МАГІСТЕРСЬКА РОБОТА

МР. ІПм - 30.00.00.000 ПЗ

Група ІПм-22-2

Гнатків Тарас

2024

Івано-Франківський національний технічний університет нафти і газу

Інститут інформаційних технологій

Кафедра інженерії програмного забезпечення

Гнатків Тарас Олегович

(прізвище, ім'я, по батькові)

УДК 004.942

(індекс)

МАГІСТЕРСЬКА РОБОТА

Інноваційні моделі, методи та технології управління розробкою програмного забезпечення

(назва роботи)

Інженерія програмного забезпечення

(назва освітньої програми)

121 - Інженерія програмного забезпечення

(шифр і назва спеціальності)

Гнатків Т.О.

(підпис, ініціали та прізвище здобувача освітнього ступеня)

Науковий керівник Бандура Вікторія Валеріївна, к.т.н., доцент

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

Допущено до захисту

В.о. завідувача кафедри

доц. Бандура В.В.

(посада) (підпис) (дата) (ініціали та прізвище)

Рецензент

доц.

(посада) (підпис) (дата) (ініціали та прізвище)

Робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Івано-Франківськ – 2024

Івано-Франківський національний технічний університет нафти і газу

Інститут інформаційних технологій

Кафедра інженерії програмного забезпечення

Освітній рівень магістр

Спеціальність 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:
В.о. зав. кафедрою **ІПЗ**
доц. В.В. Бандура
“ 04 ” вересня 2023 р.

ЗАВДАННЯ

НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Гнатківу Тарасу Олеговичу

(прізвище, ім’я, по-батькові)

1. Тема магістерської роботи “ Інноваційні моделі, методи та технології управління розробкою програмного забезпечення ”

керівник проекту (роботи) Бандура Вікторія Валеріївна, к.т.н., доцент

затверджені наказом закладу вищої освіти від “ 18 ” грудня 2023 р. № 738/7

2. Строк подання студентом проекту (роботи) 15 січня 2024 р.

3. Вихідні дані до проекту (роботи) Теоретичні концепції та формальні моделі побудови та функціонування інформаційних технологій управління проєктами ПЗ

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

1. Аналіз предметної області управління проєктами розробки програмного забезпечення
2. Інноваційні моделі управління командами проєктів розробки ПЗ
3. Інноваційні моделі та методи управління командами проєктів розробки
4. Інформаційна технологія інноваційного управління командою

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

1. Agile як загальний термін для багатьох підходів (рис. 1.2)
2. Основні особливості методології Scrum (рис. 1.4)
3. Основні особливості методології Kanban (рис. 1.5)
4. Основні особливості методології Lean (рис. 1.6)
5. Модель розвитку команди «Сходи Такмана» (рис. 1.9)

6. Консультанти розділів проекту (роботи)

Розділ	Консультант	Підпис, дата
Нормоконтроль	доц., к.т.н. Вовк Р.Б.	
Перевірка на плагіат	доц., к.т.н. Вовк Р.Б.	

7. Дата видачі завдання 04 вересня 2023 р.

Керівник _____
(підпис)

Завдання прийняв до виконання_____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назви етапів магістерської роботи	Строк виконання етапів роботи	Примітка
1	Підбір і вивчення літератури по темі магістерської роботи	01.10.2023	виконано
2	Аналіз концепцій та алгоритмів предметної області	25.10.2023	виконано
3	Інноваційні моделі управління командами проєктів розробки ПЗ	10.11.2023	виконано
4	Інноваційні моделі та методи управління командами проєктів розробки	22.11.2023	виконано
5	Інформаційна технологія інноваційного управління командою	01.12.2023	виконано
6	Реалізація функціональності запропонованої інформаційної технології	15.12.2023	виконано
7	Затвердження пояснювальної записки роботи завідувачем кафедри	15.01.2024	виконано

Студент – магістр _____
(підпис)

Керівник роботи _____
(підпис)

АНОТАЦІЯ

Магістерська робота: 87 с., 25 рис., 2 табл., 50 джерел.

Тема: Інноваційні моделі, методи та технології управління розробкою програмного забезпечення

Об'єкт дослідження: процеси управління командами проєктів розробки програмного забезпечення.

Мета роботи: підвищення ефективності управління командою інноваційних проєктів розробки програмного забезпечення за рахунок вдосконалення моделей, методів та засобів управління з використанням концепції MVP.

