необхідно для подальшої їх трансформації в так звану Guard-Stage-Milestone (GSM) нотацію, що повноцінно описує основні компоненти життєвого циклу та візуально їх відображає.

Існує кілька інструментів, які можна використовувати для перетворення мереж Петрі в модель Guard-Stage-Milestone (GSM). Ось кілька поширених інструментів: ProM, CPN Tools, Microsoft Visio, Python PetriNetSim.

ProM надає різноманітні плагіни та алгоритми для аналізу та перетворення мереж Петрі. Він має функціонал для виявлення, аналізу та перетворення мереж Петрі в різні моделі процесів, включаючи GSM.

CPN Tools спеціально розроблений для моделювання та аналізу Кольорових мереж Петрі (Colored Petri Nets). Він надає графічний редактор та можливості симуляції для мереж Петрі. Хоча він не має прямої підтримки GSM, ви можете використовувати CPN Tools для створення та візуалізації мереж Петрі, а потім вручну застосувати концепції GSM та анотації. Microsoft Visio використовується для діаграмування, який надає можливості для створення та маніпулювання різними типами діаграм, включаючи мережі Петрі. Ви можете використовувати елементи та з'єднувачі Visio, щоб представити елементи мережі Петрі та вручну додати етапи, майлстоуни та захисти для перетворення її в подібне до GSM представлення.

Python з бібліотекою PetriNetSim надає перевагу програмному підходу, де можливо використовувати Python разом з бібліотекою PetriNetSim, яка надає функціонал для створення, симуляції та аналіз мереж Петрі.

Отже, модель Guard-Stage-Milestone (GSM) – це методологія управління проектами, яка складається з трьох основних елементів: guard (захист), stage (етап) та milestone (результативна подія). Кожен з цих елементів має свої функції:

Захист (guard) описує умови, які мають бути виконані для етапу. Тож ця частина перевіряє, чи виконані всі критерії, необхідні для переходу до наступного етапу. Проект або артефакт може перейти до наступного етапу тільки у тому випадку якщо всі умови виконані.

Етап (stage) визначає часовий період, протягом якого виконується певна робота. Кожен етап має свої цілі, завдання та ресурси.

Результативна подія або майлстоун(milestone) є критичною точкою в проекті, яка вказує на досягнення певної фази або етапу проекту. Майлстоуни зазвичай є результатом успішного виконання етапу та можуть використовуватися як вимоги для проекту.

Тож загалом модель Guard-Stage-Milestone (GSM) використовується для управління життєвим циклом артефактів в процесному управлінні. Основна мета створення такої моделі полягає в полегшенні сприйняття та управління артефактами, які виникають під час виконання проекту.

І як наслідок, за допомогою моделі GSM може допомогти визначити, на якому етапі проекту знаходиться команда, а також забезпечує більш точну оцінку часових та ресурсних затрат на кожен етап. На відміну від альтернатив по типу Traditional Workflow Models, Business Process Model and Notation (BPMN), Event-driven Process Chains (EPC) і навіть Unified Modeling Language (UML), модель Guard-Stage-Milestone (GSM) створена спеціально для використання при виявленні та детермінації життєвого циклу артефактів, а також забезпечує більшу структуру та організованість управління проектом, що дозволяє більш точно контролювати його виконання та забезпечити успішність проекту. Приклад GSM моделі зображений на рисунку 1.4.

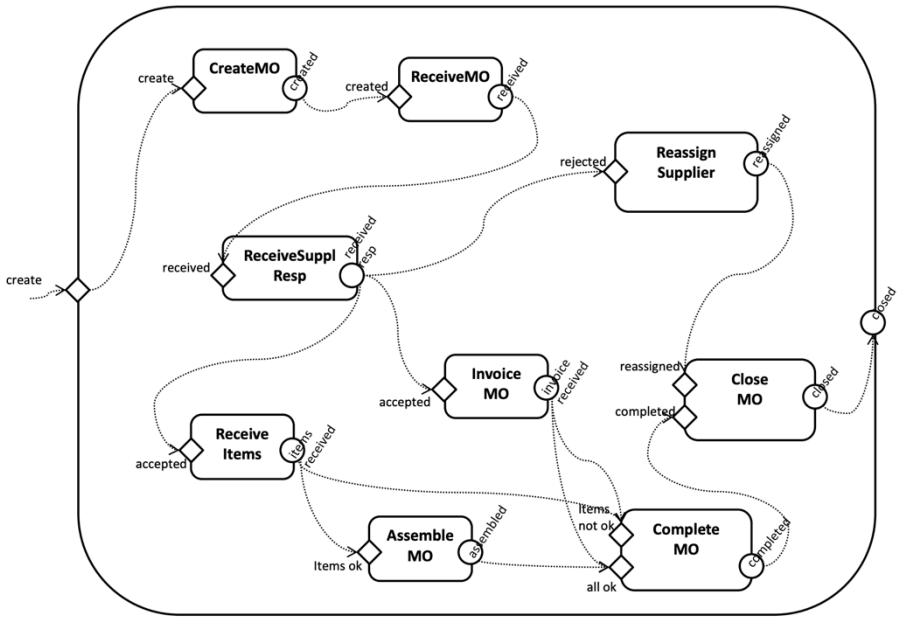

Рисунок 1.4 – GSM модель життєвого циклу артефакту

GSM також дозволяє здійснювати більш деталізований контроль за життєвим циклом артефакту. Захисти, етапи та майлстоуни можуть бути налаштовані під конкретні вимоги та обмеження системи. Це дозволяє більш ефективно та ефективно управляти життєвим циклом артефакту, оскільки модель може бути адаптована до змінних обставин та змінюваних потреб.

Модель Guard-Stage-Milestone значно більше підходить для відображення життєвого циклу артефактів, оскільки вона за своєю сутністю є декларативною, краще визначає паралельну взаємодію з об’єктом та представлена у вигляді математичної моделі скінченного автомату[3].

1.4 Постановка задачі дослідження

Актуальність дослідження обґрунтоване обмеженістю існуючих методів управління, а також моделей життєвого циклу артефактів, оскільки вони мають недоліки, є нефективними. Зазначене свідчить про важливість доречного використання процесного управління та моделей життєвого циклу артефактів. Для кожного IT-проекту важливо підтримувати високий рівень ефективності процесів та роботи.

Тож можна зазначити, що об’єктом дослідження є також IT проекти процесного управління. В свою чергу, предметом дослідження є моделі життєвого циклу артефактів в IT проектах процесного управління. Метою даної роботи можна вважати дослідження методів виявлення моделей життєвого циклу артефактів, проекти процесного управління, методи дата майнінгу, а також визначення особливостей, властивостей та характеристик артефактів IT-проектів процесного управління.

Для досягнення мети магістерської роботи необхідно вирішити такі задачі дослідження:

– аналіз підходів до процесного управління;

– аналіз властивостей та особливостей артефактів, їх життєвого циклу;

– дослідження IT-проектів процесного управління;

– виконання та експериментально перевірка реалізації артефактної моделі;

– структуризація життєвого циклу артефактів з урахуванням того, що артефакти можуть використовуватись в різних процесах.

2 МОДЕЛІ ЖИТТЄВОГО ЦИКЛУ АРТЕФАКТІВ НА ОБ'ЄКТНОМУ РІВНІ ПРОЦЕСНОГО УПРАВЛІННЯ

2.1 Методи дослідження моделей життєвого циклу артефактів

Для дослідження моделей життєвого циклу артефактів необхідно підготувати данні, що необхідні у формуванні представлення. Як було зазначено раніше в рисунку 1.2, першими кроками є формування логів (журналу подій), який буде фокусуватись артефактах, аніж процесах, як це реалізовано в консервативному представленні.

Тож дослідження моделей життєвого циклу складається з підготовки, очищення, фільтрації та обробки даних (подій), які описують як артефакти використовуються в проекті. Після цього на базі отриманих відомостей можливе створення графічне відображення життєвого циклу артефакту за допомогою мереж Петрі (первинне представлення) та GSM. Отже, процес має наступні частини:

– журнал первинних логів (raw logs). Він складається з серії всіх подій, що спостерігаються в системі. В кожній такій події зберігається, насамперед, часовий маркер, що визначає коли саме вона відбулась (або потрапила до журналу), а також колекція певних атрибутів та їх значень, що ідентифікують які саме дані приймали участь та яка операція вібдувалась;

– артефактний журнал подій (Artifact-centric logs). Оброблюючи події з первинного журналу ми можемо сформувати так званий “Artifact-centric log”, або артефактний журнал подій;

Artifact-centric log фокусуються в першу чергу самих артефактами та операціях, де вони приймають участь. Тож події в такому журналі описують переходи (transitions) та стани (states) артефактів. Такий лог формується за

допомогою визначення певного первинного ключа, що може бути або властивістю чи набором властивостей артефакту. Визначення цього ключа необхідне для того, щоб артефакт можна було ідентифікувати і чітко визначити, які саме функції проекту з ним пов’язані;

– первинне виявлення життєвого циклу (Artifact Life Cycle Discovery). Після створення артефактного журналу подій з’являється можливість сформувати схему первинну схему життєвого циклу артефактів в вигляді мереж Петрі. В свою чергу, Тож при формуванні життєвого циклу артефакту частіше за все перевага віддається моделі GSM, оскільки вона є більш гнучкою та інтуїтивною для виконання цього завдання. GSM надає просте та зрозуміле графічне представлення життєвого циклу артефакту за допомогою захистів, етапів та майлстоунів. Проектувати та підтримувати модель GSM легше, порівняно з мережами Петрі, тому що вона вимагає менше формалізму та математичних позначень;

– трасформація в GSM. Модель Guard-Stage-Milestone була представлена саме для визначення життєвого циклу артефактів. Вона представлена у вигляді скінченного автомату та підтримує ієрархічність і декларативність в межах визначеного екземпляру артефакту.