Предмет дослідження: моделі, методи та інноваційні засоби управління командами проектів розробки програмного забезпечення.

Результати дослідження:

В роботі наведена концептуальна модель управління командами проектів розробки програмного забезпечення, яка базується на технологіях розробки продукту проєкту: дизайн-мислення та технології MVP, що дає змогу сформувати гнучку команду управління проєктами.

Висновок

Вдосконалено метод формування команди проєкту розробки програмного забезпечення на основі використання генетичного алгоритму для створення мінімально-необхідної команди, який на відміну від існуючих враховує набір показників, що відображає поточний стан команди та проєкту.

ІННОВАЦІЙНИЙ ПРОЄКТ, ПРОЦЕС УПРАВЛІННЯ, КОМПЕТЕНЦІЇ, СТЕЙКХОЛДЕР, ШНУЧКІ МЕТОДОЛОГІЇ, УПРАВЛІННЯ КОМАНДОЮ, ЖИТТЄВИЙ ЦИКЛ ПРОЄКТУ

ABSTRACT

Master Thesis: 87 pp., 25 fig., 2 tab., 50 sources.

Thesis Subject: Innovative models, methods and technologies of software development management

Object of research: processes of software development project team management.

Research goal: increasing the effectiveness of the team management of innovative software development projects due to the improvement of models, methods and management tools using the MVP concept.

Subject of research: models, methods and innovative means of managing teams of software development projects.

The results:

The paper presents a conceptual model of software development project team management, which is based on project product development technologies: design thinking and MVP technology, which enables the formation of a flexible project management team.

Conclusion

The method of forming a software development project team has been improved based on the use of a genetic algorithm to create the minimum necessary team, which, unlike the existing ones, takes into account a set of indicators that reflects the current state of the team and the project.

INNOVATION PROJECT, MANAGEMENT PROCESS, COMPETENCIES, STAKEHOLDER, INNOVATIVE METHODOLOGIES, TEAM MANAGEMENT, PROJECT LIFE CYCLE

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ 9
ВСТУП 10

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ УПРАВЛІННЯ ПРОЄКТАМИ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 13
1.1. Сучасні моделі, методи, стандарти управління ІТ-проєктами 13
1.2. Особливості гнучких методологій управління проєктами 21
1.3. Особливості управління командами проєктів розробки ПЗ 25
1.4 Аналіз інформаційних технологій управління проєктами розробки ПЗ 32
1.5 Висновки до розділу 36

РОЗДІЛ 2. ІННОВАЦІЙНІ МОДЕЛІ УПРАВЛІННЯ КОМАНДАМИ ПРОЄКТІВ РОЗРОБКИ ПЗ 37
2.1. Дослідження та опис технологій управління проєктами 37
2.2 Концептуальна модель інноваційного управління командою ІТ-проєкту 44
2.3 Семіотична модель управління командою ІТ-проєкту 49
2.4 Висновки до розділу 56

РОЗДІЛ 3. ІННОВАЦІЙНІ МОДЕЛІ ТА МЕТОДИ УПРАВЛІННЯ КОМАНДАМИ ПРОЄКТІВ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 58
3.1. Метод формування інноваційної команди 58
3.2 Використання генетичного алгоритму для оптимізації складу команди проєкту розробки ПЗ 64
3.3 Практичний приклад формування команди проєкту розробки ПЗ 71
3.4 Інформаційна технологія інноваційного управління командою розробки ПЗ 77

3.5 Висновки до розділу 81

ВИСНОВКИ 82
ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ..... 83

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

МЖП - мінімально життєздатний продукт

MVP - minimum viable product

PMI - Project Management Institute

IEEE - Institute of Electrical and Electronics Engineers

ПП - програмні проекти

LOE - level-of-effort

ЖЦП - життєвий цикл проекту

ITSM - IT-Service Management

ТСПП - технологія створення програмного продукту

PSP - Personal software process

FIRO-B - Fundamental Interpersonal Relations Orientation Behavior

ККІТП - координатор команди ІТ-проєкту

T2M - time-to-market

ПРП - продукт проекту

HCD - Human-centered design

PoC - Proof of concept

WBS - work breakdown structure

OBS - organizational breakdown structure

RBS - resource breakdown structure

SCi - complete situation

SRi - Current situation

МЖК - мінімально життєздатна команда

ДМ - дизайн-мислення

MVT - minimum viable team

ВСТУП

Актуальність теми дослідження.