Загалом, процес виявлення та створення життєвого циклу артефакту також описується поняттям процесна аналітика (Process Mining). Process mining представлє галузь дослідження, що поєднує методи добування даних, машинного навчання, аналізу даних та моделювання процесів з метою вивчення поточних процесів у компанії, зокрема їх структури, поведінки та ефективності. Ця технологія дозволяє аналізувати реальні дані про процеси компанії, отримувати відповіді на різноманітні запитання та виявляти можливість для оптимізації процесів.

За допомогою процесної аналітики можна отримати інформацію про те, які кроки в процесах зустрічаються, скільки часу займає їх виконання, які завдання виконують різні робітники та які помилки виникають в процесі. Ця інформація може бути використана для вдосконалення процесів, зменшення витрат та покращення якості продукту або послуги. Тож процесне аналітика використовується в різних сферах бізнесу, таких як фінанси, логістика, виробництво, охорона здоров'я, а також навіть в окремих самостійних проектах.

2.2 Розробка моделі життєвого циклу артефактів з використанням подій процесного управління

Розробка моделі життєвого циклу артефактів дає змогу відстежувати життєвий цикл артефактів від їх створення до виконання і видалення. Одним з інструментів для створення моделі життєвого циклу артефактів є події процесного управління. Для реалізації цих завдань використовуються програмні засоби, які дозволяють відстежувати події, що відбуваються в системі, і збирати інформацію про процес.

Як було визначено, першим кроком у розробці життєвого циклу артефакту є створення журналу подій, який буде ідентифікувати артефакти та визначати як саме вони використовуються, що створюється на основі звичайного журналу подій, який надає інформацію про перебіг процесів в проекті (рисунок 2.1.)

11-24,17:12	ReceivePO	items=(it0), POrderID=1
11-24,17:13	CreateMO	supplier=supp6, items=(it0), POrderID=1, MOrderID=1
11-24,19:56	ReceiveMO	supplier=supp6, items=(it0), POrderID=1, MOrderID=1
11-24,19:57	ReceiveSupplResp	supplier=supp6, items=(it0), POrderID=1, MOrderID=1, answer=accept
11-25,07:20	ReceiveItems	supplier=supp6, items=(it0), POrderID=1, MOrderID=1
11-25,08:31	Assemble	items=(it0), POrderID=1, MOrderID=1
11-25,08:53	ReceivePO	items=(it0,it1,it2,it3), POrderID=2
11-25,12:11	ShipPO	POrderID=1
11-26,09:30	InvoicePO	POrderID=1
11-26,09:31	CreateMO	supplier=supp1, items=(it1,it2,it3), POrderID=2, MOrderID=2
11-28,08:12	ReceiveMO	supplier=supp1, items=(it1,it2,it3), POrderID=2, MOrderID=2
11-28,12:22	CreateMO	supplier=supp4, items=(it0), POrderID=2, MOrderID=3
12-03,14:34	ClosePO	POrderID=1
12-03,14:54	ReceiveMO	supplier=supp4, items=(it0), POrderID=2, MOrderID=3
12-03,14:55	ReceiveSupplResp	supplier=supp1, items=(it1,it2,it3), POrderID=2, MOrderID=2, answer=accept
12-04,15:02	ReceivePO	items=(it1,it2), POrderID=3
12-04,15:20	ReceiveSupplResp	supplier=supp4, items=(it0), POrderID=2, MOrderID=3, answer=accept
12-04,15:33	CreateMO	supplier=supp2, items=(it2), POrderID=3, MOrderID=4
12-04,15:56	ReceiveMO	supplier=supp2, items=(it2), POrderID=3, MOrderID=4
12-04,16:30	CreateMO	supplier=supp5, items=(it1), POrderID=3, MOrderID=5
12-05,09:32	ReceiveMO	supplier=supp5, items=(it1), POrderID=3, MOrderID=5
12-05,09:34	ReceiveItems	supplier=supp4, items=(it0), POrderID=2, MOrderID=3
12-05,11:33	ReceiveItems	supplier=supp1, items=(it1,it2,it3), POrderID=2, MOrderID=2
12-05,11:37	Assemble	items=(it0), POrderID=2, MOrderID=3
12-05,11:50	ReceiveSupplResp	supplier=supp2, items=(it2), POrderID=3, MOrderID=4, answer=reject
12-05,13:03	ReceiveSupplResp	supplier=supp5, items=(it1), POrderID=3, MOrderID=5, answer=accept
12-06,05:23	Assemble	items=(it1,it2,it3), POrderID=2, MOrderID=2
12-06,05:25	ReassignSupplier	items=(it2), POrderID=3, MOrderID=4
12-06,07:14	ReceiveItems	supplier=supp5, items=(it1), POrderID=3, MOrderID=5
12-06,07:15	Assemble	items=(it1), POrderID=3, MOrderID=5
12-06,07:25	InvoicePO	POrderID=2
12-06,09:34	ShipPO	POrderID=2
12-12,20:41	CreateMO	supplier=supp5, items=(it2), POrderID=3, MOrderID=6
12-12,20:50	ReceiveMO	supplier=supp5, items=(it2), POrderID=3, MOrderID=6
12-13,03:20	ReceiveSupplResp	supplier=supp5, items=(it2), POrderID=3, MOrderID=6, answer=accept
12-13,03:21	ReceiveItems	supplier=supp5, items=(it2), POrderID=3, MOrderID=6
12-13,04:30	ClosePO	POrderID=2
12-13,08:36	Assemble	items=(it2), POrderID=3, MOrderID=6
12-13,08:37	InvoicePO	POrderID=3
12-13,08:38	ShipPO	POrderID=3
12-13,08:39	ClosePO	POrderID=3

Рисунок 2.1 – Первинний журнал подій (raw logs)

Використовуючи первинний журнал подій (raw logs), ми можемо відфільтрувати події, які не визначають жодні артефакти та почати ідентифікувати та формувати сутності, які будуть потім відображати конкретні артефакти проекту. В результаті буде отримано новий артефактний журнал подій. Артефактний журнал подій представляє собою перелік, який містить інформацію про всі події, пов'язані з життєвим циклом артефакту в процесі його створення, використання та знищення. Кожна подія має свій часовий штамп та

атрибути, які характеризують стан артефакту на момент події. Він може бути використаний для аналізу та вдосконалення процесів, пов'язаних з артефактами, а також для виявлення проблем та недоліків у цих процесах. Процес створення артефактного журналу подій описаний на рисунку 2.2.

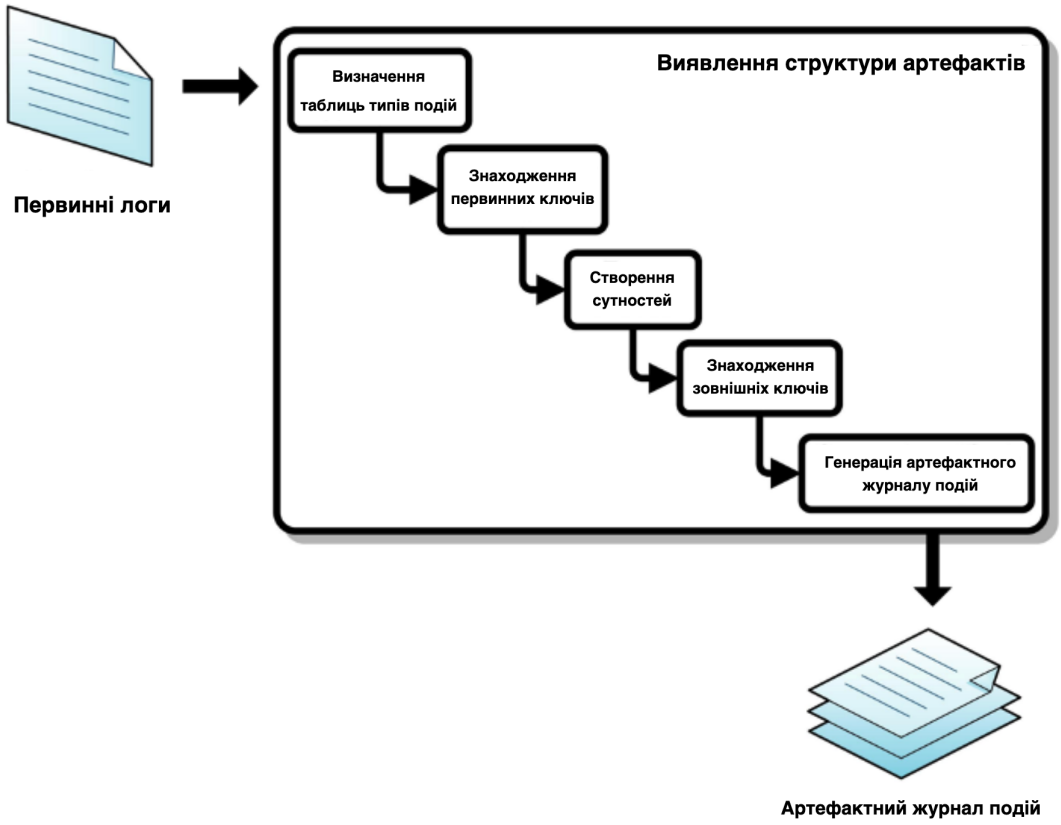

Рисунок 2.2 – Створення артефактного журналу подій (ACL)

– спочатку ми визначаємо таблиці типів для подій, де будуть зазначені які атрибути та значення властиві певній обраній події;

Подія можна бути представити у вигляді формули 2.1:

$$\{A1, A2, \ldots, An\} \text{ and } \{D1, D2, \ldots, Dn\}, \quad (2.1)$$

де *Ai* – є назвами атрибути;

Di – представляють можливі значення *Ai*.

– після цього, коли ми маємо таблицю атрибутів, що описують подію, ми можемо чітко виділити первинний ключ (Primary Key), що може бути однією властивістю подію, або совокупністю декількох характеристик;

– з визначиними атрибутами та первинним ключем можна створити сутність (Entity). Сутність представляє собою кортеж типових таблиць подій, що між собою розділяють первинний ключ, який унікально ідентифікує ці сутності.

Нехай *T* являє собою кортеж таблиць кластерів типів, які беруться з початкового журналу подій. *L*. Тоді *A* визначає первинний ключ для таблиць в кортежі *T*. Сутність *E* для *T* визначається як рівняння:

$$\begin{aligned} \mathrm{T} &= \{\mathrm{T1}, \mathrm{T2}, \ldots, \mathrm{Tm}\} \\ \mathrm{A} &= \{\mathrm{A1}, \mathrm{A2}, \ldots, \mathrm{An}\} \\ \mathrm{E} &= (\mathrm{T}, \mathrm{A}), \end{aligned} \tag{2.2}$$

де *A* – це екземпляр ідентифікатора для *E*.

Тоді маємо, що пара атрибут-значення *(Ai , di)* знаходяться в межає і $n + 1 \leq i \leq l$, *iff* та сутність $t \in T$, $T \in T$, *s.t.* $t.Ai = di$ тож для всіх $1 \leq k \leq n$ $t.Ak = dk$. Таким чином визначити екземпляр можна використовуючи формулу:

$$\mathrm{s} = ((\mathrm{A1}, \mathrm{d1}), \ldots, (\mathrm{An}, \mathrm{dn}), (\mathrm{An{+}1}, \mathrm{dn{+}1}), \ldots, (\mathrm{Al}, \mathrm{dl})), \tag{2.3}$$

де *dk* – знаходиться в межах $1 \leq \mathrm{k} \leq \mathrm{n}$ t.Ak.

– після цього можуть бути визначені зовнішні ключі (Foreign keys), які відображають залежності одних сутностей від інших.

Нехай база даних *D* представляє собою кортеж *T*, що складається з таблиць з відповідною схемою *S(T)*.

$$\mathrm{S(T)}, \mathrm{T} \in \mathrm{T} \rightarrow \mathrm{D} = (\mathrm{T}\ , \mathrm{K}), \tag{2.4}$$

де *K* – це ключовий зв'язок.

Таким чином, що їх атрибути не пересікаються і тому ключовий взаємозв'язок *K*. Ключовий зв'язок *K* відображає відношення зовнішнього ключа до первинного ключа між таблицями *T* : стверджуємо, що:

$$\begin{gathered} \mathrm{K} \subseteq (\mathrm{A(T}\ ) \times \mathrm{A(T}\ ))\mathrm{N} \\ (\mathrm{A1}, \mathrm{A0}\ 1\ ), \ldots ,(\mathrm{Ak}, \mathrm{A0\ k}\ )\ \in \mathrm{K} = \mathrm{T} \in \mathrm{T} = 0 \in \mathrm{T}\ \rightarrow \\ \mathrm{A0\ k} \in \mathrm{A(T}\ 0\ )\ \mathrm{of\ T}\ 0, \end{gathered} \tag{2.5}$$

де *Ak* – знаходиться в межах $1 \leq \mathrm{k} \leq \mathrm{n}$ A0 k.

– в результаті попередніх дій ми отримуємо повноцінний артефактний журнал подій, що виділяє головним чином не процеси і їх перебіг, а взаємодію та операції, що проводяться разом з артефактами проекту[16].

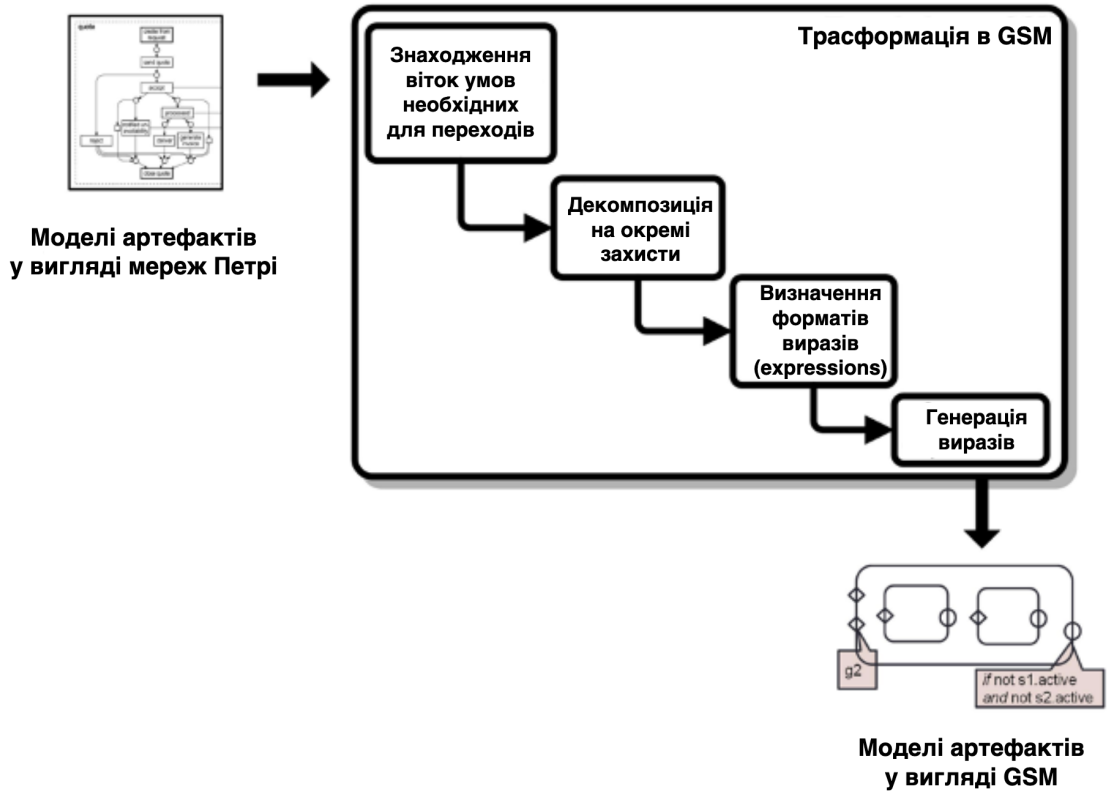

Рисунок 2.3 – Трасформація мереж Петрі в модель Guard-Stage-Milestone

Після отримання артефкатного журналу подій можна створити мережі Петрі, які зможуть візуально відобразити життєвий цикл артефактів на цьому етапі. Таким чином ми отримаємо так зване первинне відображення, але ціль створення мереж буде подальше їх використання у формуванні більш точної, гнучкої та відповідної GSM моделі. Ця трансформація описана в рисунку 2.3.

Мережі Петрі можна створити за допомогою декількох способів, серед яких можна виділити ILP Miner та Tree Miner. ILP Miner (майнер інтервального лінійного програмування) – це алгоритм майнінгу процесів, який використовує техніки інтервального лінійного програмування для виявлення моделей мереж Петрі з журналів подій. Цей алгоритм розроблений для роботи з проблемою

"ексклюзивного вибору", яка виникає, коли декілька переходів конкурують за одне вхідне або вихідне місце.

Щоб створити мережу Петрі з використанням ILP Miner, спочатку потрібно надати йому файл журналу подій, який містить список подій, що відбулися у процесі. Алгоритм аналізує журнал подій, щоб визначити можливі переходи, місця та зв'язки в моделі мережі Петрі.

Алгоритм ILP Miner працює, спочатку генеруючи набір кандидатів на моделі мереж Петрі, кожен з різною структурою та кількістю зв'язків. Потім він використовує техніку оптимізації інтервального лінійного програмування, щоб вибрати найкращу модель мережі Петрі, яка відповідає спостережуваним журналам подій. ILP Miner є це досить складний алгоритм майнінгу процесів, який потребує розуміння лінійного програмування та моделювання мереж Петрі. Однак він може бути потужним інструментом для автоматичного виявлення моделей зі складних журналів подій.

Tree Miner є також алгоритмом процесного майнінгу, який можна використовувати для створення моделей Петрі з журналів подій. На відміну від ILP Miner, який використовує техніки цілочисельного програмування, Tree Miner використовує алгоритми дерева рішень для виявлення структури моделі Петрі. Для використання Tree Miner для генерації моделей Петрі необхідно спочатку надати йому файл журналу подій. Алгоритм аналізує журнал, щоб ідентифікувати можливі переходи, місця та дуги в моделі Петрі. Алгоритм Tree Miner працює шляхом побудови дерева рішень, яке описує взаємозв'язок між подіями в журналі. Це дерево рішень потім використовується для створення моделі Петрі, яка відображає спостережувану поведінку в журналі.

Tree Miner може бути корисним інструментом для генерації моделей Петрі з журналів подій, особливо в складних процесах, де є багато можливих шляхів через процес. Однак, важливо зазначити, що згенерована модель Петрі може не

бути найбільш оптимальною або ефективною для процесу, і може потребувати ручного уточнення та оптимізації.

ILP Miner і Tree Miner є два популярними алгоритми process mining, які використовуються для виявлення моделей процесів з журналів подій. Хоча обидва алгоритми спрямовані на побудову моделі процесу з даних журналу подій, вони відрізняються у способі подання виявленої моделі.

ILP Miner (Integer Linear Programming Miner) є підходом на основі правил, який використовує логічне програмування для вивчення моделі Petri з даних журналу подій. Він створює набір правил у вигляді логічних висловлювань, які описують патерни та залежності між подіями. Правила потім перетворюються у модель Petri.

З іншого боку, Tree Miner є підходом на основі дерева, який будує дерев'яну структуру з даних журналу подій. Дерев'яна структура представляє різні шляхи, які беруться подіями в журналі. Шляхи потім комбінуються для формування моделі Petri.