Процеси формування та управління командою будь-якого проєкту є слабко структурованими та невизначеними. Обрання критеріїв та стратегії формування команди залежить від значної кількості факторів: від цілей та задач проєкту, від бачення його учасників, взаємодії стейкхолдерів, галузі застосування продукту, від особистісних характеристик замовника, керівника команди проєкту та інших ключових учасників.

ІТ-галузь, як будь-яка інша, має свою специфіку. З одного боку, існують розроблені та протестовані методики створення та управління ІТ-проєктами: від водоспадної до гнучкої. З іншого боку, перелік робіт ІТ-проєктів є, зазвичай, жорстко формалізованим: збір вимог, формування ТЗ, розробка архітектури, кодування, тестування, впровадження продукту у замовника. Цей перелік робіт формує перелік функціональних вимог та обмежень до ролей майбутньої команди проєкту, яка зазвичай формується з персоналу ІТ-компанії та управляється за прийнятою в компанії методологією.

Критерії оцінки успішності проєкту неухильно зміщуються в бік забезпечення задоволеності його головних стейкхолдерів. Для цього розробляють нові, або застосовують раніше не використовувані засоби. Серед них можемо відзначити методику дизайн-мислення та концепцію МЖП. ІТ-проєкт - це завжди командна робота, в якій присутні дві ролі: «програміст» і «споживач». «Програміст» з використанням інтелектуального капіталу та когнітивних процесів створює програмний продукт, при цьому використовує інноваційні рішення, та передає готовий продукт «споживачеві».

Для успішного завершення таких проєктів з високою долею інновацій, необхідна команда, що самоорганізується та самокерується. Така команда сама приймає рішення щодо структури, методології та форми реалізації

проєкту. У традиційних командах та великих ІТ-компаніях фазі реалізації передує планування (навіть, якщо мова йде про гнучкі технології розробки). Чим більше приймається рішень щодо самої команди, концепту майбутньої розробки, тим більш така команда обмежена, тим менш вона керує власним простором. Це обмежує можливість самоорганізації такої команди та перешкоджає управлінню проєктами з високою часткою інновацій.

У зв'язку з цим виникає важлива науково-прикладна задача щодо ефективного управління командами ІТ-проєктів з високою часткою інновацій в переліку їх проєктних рішень за рахунок розробки нових моделей, методів та інформаційних засобів креативного та гнучкого управління такими проектами.

Мета і завдання дослідження.

Метою магістерської роботи є підвищення ефективності управління командою інноваційних проєктів розробки програмного забезпечення за рахунок вдосконалення моделей, методів та засобів управління з використанням техніки дизайн-мислення та концепції MVP.

Для досягнення вказаної мети в роботі виділено наступні **задачі дослідження:**

- аналіз предметної області та особливостей інноваційних проєктів та особливостей управління ними, огляд сучасних моделей, методів та інформаційних засобів управління командами;

- вдосконалення концептуальної моделі інноваційного управління командою проєкту розробки ПЗ;

- розробка семіотичної моделі управління командою ІТ-проєкту з інноваційною складовою;

- удосконалення методу формування команди ІТ-проєкту з інноваційною складовою;

- представлення інформаційної технології інноваційного управління командою проєкту розробки ПЗ.

Об’єкт дослідження - процеси управління командами проєктів розробки програмного забезпечення.

Предмет дослідження - моделі, методи та інноваційні засоби управління командами проектів розробки програмного забезпечення.

Методи дослідження. Науковим і теоретичним підґрунтям роботи стали наукові основи управління ІТ-проєктами, фахова література, наукові праці вчених, практики управління командами. В роботі було використано: системний підхід, порівняльний аналіз, методи управління проєктами, оптимізаційні методи, генетичний алгоритм, ситуаційне та семіотичне моделювання.

Наукова новизна одержаних результатів полягає в розробці концептуальної моделі управління командами проектів розробки програмного забезпечення, яка базується на технологіях розробки продукту проєкту: дизайн-мислення та технології MVP, що дає змогу сформувати гнучку, мінімально-життєздатну команду управління проєктами.

Практичне значення одержаних результатів полягає в дослідженні та вдосконаленні методу формування команди ІТ-проєкту на основі використання генетичного алгоритму для створення мінімально-необхідної команди, який на відміну від існуючих враховує не єдиний інтегральний показник компетентностей команди проєкту, а набір показників, що відображає поточний стан команди та проєкту.

Структура магістерської роботи.