Один із переваг ILP Miner полягає в тому, що він може обробляти більш складні моделі процесу, ніж Tree Miner. ILP Miner може вивчати Petri зі складними залежностями між діяльностями, включаючи конструкції XOR та OR-split. Tree Miner, з іншого боку, обмежений більш простими моделями та може не впоратися з більш складними потоками процесів.

Однак Tree Miner має перевагу в тому, що він швидше за ILP Miner та може ефективніше обробляти більші журнали подій. Крім того, Tree Miner є простішим для розуміння та реалізації порівняно з ILP Miner.

Також важливо згадати, що є і інші інструменти, що використовуються в process mining, такі як Alpha Miner, що заснований на аналізі прямої залежності між подіями та побудові моделі процесу з використанням діаграми потоку подій (Event Flow Graph); Heuristics Miner, який користується евристичними правила

для виявлення моделів процесу з журналу подій та базується на аналізі послідовностей подій, використовуючи правила по типу прямого з'єднання, причина-наслідок та розділення; Fuzzy Miner, який використовує нечіткі правила для виявлення моделі процесу з журналу подій та дозволяє враховувати невизначеність та неповноту даних та створює нечіткі правила, які відображають відносну частоту взаємодій між подіями; Genetic Miner, що опирається на генетичні алгоритми для виявлення моделі процесу з журналу подійта має еволюційний підхід, де кращі моделі процесу відбираються та комбінуються для створення нових поколінь моделей; а також Transition System Miner, який використовує поняття переходів між станами для виявлення моделі процесу з журналу подій та аналізує послідовності подій, після чого створює переходи між станами, що відображають частоту та послідовність взаємодій між подіями

В результаті використання рекомендованих ILP Miner чи Tree Miner, ми отримуємо мережі Петрі, які наразі вже можуть відображати життєвий цикл артефактів проекту та які можуть вже використовуватись для подальшої генерації Guard-Stage-Milestone моделі. Щоб перетворити мережу Петрі на модель GSM, потрібно виконати всього декілька операцій.

Перше, необхідно визначити стани, які є кінцевими станами процесу і позначте їх як відповідні знаки мілітону в моделі GSM. Після цього можна знайти стани, які є стартовими станами процесу і позначаються як відповідні знаки захистів(Guard) в моделі GSM. Наступними визначаються стани, які є проміжними станами процесу, тобто ті стани, які не є або кінцевими, або стартовими станами. Вони записуються як відповідні знаки етапів в моделі GSM.

Додаються дуги між станами мережі Петрі відповідно до залежностей між знаками в моделі GSM та знаки «start» і «stop», які позначають початок та кінець процесу в моделі GSM. В результаті отримана модель GSM перевіряється на коректність та відповідність оригінальній мережі Петрі.

3 ПЛАНУВАННЯ ІТ ПРОЕКТУ МОДУЛЮ ПОБУДОВИ АРТЕФАКТІВ НА ОСНОВІ АНАЛІЗУ ЛОГІВ

3.1 Суть та опис проекту

Проект полягає у створенні системи, що дозволяє будувати артефакти на основі аналізу логів. Артефакти, які будуть створюватись, можуть бути різного типу, включаючи мережі Петрі, діаграми потоків даних, графіки та інші.

Основна мета проекту - забезпечити автоматичне створення артефактів на основі аналізу логів, що дозволить значно зменшити час та зусилля, які зазвичай потрібні для їх створення вручну. При цьому, система повинна бути простою та зручною у використанні, щоб користувачі з різним рівнем експертизи могли легко з нею працювати.

Система буде базуватись на алгоритмах аналізу логів, таких як ILP Miner та Tree Miner, що дозволяють автоматично створювати мережі Петрі на основі логів. Для діаграм потоків даних та інших артефактів будуть використовуватись інші алгоритми та методи.

Система буде мати інтерфейс, що дозволить користувачам завантажувати логи, вибирати тип артефакту та налаштовувати параметри генерації. Для зручності користувачів буде створено документацію та навчальний матеріал, що допоможе їм ознайомитись з системою та її можливостями.

Очікуваний результат проекту - створення системи, що дозволяє автоматично створювати артефакти на основі аналізу логів, що значно зменшить час та зусилля, потрібні для створення артефактів вручну. Це зробить процес створення артефактів більш ефективним та продуктивним.

Проект передбачає побудову артефактів на основі аналізу логів з метою виявлення та оптимізації процесів в організації. Для досягнення цієї мети, в

проекті будуть використані різні алгоритми процесного майнінгу, зокрема ILP Miner, Tree Miner та GSM.

У першу чергу, для проекту необхідно мати доступ до логів процесів, що відбуваються в організації. Ці логи будуть використані для побудови моделей процесів та виявлення потенційних проблем, таких як затримки, збої, надмірні витрати тощо.

Для побудови моделей процесів будуть використані різні алгоритми процесного майнінгу, які дозволяють автоматично побудувати моделі процесів на основі логів. Використання різних алгоритмів дозволить отримати різні моделі, що дозволить зробити більш об'єктивний аналіз процесів.

Побудовані моделі будуть допомагати виявляти можливі проблеми та затримки в процесах, а також здійснювати їхню оптимізацію. Наприклад, модель процесу може вказувати на те, що деякі кроки можна автоматизувати або покращити, що дозволить зменшити час виконання процесу та скоротити витрати.

Для реалізації цього проекту ми плануємо використовувати різні методи аналізу даних та машинного навчання, зокрема класифікацію, кластеризацію, регресійний аналіз та асоціативні правила. Ми також плануємо використовувати інструменти процесного аналізу, такі як ILP Miner та Tree Miner, для виявлення і побудови моделей процесів на основі логів подій.

Таким чином, проект передбачає використання алгоритмів процесного майнінгу для побудови моделей процесів та оптимізації процесів в організації, що може призвести до підвищення ефективності та ефективності роботи організації.

3.2 Обмеження проекту та його життєвий цикл

ІТ-проекти процесного управління також мають свої обмеження, які можуть залежати від різних факторів, таких як:

– бюджет. Одним з головних обмежень є бюджет проекту. Недостатній бюджет може обмежувати можливості проекту в плані відповідного функціоналу, якість реалізації, терміни та ресурси. Обмеження бюджету можуть змусити команду проекту знайти компромісні рішення, які не завжди будуть оптимальними;

– терміни. Обмеження термінів можуть змушувати компанії вибирати менш оптимальні рішення, або пропустити певні етапи розробки;

– технології. Обмеження технологій можуть призвести до обмеження функціональних можливостей продукту. Також важливо враховувати сумісність з іншими системами, яка може бути обмежена в залежності від технології використання;

– людський фактор. Обмеження в людських ресурсах, знаннях та навичках можуть вплинути на якість та швидкість розробки проекту. Недостатня кваліфікація команди може призвести до погіршення якості продукту та зниження продуктивності;

– середовище. Обмеження, які стосуються середовища, можуть включати в себе доступність ресурсів, обмеження щодо сумісності зі старішими версіями програмного забезпечення або обмеження щодо використання в певних середовищах.

Врахування цих обмежень є дуже важливим для успішної розробки ІТ-проекту процесного управління. Керування проектом та ретельне планування

можуть допомогти зменшити вплив цих обмежень на проект і забезпечити виконання проекту в обумовлених рамках.

Життєвий цикл зображує основні фази реалізації проекту за конкретний період часу. Окремі фази та етапи розробки модулю відображено в таблиці 3.1.

Таблиця 3.1 – Життєвий цикл проекту

Етап	Ініціація та первинний аналіз	Планування реалізації	Реалізація та моніторинг виконання	Введення в експлуатацію
Дати початку	01.05.2023	10.05.2023	20.06.2023	20.06.2023
Дати завершення	10.05.2021	20.05.2021	20.06.2021	30.06.2023
Мета та основні операції	Визначення мети, цілей, завдань, учасників та специфікацій проекту. Запровадження основних понять та етапів.	Планування виконання задач та цілей. Визначення бюджетів. ресурсів та процесів проекту.	Виконання поставлених цілей, розробка та тестування технічного рішення. Підготовка до введення до експлуатації.	Введення рішення в роботу для третіх осіб. Збір аналітики використання продукту та аналіз наступних планів.

3.3 Визначення складу учасників проекту, організаційна структура виконавців та матриця відповідальностей

Учасниками проекту є люди, організації або інші структури, які займають певні ролі в проекті. Визначення учасників допомагає зрозуміти, хто з ким буде працювати на проекті, які є очікування кожного учасника, та які ресурси будуть необхідні для успішного завершення проекту. У визначення учасників проекту також можуть бути включені ролі та відповідальності кожного учасника. Це допомагає уникнути дублювання зусиль та необхідних дій, а також забезпечити, що кожен учасник знає, що його очікує в проекті та які результати він повинен досягти.

Тож для успішного виконання проекту важливо знати, хто є учасниками проекту та які є їх ролі. Визначення учасників проекту допомагає забезпечити кращу комунікацію, визначити відповідальності, виділити ресурси та визначити потреби в тренуванні та розвитку.

Тим не менш, з точки зору процесного управління, не кожна особа або група є учасником IT проекту. Особи, які не мають прямого впливу на виконання процесів IT проекту або не залучаються до них, не можуть бути вважатися учасниками проекту. Як приклад, учасниками не вважаються керівники або управлінський персонал, які не беруть безпосередню участь у виконанні процесів проекту, Інші відділи або підрозділи компанії, Зовнішні стейкхолдери, такі як клієнти, постачальники або зовнішні консультанти, які не виконують роль учасників процесів IT проекту, Інші співробітники компанії, які не мають прямого впливу на процеси IT проекту і не залучаються до їх виконання.

У проекті зі створення артефактів на основі аналізу логів можуть брати участь різні фахівці, такі як аналітики даних, розробники програмного

забезпечення, тестувальники, менеджери проекту та замовник. А отже учасники проекту мають різні ролі та приймають різну участь в його реалізації. Незважаючи на це, завданням всіх учасників є супроводження проекту від його початкової фази до кінцевої. Для проекту було виділено декілька ключових учасників: власник проекту, проектний менеджер та команда. Всі ці ролі відносяться до так званої внутрішньої сторони проект. Але також може бути зовнішня, якщо проект, наприклад, був виконаний на замовлення. В такому випадку власник є зовнішньою сторону проекту. Таблиця 3.2 відображає учасників проекту.

Таблиця 3.2 – Учасники проекту

Учасник / Етап	Ініціація та первинний аналіз	Планування реалізації	Реалізація на моніторинг виконання	Введення в експлуатацію
Власник проекту	Проводить первинний аналіз продукту			Перевірає задоволення всіх вимог, цілей та задач
Проектний менеджер	Приймає участь у аналізі продукту, визначає вимоги системи	Впроваджує конкретні цілі, мету, процеси, вимоги та специфікацію проекту	Контролює та керує виконанням поставлених задач та цілей	Відслідковує вдале введення продукту в експлуатацію та подальшу його роботу
Команда		Приймає участь в оцінюванні запланованих завдань та цілей	Реалізовову та виконує завдання. Веріфікує результат роботи	Вводить кінцеву версію продукту в експлуатацію

Організаційна структура виконавців IT проекту процесного управління - це розподіл відповідальності та ролей між членами команди, що включає керівників, розробників, тестувальників та інших спеціалістів. Вона допомагає забезпечити ефективну організацію процесів розробки та впровадження продукту або послуги, зменшити можливі ризики та збільшити шанси на успіх проекту. На рисунку 3.3 відображається така організаційна ієрархічна структура проекту для даного проекту.

Рисунок 3.3 – Організаційна ієрархічна структура проекту

Завдяки формуванню організаційної ієрархічної структури проекту, яка ще може називатись OBS, та ієрархічної структури декомпозиції робіт (OWS) можна створити матрицю відповідальностей.

Матрицею відповідальності (англ. Responsibility Assignment Matrix, RAM) є інструмент управління проектами, що використовується для визначення ролей та відповідальності за конкретні завдання або етапи проекту між різними учасниками команди проекту.

У матриці відповідальності прописуються всі завдання проекту та призначається конкретний виконавець або група виконавців для кожного завдання. Матриця відображає, які учасники проекту відповідають за які завдання та наскільки вони несуть відповідальність за успіх проекту.

Матриця відповідальності може бути представлена у вигляді таблиці, де рядки представляють завдання, а стовпці - учасників команди. У кожній клітинці таблиці вказується рівень відповідальності конкретного учасника проекту за виконання певного завдання. Рівень відповідальності може бути позначений спеціальними абревіатурами, наприклад, RACI (Responsible, Accountable, Consulted, Informed), що допомагає однозначно зрозуміти ролі та відповідальність учасників команди проекту.

Матриця відповідальності допомагає уникнути дублювання роботи, підвищити ефективність співпраці між учасниками проекту, покращити комунікацію та зменшити можливість конфліктів.

Для створення матриці відповідальності необхідно спочатку зазначити основні активності проекту: З - затвердження та узгодження, У - управління та координація роботи, В - виконання роботи, П - перевірка результату, контроль розробки. Після формування основних активностей проекту, організаційної ієрархічої структури проекту, ієрархічної структури декомпозиції робіт була сформована матриця відповідальностей, що наведена в таблиці 3.3.

Таблиця 3.3 – Матриця відповідальностей

Фаза проекту	Власник проекту	Проектний менеджер	Команда
Ініціація роботи над проектом			
1. Планування технічного проекту	З	У П	В
1.1 Аналіз функціональних вимог системи		У	В
1.1.1 Аналіз програмного забезпечення		У	В
1.2 Аналіз можливостей розробки		У	В
2. Розробка технічного рішення	З	У П У	В
2.1 Розробка концепції проектування		У П	В
2.2 Розробка і вибір варіантів концепції за вимогами замовника		У	В
2.3 Розробка ескізного проекту		У	В
2.4 Розробка проектних рішень по системі та її частинам		У	В
3. Проведення тестування	З	У П	В
3.1 Проведення мануального та автоматизованого тестування		У	В
3.1.1 Визначення тест кейсів		У	В
3.1.1. Планування тестування та перевірки системи		У	В
4. Введення проекту в експлуатацію	З	У П В	У П В
4.1 Презентація фінального проекту	П	У В	
4.2 Підготовка документації для подальшої підтримки		У П	П В

Отже, матриця відповідальності є інструментом управління проектами, який використовується для визначення ролей та обов'язків учасників проекту. Ця матриця представляється у вигляді таблиці, в якій по вертикалі відображені завдання, а по горизонталі - учасники проекту. Кожному завданню та учаснику призначається роль або обов'язок, який він повинен виконувати.

Матриця відповідальності дозволяє краще розуміти, хто відповідає за які завдання, забезпечує більш ефективне розподілення ресурсів, зменшує ризики виникнення конфліктів та сприяє більш ефективному керуванню проектом. Вона також допомагає кожному учаснику проекту бути впевненим у своїй ролі та зрозуміти, як його робота сприяє загальному успіху проекту.

Таким чином за допомогою матриці відповідальностей можна чітко розділити межі та цілі всіх учасників проекту, а також формалізувати їх конкретні обов’язки.

3.4 Мережевий графік проектування

Мережевий графік – це динамічна модель виробничого процесу, що відображає технологічну залежність і послідовність виконання комплексу робіт. Тобто, графік погоджує завершення робіт у часі з урахуванням витрат ресурсів і вартості робіт з виділенням під час цього критичних місць. Основні елементи мережевого графіка — робота і подія. Робота відображає трудовий процес, в якому беруть участь люди, машини, механізми, матеріальні ресурси або процес очікування. Кожна робота мережевого графіка має конкретний зміст. Робота як трудовий процес вимагає витрат часу і ресурсів, а як очікування — тільки часу. В таблиці 3.4 наведен мережевий графік.

Таблиця 3.4 – Мережевий графік

Назва задачі	Витрати ресурсів	Тривалість	Початок	Кінець
1. Планування технічного проекту	50 годин	10 днів	15.05.23 10:00	30.05.23 19:00
1.1 Аналіз функціональних вимог системи	35 годин	3 дні	15.05.23 10:00	18.05.23 16:00
1.1.1 Аналіз програмного забезпечення	4 години	3 дні	15.05.23 16:00	18.05.23 19:00
1.2 Аналіз можливостей розробки	13 годин	1 день	16.05.23 10:00	16.05.23 18:00
2. Розробка технічного рішення	5 годин	1 день	16:05.23 10:00	17.05.23 10:00
2.1 Розробка концепції проектування	15 годин	7 днів	18.05.23 9:00	25.05.23 20:00
2.2 Розробка і вибір варіантів концепції за вимогами замовника	7 годин	2 дні	18.05.23 9:00	20.05.23 20:00
2.3 Розробка ескізного проекту	8 годин	2 дні	18.05.23 9.00	20.05.23 20:00
2.4 Розробка проектних рішень по системі та її частинам	54 години	10 днів	30.05.23 20:00	10.06.23 13:00
3. Проведення тестування	30 годин	15 днів	01.06.23 13:00	15.06.23 17:00
3.1 Проведення мануального та автоматизованого тестування	10 годин	5 днів	01.06.23 17:00	05.06.23 12:00
3.1.1 Визначення тест кейсів	20 годин	5 днів	01.06.23 12:00	05.06.23 13:00
3.1.1. Планування тестування та перевірки системи	20 годин	4 днів	05.06.23 13:00	09.06.23 18:00
4. Введення проекту в експлуатацію	10 годин	5 днів	05.06.23 18:00	10.06.23 14:00

3.5 Оптимізація проекту за показниками час – вартість та ресурсами

Вартість проекту визначається сукупністю вартостей ресурсів проекту, вартостями і часом робіт проекту. Управління вартістю проекту забезпечує його завершення в рамках затвердженого бюджету і включає процеси планування ресурсів, оцінки вартості, розробки бюджету та управління вартістю. Основним документом, за допомогою якого здійснюється управління вартістю проекту, є бюджет.

Бюджетування проекту. Звітність за витратами. Оптимізація плану проекту за показником час / вартість (метод PERT / COST).

PERT (Program Evaluation and Review Technique) є технікою оцінки та аналізу програми або сукупності програм. Тобто в першу чергу цей метод використовується для передбачення можливих ризиків, пов’язаних з часом виконання певних робіт. PERT за основу бере 3 оцінки:

Оптимістичну (Optimistic - O)

Песимістичну (Pessimistic - P)

Вірогідну (Most likely - M)

PERT естімейт вираховується за допомогою формули:

(O + (M*4) + P) / 6

Якщо провести розрахунок оцінки роботи, що відноситься до проекту курсової, то передбачення є виконання всіх операцій має наступний вигляд:

Таблиця 3.5 – Оптимізація проекту за показниками час – вартість.

№	Операція	Оптимістично	Вирогідно	Песимістично	Оцінка
1	Планування технічного проекту	2	5	8	5
2	Аналіз функціональних вимог системи	5	5	10	5,5
3	Аналіз програмного забезпечення	2	4	6	4
4	Аналіз можливостей розробки	1	3	4	2,83
5	Розробка технічного рішення	7	13	20	13,16
6	Розробка концепції проектування	5	10	15	10
7	Розробка і вибір варіантів концепції за вимогами замовника	3	7	11	7
8	Розробка ескізного проекту	2	3	5	3,16
9	Розробка проектних рішень по системі та її частинам	5	10	16	10,16
10	Проведення мануального та автоматизованого тестування	6	12	24	13
11	Визначення тест кейсів	3	6	10	6,16
12	Планування тестування та перевірки системи	2	4	7	4,16
13	Презентація фінального проекту	2	4	7	4,16
14	Підготовка документації для подальшої підтримки	5	12	17	11,6
				Всього	99,89

Метод оптимізації ресурсів використовується для регулювання дат старта та фініша різних оперцій задля того, щоб оптимізувати їх зайнятість та ефективно використовувати наявні потужності.