Робота включає вступ, перелік умовних позначень, чотири розділи, та висновки. Обсяг магістерської роботи - 87 сторінок, і містить 25 рисунків, 2 таблиці та 50 використаних джерел.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ УПРАВЛІННЯ ПРОЄКТАМИ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1.1. Сучасні моделі, методи, стандарти управління ІТ-проєктами

1.1.1 Особливості сучасних ІТ-проєктів

Пошук ефективних механізмів управління системою зазвичай треба розпочинати з виокремлення та характеристики предмету дослідження. В даному випадку мова йде про сутність та характеристики ІТ-проєктів. Аналізуючи доступні до розгляду наукові та професійні джерела з управління ІТ-проєктами було виявлено що найменш дві тенденції.

Перша — це випадок, коли ІТ-проєкт розглядається, як комплекс заходів в середині НЕ ІТ-компанії, яка бажає автоматизувати власні бізнес-процес, зробити їх більш ефективними, розробивши та впровадивши в себе самотужки нові ІТ. Такі ІТ-проєкти можуть бути реалізовані у будь-якій галузі економіки, наприклад в енергетичній. «Практично кожна організація в процесі своєї діяльності стикається з тим чи іншим ІТ-проєктом, причому ІТ-проєкт в організації повинен розглядатися як частина великої системи».

Друга — це застосування для виокремлення особливостей та характеристик ІТ-проєкту критеріїв, що застосовують до звичайних інвестиційних проєктів, наприклад, проєктів будівництва. Це потребує, на нашу думку, уточнити визначення ІТ-проєкту за рахунок його значущих особливостей. Тому далі представлено результати проведеного аналізу.

Існує значна група робіт, в яких визначення ІТ-проєкту сформовано за класичним підходом на основі PMBoK, без жодного врахування особливостей галузі. Проєкт — «це тимчасовий захід, спрямований на розробку унікального продукту, що має чітко визначений термін виконання, обмеження за ресурсами, свої критерії якості і поняття про успішне завершення. Далі, зазначається, що «відповідно до цільової спрямованості, ІТ-проєкт — це проєкт, в рамки якого входять роботи, пов'язані з

інформаційними технологіями, які в свою чергу спрямовані на створення, розвиток і підтримку інформаційних систем». Такі визначення не досить повні, як для виокремлення ІТ-проєктів із загального пулу інших проєктів, так і для вибору або формування засобів управління, що враховуватимуть ІТ-особливості, задля забезпечення успішного завершення ІТ-проєктів.

І навіть доповнення, що подано в [5] у вигляді особливостей ІТ-проєктів, не дають чіткого уявлення про їхню сутність - «нестандартний життєвий цикл, який може включати в себе також тестовий, гарантійний та післягарантійний етапи розробки». Тут слід зауважити, що гарантійне або постгарантійне обслуговування не можуть бути етапами розробки. Крім того, в роботі приймемо, що ЖЦП завершується після передачі його замовнику та введення до експлуатації. За необхідності будь-якого супроводу, за узгодженням сторін, укладається окремий договір, та засоби, що будуть розроблені для управління ІТ- проєктом не розповсюджуватимуться на гарантійне та постгарантійне обслуговування. Однією з характеристик ІТ-проєкту в [5] зазначають «... необхідність чіткого визначення, вже на етапі ініціації, вимог до ІТ-проєктів, незважаючи на рухливість і неоднозначність.», але, в сучасних проєктах вимоги можуть формуватися аж до його завершення.

Класичний підручник з управління ІТ-проєктами представляє визначення таким чином: «Проєкт - це скінчений в часі захід, призначений для створення унікальних продуктів або послуг. Унікальність продуктів або послуг проєкту обумовлює необхідність послідовного уточнення їхніх характеристик протягом виконання проєкту». Аналогічно, у фундаментальному виданні [8] визначення ІТ-проєкту подано у досить широкому та загальному сенсі. «Проєкт — це унікальна, орієнтована на досягнення мети, тимчасова та обмежена умовами дія».

Далі автори подають розширення значень проєкту: « ... це велика або важлива дія (послідовність дій)», як зазначають самі автори, таки визначення

є декілька примітивним, та надалі використовують визначення пропоновані інститутами PMI та IEEE.

Наступною характеристикою, яка пов’язана з попередньою, є критичність комунікацій між замовниками та розробниками ІТ-проєкту. З огляду на складність уточнення вимог, важливо підтримувати із замовником відкриті та конструктивні комунікації.