Оптимізація проекту за ресурсами в процесному управлінні має на меті ефективне використання ресурсів, таких як людські ресурси, матеріали, обладнання та фінансові ресурси. Проведення оптимізації проекту дозволяє ефективно використовувати ресурси (прозорість створює умови для виявлення найбільш оптимізованих методів досягнення цілей), здійснювати контроль витрат, зменшувати кількість витраченого часу (завдяки більш продуктивному розподілу ресурсів чи роботи), мінімізувати ризики (оскільки стає простіше виявити ресурси, що можуть використовуватись неправильно чи малорезультативно) та підвищити якість продукту шляхом збільшення загального рівня процесів, розробки та завдяки забезпеченню належного контролю на час виконання проекту.

В рамках даного проекту було використано метод вирівнювання ресурсів для оптимізації їх використання, що нормалізує рівновагу між запитом на окремі потужності та їх наявністю. Цей підхід може використовуватись як для планування зайнятості ресурсів загального характера, так і для критично важливих, чия доступність може бути обмежена певними умовами (наприклад, часом). Для вирівнювання ресурсів використовується доступний резерв часу, який використовується для рівномірного розподілу ресурсів за наявними операціями, які потрібно виконати для досягнення поставленої мети.

Таблиця 3.6 – Оптимізація проекту за ресурсами

№	Операція	Березень	Квітень	Травень
1	Планування технічного проекту	▼		
2	Аналіз функціональних вимог системи		▼	
3	Аналіз програмного забезпечення		▼	
4	Аналіз можливостей розробки		▼	
5	Розробка технічного рішення	▼		
6	Розробка концепції проектування			▼
7	Розробка і вибір варіантів концепції за вимогами замовника			▼
8	Розробка ескізного проекту	▼		
9	Розробка проектних рішень по системі та її частинам	▼		
10	Проведення мануального та автоматизованого тестування		▼	
11	Визначення тест кейсів	▼		
12	Планування тестування та перевірки системи			▼

4 РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНА ПЕРЕВІРКА

4.1 Реалізація модулю створення артефактного журналу подій

Первинні логи, або первинний журнал подій, представляють собою послідовність подій, пов'язаних з життєвим циклом певних процесів чи проекту. Кожна подія вказує на конкретну дію та описує, що саме сталось з процесом чи артефактом, як змінився його стан після виконання події та інші відомості. В цій роботі ми будемо використовувати спрощений журнал подій, тому він буде мати лише 4 базових параметра:

– timestamp: Часова мітка, що вказує коли сталася подія;

– event: Назва події, яка відбулася, наприклад, "Create Document" або "Update Document";

– artifact: Артефакт, з яким пов'язана подія, наприклад, "Document1";

– state_change: Зміна стану артефакту, що відбулася в результаті події, наприклад, "created", "modified" або "finalized".

У цьому лозі ми спостерігаємо послідовність подій, що відображають стадії життєвого циклу різних артефактів.

Первинні логи можуть бути використані для подальшого аналізу та моделювання життєвого циклу артефакту за допомогою методів process mining, включаючи побудову Petri net моделей та візуалізацію процесів управління артефактами.

```
/Users/dmytro.shuba/PycharmProjects/diploma/venv/bin/python
+---------------------+-----------------+------------+--------------------------------+
|      Timestamp      |      Event      |  Artifact  |          State Change          |
+---------------------+-----------------+------------+--------------------------------+
| 2023-05-01 09:00:00 | Receive Request | Order #123 |            Created             |
| 2023-05-02 10:15:30 |  Update Status  | Order #123 |  Status Updated to Processing  |
| 2023-05-03 14:30:45 |     Add Item    | Order #123 |        Item #456 Added         |
| 2023-05-04 16:45:15 | Update Quantity | Order #123 | Quantity Updated for Item #456 |
| 2023-05-05 11:20:10 |    Ship Order   | Order #123 |       Order #123 Shipped       |
| 2023-05-06 08:10:25 | Receive Payment | Order #123 |        Payment Received        |
| 2023-05-07 13:05:40 |  Update Status  | Order #123 |  Status Updated to Completed   |
| 2023-05-08 09:30:55 |     Add Item    | Order #789 |        Item #321 Added         |
| 2023-05-09 11:40:20 | Update Quantity | Order #789 | Quantity Updated for Item #321 |
| 2023-05-10 15:55:30 |   Cancel Order  | Order #789 |      Order #789 Cancelled      |
| 2023-05-11 14:20:15 |  Create Invoice | Order #123 |        Invoice Created         |
| 2023-05-12 16:30:05 | Receive Payment | Order #789 |        Payment Received        |
| 2023-05-13 10:25:50 |  Update Status  | Order #789 |  Status Updated to Completed   |
+---------------------+-----------------+------------+--------------------------------+
```

Рисунок 4.1 – Таблиця з подіями первинного логу

Для того, щоб можна було визначити життєвий цикл артефакту і в подальшому побудувати відповідну діаграму мереж Петрі, необхідно спочатку привести їх до правильного формату, а тобто сформувати артефактний журнал подій, що буде фокусуватись саме на подіях, що сталися з конкретними артефактами проекту та описувати, як змінювався стан цих артефактів.

Артефактний журнал подій дозволяє фіксувати кожну важливу подію, пов'язану з артефактом, включаючи його створення, зміни стану, видалення та інші події, які відбуваються протягом життєвого циклу артефакту. Цей журнал допомагає зрозуміти послідовність подій і відношення між ними, що дозволяє побудувати діаграму мереж Петрі для подальшого аналізу та візуалізації процесу управління артефактами.

Лістинг 4.1 – Трансформація артефактного журналу подій на базі первиного журналу подій

```
import csv

def transform_logs(raw_logs_path, artifact_logs_path):
    artifact_logs = []

    with open(raw_logs_path, 'r') as raw_file:
        csv_reader = csv.DictReader(raw_file)
        for row in csv_reader:
            timestamp = row['Timestamp']
            event = row['Event']
            artifact = row['Artifact']
            state_change = row['State Change']

            artifact_logs.append({'Timestamp': timestamp, 'Artifact':
artifact, 'Event': event, 'State Change': state_change})

    with open(artifact_logs_path, 'w', newline='') as artifact_file:
        fieldnames = ['Timestamp', 'Artifact', 'Event', 'State Change']
        csv_writer = csv.DictWriter(artifact_file, fieldnames=fieldnames)
        csv_writer.writeheader()
        csv_writer.writerows(artifact_logs)

    print("Artifact-Centric logs generated successfully!")


# Example usage
transform_logs('raw_logs.csv', 'artifact_logs.csv')
```

Формування артефактного журналу подій вимагає врахування всіх важливих подій, які стосуються артефакту, і фіксування відповідних змін стану. Це дозволяє збирати необхідну інформацію для подальшого аналізу та моделювання життєвого циклу артефакту, що є важливим етапом в процесному управлінні.

Отже, формування артефактного журналу подій є першим кроком у визначенні життєвого циклу артефакту та побудові відповідної діаграми мереж Петрі для аналізу та візуалізації процесу управління артефактами.

Результатом формування артефактного журналу подій зображений у вигляді таблиці на рисунку 4.2.

```
/Users/dmytro.shuba/PycharmProjects/diploma/venv/bin/python
+---------------------+------------+-----------------+--------------------------------+
|      Timestamp      |  Artifact  |      Event      |          State Change          |
+---------------------+------------+-----------------+--------------------------------+
| 2023-05-01 09:00:00 | Order #123 | Receive Request |            Created             |
| 2023-05-02 10:15:30 | Order #123 |  Update Status  |  Status Updated to Processing  |
| 2023-05-03 14:30:45 | Order #123 |     Add Item    |        Item #456 Added         |
| 2023-05-04 16:45:15 | Order #123 | Update Quantity | Quantity Updated for Item #456 |
| 2023-05-05 11:20:10 | Order #123 |    Ship Order   |       Order #123 Shipped       |
| 2023-05-06 08:10:25 | Order #123 | Receive Payment |        Payment Received        |
| 2023-05-07 13:05:40 | Order #123 |  Update Status  |  Status Updated to Completed   |
| 2023-05-08 09:30:55 | Order #789 |     Add Item    |        Item #321 Added         |
| 2023-05-09 11:40:20 | Order #789 | Update Quantity | Quantity Updated for Item #321 |
| 2023-05-10 15:55:30 | Order #789 |   Cancel Order  |      Order #789 Cancelled      |
| 2023-05-11 14:20:15 | Order #123 |  Create Invoice |        Invoice Created         |
| 2023-05-12 16:30:05 | Order #789 | Receive Payment |        Payment Received        |
| 2023-05-13 10:25:50 | Order #789 |  Update Status  |  Status Updated to Completed   |
+---------------------+------------+-----------------+--------------------------------+
```

Рисунок 4.2 – Артефактний журнал подій

Артефактний журнал подій є інструментом, що використовується в процесному управлінні для фіксації та аналізу подій, пов'язаних з артефактами проекту. Він містить записи про різні події, що відбуваються з артефактами, такі як створення, зміна стану, переміщення, видалення тощо.

4.2 Генерація мереж Петрі

Для створення мережі Петрі, нам необхідно зчитати артефактний журнал подій та використати параметри звідти для побудови станів (places), переходів (transitions) та зв’язків (arcs). Створення мереж Петрі наведено в лістингу 4.2.