І ще однією характеристикою ІТ-проєкту зазначається постійне коригування припущень, за якими буде виконуватися проєкт. Наприклад, можуть змінюватися компроміси щодо бюджету, графіку та функціонального змісту проєкту.

В [10] «... успішні ІТ-проєкти повинні мати добре позначену основну контрольну точку в здійсненні проєкту, після досягнення якої відбувається відчутний перехід від стану досліджень до стану виробництва». Тут не дається визначення проєкту в явному вигляді. Замість того в тексті присутні визначення: «Проєктування», «Проєктної моделі» та «Проміжної контрольної точки», які у сукупності окреслюють проєктний простір.

В [13] зазначено, що «.. проєкти розробки програмного забезпечення або ІТ- проєкти залишаються, в більшості випадків, ЗАГАДКАМИ - важко передбачуваними, важко реалізованими і важкокерованими». Тому далі є доцільним розглянути безпосередньо стандарти з досліджуваної предметної області, а саме, згаданий вище стандарт Програмної інженерії SWEBOK V3, та розширення PMBoK для програмних проєктів.

Так, згідно [15], « ... проєкт — це тимчасова робота (endeavor undertaken), що робиться для створення унікального продукту, послуги або результату». Крім розробки нових продуктів, вони модифікують існуючі, розширюють їх можливості, інтегрують нововведення до існуючої програмної інфраструктури.

Роботи з програмування можуть позиціонуватися як дії рівня зусиль (level - of-effort, або LOE), наприклад, задоволення запитів на обслуговування, забезпечення операційної підтримки, тощо. Але, якщо вони

будуть обмеженими у часі для забезпечення результатів, то їх можна вважати ІТ-проєктами.

Комунікації всередині команди проєкту розробки програмного забезпечення та комунікації зі стейкхолдерами характеризуються як такі, в яких «.. не вистачає ясності та однозначності», це характеризує проєкт як систему, зі складністю взаємодії з її частинами.

Таким чином, програмні проєкти збільшують причини появи ризиків та невизначеності за рахунок інновацій, нематеріальності продукту, неузгодженості потреб стейкхолдерів та чіткого бачення продукту [2].

Аналіз сучасного стану ринку ІТ-проєктів виявив новий акцент проєктної діяльності в ІТ-галузі. Це так звані ІТ-сервіси.

Компанії, для досягнення своїх бізнес-цілей у більшості випадків застосовують ІТ у вигляді якісних ІТ-послуг, які відповідають цілям бізнесу, вимогам і очікуванням замовника, при цьому останнім часом все більша увага приділяється не розробці ІТ-рішень (наприклад, бізнес-додатків), а управління послугами з їх супроводу, що гарантує високу доступність рішення для кінцевих користувачів [16]. При цьому в життєвому циклі ІТ-рішень на їх експлуатацію доводиться 70-80% часу і фінансових коштів і лише 20-30% часу і коштів витрачається на розробку (придбання) та впровадження продукту.

Відзначимо, що сьогодні керівники багатьох компаній незадоволені якістю ІТ-послуг, що надаються їхніми власними ІТ-підрозділами. Для цього є багато причин. Дуже часто ІТ-проєкти не завершуються у запланований термін та бюджет [16].

Далі представлено аналіз визначень ІТ-проєктів за різними джерелами. Загальний перелік визначень ІТ-проєкту не матиме сенсу, його необхідно згрупувати за певними критеріями, до окремих груп яких можна буде застосовувати схожі засоби управління.

Так у якості критерія для виділення типів проєктів використовується кількісний склад команди. Загалом, за цим критерієм представлено три типи проєктів.

1. **«Супермен-одинак»**. Проєкт простий, бере участь одна людина, яка робить все сама, від проведення ринкових досліджень до написання коду. Використовує власні джерела фінансування.

2. **Невелика команда**, що працює за контрактом. IT-компанія складається з п'яти - десяти програмістів і керівника, найнята замовником для створення IT- продукту. Працюють за контрактом, по завершенні якого усі зв'язки обриваються.

3. **Численна команда** штатних співробітників, кількістю зі 100 чоловік, працюють в корпорації по найму. IT-продуктом може бути: призначений для продажу (так званий коробковий) продукт; корпоративна система управління; система автоматизації бізнесу, тощо.

Ці різновиди представлені з метою виокремлення важливих відмінностей в підходах до управління IT-проєктом.

Наступним кроком аналізу проблеми стало пошук та дослідження засобів управління IT-проєктами, що мають певні особливості. Саме для таких проєктів застосовують «правило хреста»: якісно, швидко, дешево, готово. Це правило є адаптацією чи розвитком відомого проєктного трикутника: якість (обсяг робіт), час, вартість (рис. 1.1).

Рис. 1.1. Проєктний трикутник

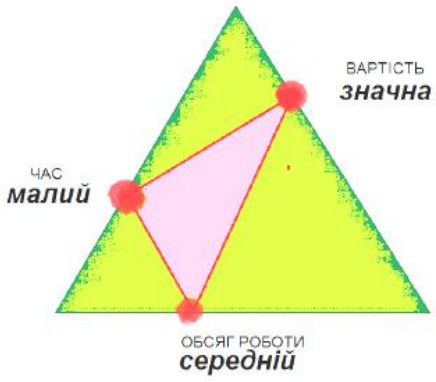


Рис. 1.2. Визначення якості робіт

На перетині прямих, які відображають кількість кожного із зазначених параметрів, формують фігуру, площа якої може бути інтерпретована, як якість робіт, що виконується за обраними обмеженнями (рис.1.2). Можна вважати, що «правило хреста» модернізувало поняття «якості проєкту» із загально управлінської площини в поняття «готовності ІТ-продукту», тобто працездатності програмного забезпечення.

«Правило хреста» можна інтерпретувати наступним чином: «Обирай будь-яких три пункти, але четвертий буде не під силу». Проєкт може бути одночасно якісним, дешевим, швидко виконуватися, але його ніколи не буде завершено. Або, навпаки, проєкт буде завершений швидко та дешево, але якісним зробити його не вийде.

«Правило хреста» може бути прийнятим як компоненти загальної системи управління ІТ-проєктами з точки зору можливості їх спостерігати та керувати ними. Тобто компонентам системи управління треба задати наступні властивості: деякі компоненти можуть бути виявлені (бо вони мають суттєвий вплив на успішне завершення ІТ-проєкту), деякі з них ми можемо спостерігати (у загальному випадку не всі з виявлених), деякими з них ми можемо керувати. Визначення цих компонентів та їх властивостей завершується формуванням стратегії управління кожним окремо взятим проєктом. Тому далі в дослідженні було проаналізована сутність управління ІТ-проєктами.

Найбільш вагомим, з точки зору компетентності визначень стосовно управління саме ІТ-проєктами, вважаємо «Розширення до PMBoK при створенні програмного забезпечення» (Software Extension to the PMBoK). Там зазначено, що «... управління проєктами — це застосування знань, навичок, інструментів та прийомів для проєктної діяльності для задоволення вимог проєкту». «Управління проєктами здійснюється шляхом використання логічно згрупованих процесів управління проєктами, що складаються у групи: ініціації, планування, виконання, моніторингу та контролю, і закриття». Згадане розширення згідно специфіки ІТ-проєктів формує

особливості управління які загалом сформовані через обмеження. Вони звичайно формуються ТСПП, і згруповані через технологічні фактори, які ці обмеження продукують.

До технологічних факторів, які можуть накладати обмеження на програмні проєкти та програмні продукти, відносять: стан апаратних та програмних технологій; апаратні платформи, програмні платформи, операційні системи та протоколи зв'язку; цілісність, обмеження та протоколи архітектури ІТ та інші.

До нетехнологічних факторів, що можуть накладати обмеження на проєкти програмного забезпечення, відносять: відповідність безпеці, надійності, доступності, масштабованості, продуктивності, тестування, забезпечення інформації, локалізації, ремонтопридатності, підтримки, правил, політики клієнтів, підтримки інфраструктури, доступність та вміння членів команди, середовище та методи розробки програмного забезпечення, а також організаційна зрілість та можливості.