Лістинг 4.2 – Генерація мереж петрі

```
import csv
import networkx as nx
import matplotlib.pyplot as plt

with open('artifact_logs.csv', 'r') as file:
    reader = csv.reader(file)
    next(reader)
    logs = list(reader)

places = set()
transitions = set()

for log in logs:
    timestamp, artifact, event, state_change = log
    transitions.add(event)
    places.add(artifact)

petri_net = {'places': places, 'transitions': transitions, 'arcs': []}

for log in logs:
    timestamp, artifact, event, state_change = log
    petri_net['arcs'].append((event, artifact, state_change))

print("Places:", petri_net['places'])
print("Transitions:", petri_net['transitions'])
print("Arcs:", petri_net['arcs'])
```

Отже, цей код зчитує дані з CSV-файлу, розбиває їх на окремі елементи (timestamp, artifact, event, state_change), формує множини унікальних значень для

місць і переходів, а також створює список зв'язків між переходами, артефактами та зміною стану.

Наступним кроком ми маємо візуально відобразити отриману мережу Петрі. Отже, цей код використовує бібліотеки networkx та matplotlib.pyplot для створення та візуалізації графу Петрі на основі отриманої структури даних. Вузли графу представляють місця і переходи, а ребра відображають зв'язки між ними. Код з їх використання для рендерінгу простої мережі Петрі є в лістингу 4.3.

Лістинг 4.3 – Спрощений рендеринг мережі Петрі

```
import networkx as nx
import matplotlib.pyplot as plt

places = petri_net['places']
transitions = petri_net['transitions']
arcs = {(source, target): label for source, target, label in
petri_net['arcs']}

petri_net = nx.DiGraph()

petri_net.add_nodes_from(places, node_color='lightblue')
petri_net.add_nodes_from(transitions, node_color='lightgreen')

petri_net.add_edges_from(arcs)

#node_positions = nx.spring_layout(petri_net)
node_positions = nx.kamada_kawai_layout(petri_net)
node_positions = {node: (x, y*1.5) for node, (x, y) in
node_positions.items()}

nx.draw_networkx(petri_net, pos=node_positions, with_labels=True,
node_size=1500, node_color='lightgray',
                edge_color='black', font_color='black',
font_weight='bold')

plt.title("Petri Net")
plt.axis('off')

plt.show()
```

Networkx – це бібліотека для роботи з графами і мережами. Вона надає широкий спектр функцій для створення, маніпулювання та аналізу графів. networkx підтримує різні типи графів, включаючи неорієнтовані, орієнтовані та зважені графи. З його допомогою можна додавати вузли, дуги, визначати атрибути вузлів та дуг, знаходити найкоротші шляхи, виконувати аналіз графів та багато іншого.

Matplotlib – це бібліотека для візуалізації даних, включаючи графіки, діаграми та інші типи візуалізацій. Вона надає потужні інструменти для створення візуальних елементів, налаштування їх вигляду та розміщення на графічній площині. matplotlib дозволяє створювати різноманітні типи графіків, включаючи лінійні графіки, стовпчикові діаграми, колові діаграми та багато інших.

У випадку використання разом networkx та matplotlib для візуалізації графів, networkx забезпечує функціональність для побудови графів, а matplotlib використовується для створення візуального представлення цих графів. За допомогою цих бібліотек ви можете створювати графічні представлення графів, налаштовувати їх вигляд та відображати їх у зручному форматі для аналізу та відображення результатів вашої роботи.

Існує кілька альтернативних бібліотек для візуалізації даних, які можуть бути використані замість matplotlib. Ось декілька популярних альтернатив: Seaborn (побудована на основі matplotlib) надає більш простий та зручний інтерфейс для створення красивих графіків та діаграм; Plotly дозволяє створювати інтерактивні графіки, діаграми та візуалізації для веб-середовища; Bokeh спрямована на створення високопродуктивних інтерактивних візуалізацій для веб-середовища;

ggplot є розширенням графічного мови R під назвою ggplot2 для Python. Він надає декларативний підхід до створення графіків, який базується на граматиці графіків. Результат рендерінгу простої мережі Петрі зображений на рисунку 4.3.

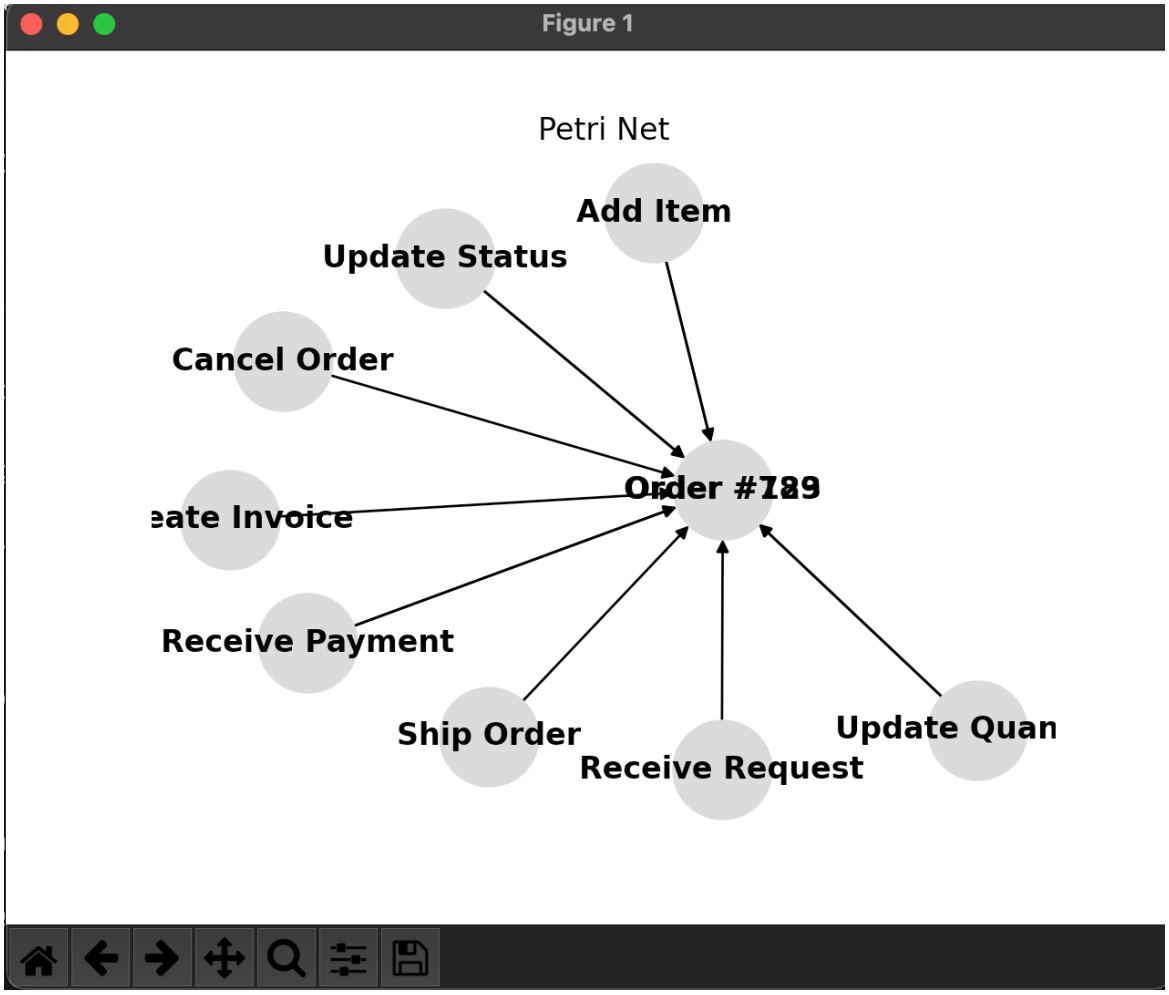

Рисунок 4.3 – Спрощена візуалізація мережі Петрі

4.3 Створення GSM моделі

Створення моделі GSM на основі мережі Петрі використовує підхід, в якому перехід в мережі Петрі відповідає захисту в моделі GSM, стан в мережі Петрі відповідає етапу в моделі GSM, а дуга в мережі Петрі відповідає майлстоуну в моделі GSM. У коді проекту реалізовано такий процес. Спочатку зчитуються дані з CSV-файлу, де записи містять інформацію про артефакти, події та зміни стану. Створюються множини для зберігання унікальних місць (етапів) та переходів (захистів). Проходяться всі записи журналу із CSV-файлу, в разульатті чого додаються множини етапів, множини захистів та створюється порожня графова структура GSM за допомогою nx.DiGraph().

Слідом додаються вузли (етапи захисти) до графа GSM з відповідними параметрами і додаються дуги (майлстоуни) між артефактами (етапами) і переходами (захистами) на основі записів з журналу. Встановлюються позиції вузлів для візуалізації графа GSM. Граф GSM візуалізується за допомогою nx.draw_networkx, що дає змогу побачити структуру моделі GSM з використанням кольорів та форм вузлів.

Таким чином, створення моделі GSM на основі мережі Петрі передбачає використання захистів як переходів, етапів як місць та як майлстоунів між артефактами та відповідними переходами. Цей підхід дозволяє візуалізувати модель GSM та розглядати залежності між артефактами, етапами, захистами та майлстоунами у вигляді графа.