Окремо робиться наголос на складність управління ІТ-проєктами і формують перелік факторів, що ускладнюють управління ІТ-проєктами. В рамках дослідження були виокремлені ти з них, наявність яких вимагатимє від команди проєкту додаткових зусиль та, відповідно, додаткових заходів з управління проєктом. Серед них:

- нематеріальність та податливість (гнучкість) продукту проєкту;
- розробка ПЗ часто характеризується як процес навчання команди проєкту, в процесі якого здобуваються знання;
- нелінійне масштабування ресурсів ІТ-проєкту, виміру проєкту і продукту, вхідна невизначеність в проєкті і в змісті його продукту;
- вимоги до ПЗ часто змінюються в процесі його розробки, під час накопичення знань та масштабування проєкту і продукту;
- інтелектуальний капітал команди ІТ-проєкту, яка займається розробкою ПЗ, є основним капіталом для програмних проєктів та ІТ-

компаній, що займаються розробкою ПЗ, оскільки ПЗ є прямим продуктом когнітивних процесів людини;

- в комунікаціях та координації всередині команд розробників ПЗ і з зацікавленими сторонами проєкту часто не вистачає ясності;
- багато інструментів та методів, що застосовують в розробці ПЗ, призначені для організації та управління проєктною комунікацією та координацією, які не є складовими ТСПП;
- створення ПЗ вимагає інноваційних підходів при вирішенні проблем. Більшість ПП надають унікальні продукти і вони більше схожі на проєкти досліджень і новітніх розробок, ніж на будівельні або виробничі проєкти;
- зазначені вище інновації, нематеріальність продукту, стейкхолдери, які не можуть чітко сформулювати або узгодити вимоги, які повинні бути задоволені за допомогою програмного продукту - все це формує додаткові ризики та невизначеності, які у свою чергу напряму впливають на команду ІТ-проєкту;
- первісне планування і оцінка ПП є складним завданням, тому що ці дії залежать від вимог, які часто є неточними або є частиною історичних даних, які часто відсутні або непридатні;
- точне оцінювання проєктних показників також є складним завданням, оскільки ефективність і результативність розробників дуже різняться;
- об'єктивна кількісна оцінка і вимір якості ПЗ складні через нематеріальну природу ПЗ;
- розробники ПЗ використовують процеси, методи і інструменти, які постійно розвиваються і часто оновлюються;
- технології, платформи, ПЗ інфраструктури часто змінюються або оновлюються, що може вимагати внесення змін до ПЗ, що розробляється.

Таким чином, можна зазначити, що нематеріальна природа продукту ІТ- проєкту створює додаткові проблеми в управлінні, при вимірюванні

поточного стану продукту, що, в свою чергу, ускладнює моніторинг і контроль впродовж ЖЦП. Традиційні підходи управління проєктами, такі, як структурна декомпозиція робіт, календарні розклади, сітьові діаграми, звіти про прибутки та збитки, погано адаптуються до потреб ПП, в яких, впродовж ЖЦП необхідна постійна демонстрація частково завершеного продукту проєкту. ПЗ є прямим продуктом когнітивних процесів людей, залучених в інтелектуальну, інноваційну командну роботу.

Додатковою складністю ІТ-проєктів є зміна у часі їх структури та зв'язків між зазначеними абстракціями. І усім цим необхідно управляти для забезпечення успішного завершення ІТ-проєкту. Класичні засоби управління проєктами в таких умовах або мало ефективні, або ж неефективні цілком. Загальновідоме співвідношення розпочатих та успішно завершених проєктів. Зазвичай, воно знаходиться в інтервалі 10%-30%. І, як було сказано вище, причиною провалів є неадекватні засоби управління.

Для ІТ-галузі виділяють чотири варіації управління за критеріями жорсткості та точності: 1) як вийде, 2) жорсткий водоспад, 3) гнучке управління, 4) метод частих ітерацій. Для зазначеної вище групи проєктів, в яких є задіяним інтелектуальний капітал команди для формування інноваційних проєктних рішень, перша варіація — «як вийде» буде не бажаною, друга взагалі не працюватиме, тому розглянемо засоби, що забезпечать гнучке управління або метод частих ітерацій.

Враховуючи це, PMI у 2017 створив Agile practice guide як додаток до шостого PMBoK. Автори відзначають, що гнучкі підходи і гнучкі методи « ... це загальні терміни, що охоплюють безліч структур і методів».

1.2. Особливості гнучких методологій управління проєктами

Agile - це загальний термін, що стосується будь-якого підходу, техніки, структури, методу або практики, які відповідають цінностям і принципам

Agile Manifesto та принципам ощадливого виробництва. Схематично це показано на рис. 1.3.

Під бережливим виробництвом автори мають на увазі: «акцент на цінності», «невеликі партії» та «усунення відходів». Agile — це і підхід, і метод, і практика, і техніка, і фреймворк. Може застосовуватися будь-який термін в залежності від конкретної ситуації та конкретного проєкту.