Лістинг 4.4 – Генерація GSM моделі

```
for log in logs:
    timestamp, artifact, event, state_change = log
    transitions.add(event)
    places.add(artifact)
    if 'Guard:' in state_change:
        guard = state_change.split(':')[1].strip()
        guards.add(guard)
    elif 'Stage:' in state_change:
        stage = state_change.split(':')[1].strip()
        stages.add(stage)
    elif 'Milestone:' in state_change:
        milestone = state_change.split(':')[1].strip()
        milestones.add(milestone)
gsm = nx.DiGraph()
add_notes(gsm, places, transitions, guards, stages, milestones)
for log in logs:
    timestamp, artifact, event, state_change = log
    if 'Guard:' in state_change:
        guard = state_change.split(':')[1].strip()
        if guard in gsm: gsm.add_edge(artifact, guard, label=event)
    elif 'Stage:' in state_change:
        stage = state_change.split(':')[1].strip()
        if stage in gsm: gsm.add_edge(artifact, stage, label=event)
    elif 'Milestone:' in state_change:
        milestone = state_change.split(':')[1].strip()
        if milestone in gsm: gsm.add_edge(artifact, milestone,
label=event)
    else:
        gsm.add_edge(artifact, event, label=state_change)
node_positions = nx.spring_layout(gsm)
nx.draw_networkx(gsm, pos=node_positions, with_labels=True,
node_size=1500, node_color='lightgray', edge_color='black',
font_color='black', font_weight='bold')
node_shapes = nx.get_node_attributes(gsm, 'shape')
node_colors = nx.get_node_attributes(gsm, 'node_color')
for node in gsm.nodes: nx.draw_networkx_nodes(gsm, pos=node_positions,
nodelist=[node], node_shape=node_shapes.get(node, 'o'),
node_color=node_colors.get(node, 'lightblue'), node_size=1500)
edge_labels = nx.get_edge_attributes(gsm, 'label')
nx.draw_networkx_edge_labels(gsm, pos=node_positions,
edge_labels=edge_labels)
plt.title("Guard-Stage-Milestone (GSM) Model")
plt.axis('off')
plt.show()
```

Рисунок 4.2 – Рендерінг елементу життєвого циклу артефакту

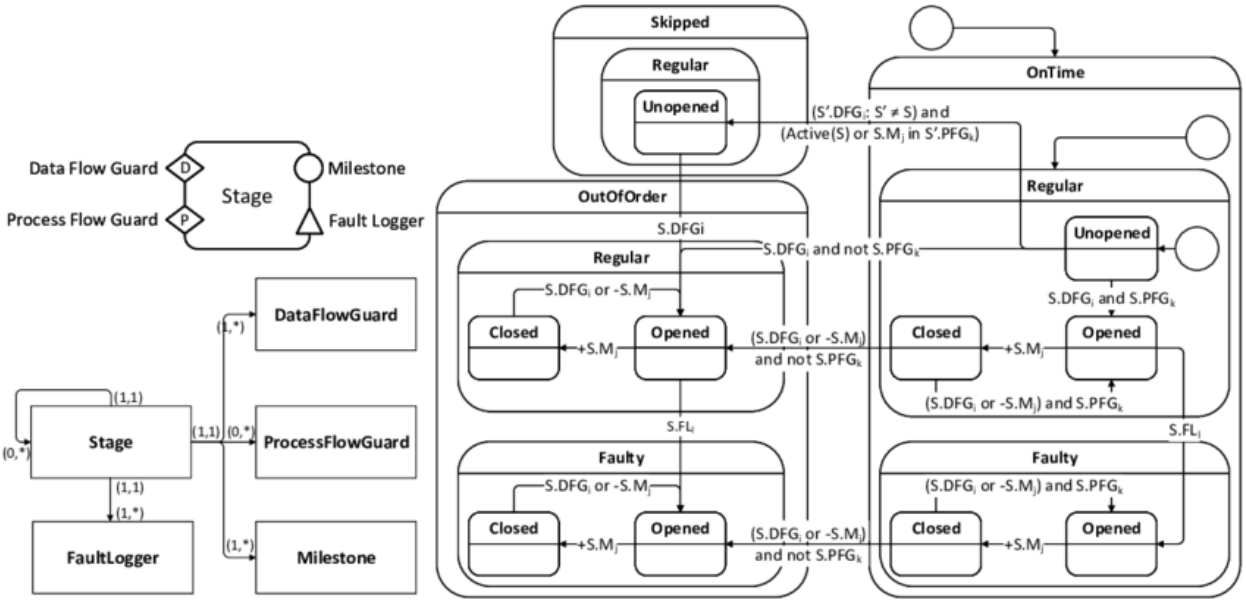

Рисунок 4.3 – Приклад рендерінгу повної GSM моделі

ВИСНОВКИ

Управління бізнес процесами та дослідження моделей життєвого циклу артефактів є надзвичайно важливими для ефективного управління процесами в організації. Наявність чітких моделей та стандартів управління процесами дозволяє підвищити продуктивність, зменшити витрати та збільшити якість продукції або послуг. Моделі життєвого циклу артефактів, які використовуються в управлінні бізнес процесами, дозволяють ефективно планувати та керувати процесами. Для досягнення максимальних результатів важливо використовувати правильні методи виявлення та аналізу артефактів, що дозволить забезпечити їх якість та вчасно виявляти помилки та недоліки.

Дослідження моделей життєвого циклу артефактів дозволяють не тільки ефективніше управляти бізнес процесами, а й покращувати самі моделі та стандарти. Аналіз даних та виявлення недоліків дозволяє вчасно реагувати на проблеми та вдосконалювати процеси, що забезпечує зростання ефективності та конкурентоспроможності компанії. Крім того, дослідження моделей життєвого циклу артефактів дозволяють виявляти та аналізувати ризики та можливості, що допомагає управляти компанією більш ефективно та прогнозувати її розвиток.

Загалом, використання моделей життєвого циклу є корисним інструментом для ефективного управління проектами та розробки артефактів. Однак, вибір конкретної моделі та її використання повинні враховувати особливості проекту, потреби команди та характеристики середовища.

Було виконано аналіз підходів до процесного управління, аналіз властивостей артефактів. Здіснена структуризація життєвого циклу артефактів з урахуванням того, що артефакти можуть використовуватись в різних процесах. Виконано та експериментально перевірено реалізацію артефактної моделі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Бортнік А. М. Процесне управління бізнесом: сутність і переваги впровадження / А. М. Бортнік // Науковий вісник Національного університету ДПС України (економіка, право). – 2013. – № 3 (62). – С. 30–36.

2. Nigam, A. and N.S. Caswell, Business artifacts: An approach to operational specification. IBM Syst. J., 2003. 42(3): p. 428-445.

3. Маклаков С.В. BPwin и Erwin : CASE-средства разработки информационных систем / С.В. Маклаков. – М.: Диалог-МИФИ, 1999. – 256 с.

4. Методичні вказівки щодо розробки та оформлення магістерської кваліфікаційної роботи за спеціальністю 122 Комп'ютерні науки (освітня програма «Управління проєктами в галузі інформаційних технологій» освітньо-кваліфікаційного рівня «магістр» / Упоряд.: Петров К.Е., Левикін В.М., Чалий С.Ф., Євланов М.В., Саєнко В.І., Міхнов Д.К., Міхнова А.В., Чала О.В. – Харків: ХНУРЕ, 2021. – 28 с

5. ДСТУ 8302:2015. Інформація та документація. Бібліографічні посилання. Загальні положення та правила складання. – Чинний від 04.03.2016. – Київ: ДП «УкрНДНЦ», 2016. – 20 с.

6. ДСТУ 3008:2015. Інформація та документація. Звіти у сфері науки і техніки. Структура і правила оформлювання. – Чинний від 22.06.2015. – Київ: ДП «УкрНДНЦ», 2016. – 31 с.

7. Nigam, A., Caswell, N.S.: Business artifacts: An approach to operational specification. IBM Systems Journal, 42(3), 428-445 (2003).

8. Ouyang, C., Dumas, M., Breutel, S. ter Hofstede, A.H.M.: Translating Standard Process Models to BPEL. In: CAiSE'06, 417–432 (2006).

9. Pesic, M., van der Aalst, W.: A Declarative Approach for Flexible Business

Processes Management. In: Business Process Management Workshops. Springer Berlin / Heidelberg, 169–180 (2006).

10. Rozinat, A., van der Aalst, W.M.P.: Decision Mining in ProM. In: S. Dustdar, J.L. Fiadeiro, and A. Sheth, editors, BPM 2006, 420-425. Springer-Verlag (2006).

11. Rozinat, A., Van der Aalst, W.M.P.: Conformance Checking of Processes Based on Monitoring Real Behavior. Information Systems 33, 64-95 (2008).

12. Silberschatz, A., Korth, H. F., Sudarshan, S.: Database System Concepts, 4th Edition. McGraw-Hill Book Company (2001).

13. Fahland, D., van der Aalst, W.M.P.: Repairing process models to reflect reality. In: Business Process Management 2012, volume 7481 of Lecture Notes in Computer Science, 229-245. Springer (2012).

14. Feldman, P., Miller, D.: Entity model clustering: Structuring a data model by abstraction. The Computer Journal, 29(4), 348–360 (1986).

15. G¨unther, C., van der Aalst, W.: Fuzzy Mining Adaptive Process Simplification Based on Multi-perspective Metrics. In: Business Process Management, vol. 4714, Springer Berlin / Heidelberg, 328-343, (2007).

16. G¨unther, C., van der Aalst, W.: Mining Activity Clusters from Low-Level Event Logs, BE

17. Phillips, J. IT Project Management: On Track from Start to Finish. New York: McGraw-Hill Education, 2015. 15-31 p.

18. Фунтов, В. Н. Управління проектами в галузі інформаційних технологій. Лори, 2018. 3–10 с.

19. ДСТУ 3008:2015. Інформація та документація. Звіти у сфері науки і техніки. Структура і правила оформлювання. – Чинний від 22.06.2015. – Київ: ДП «УкрНДНЦ», 2016. – 31 с.

20. Fahland, D., De Leoni, M., van der Aalst, W. M. P.: Many-to-many: Some observations on interactions in artifact choreographies., 9–15. CEUR-WS.org (2011).