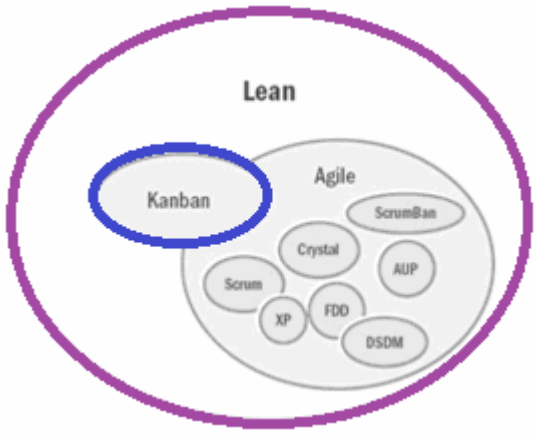


Рис. 1.3. Agile як загальний термін для багатьох підходів

Crystal - сімейство методологій розробки ПЗ із загальним генетичним кодом, що включає: часту доставку, особисті комунікації та вдосконалення через рефлексію.

Scrum - це гнучка методологія Agile, філософія розробки ПЗ, сукупність методів розробки і управління, гармонічно поєднаних між собою, в основі яких лежать принципи виробничої ітеративності, що в свою чергу, дозволяють замовнику частіше отримувати певний робочий функціонал в рамках розробки проєкту.

Підсумовуючи, можна сказати, що Scrum - це набір принципів, на яких будується процес розробки ПЗ, який дозволяє в короткий проміжок часу (спрінт), надавати замовнику працююче ПЗ з доданими можливостями, для яких був визначений найбільший пріоритет. Потрібний функціонал для реалізації в черговому спринті визначається до його початку на етапі планування і не може змінюватися на протязі всього спрінта. При цьому

жорстко фіксований невеликий термін спрінта додає процесу розробки гнучкості. Так, у [31] подана графоаналітична модель ситуаційного управління IT-проектами, які застосовують для розробки технологію Scrum. Особливості Scrum згруповані на рис. 1.4.

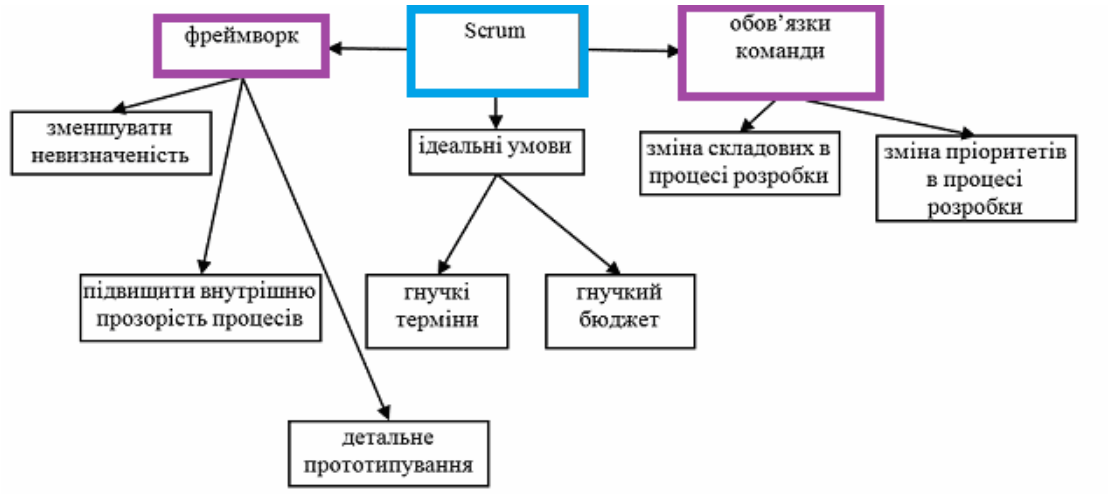


Рис. 1.4. Основні особливості методології Scrum

Методологія Kanban запроваджує систему «ощадливого виробництва», та провадить ідею створення єдиного потоку (процесу) без простоїв від незавершених або неузгоджених завдань. Це дозволило уникнути незапланованих витрат, покращити якість продукту, знизити вартість та скоротити терміни виконання. Kanban добре працює на стартапах, тобто там, де не має чіткого плану, але активно працюють над розробкою.

Формалізація процесів в спринті відбуваються наступним чином: взявшись до роботи, програміст перетягує завдання з колонки «To do» в «In progress». В цей момент він має лише одну задачу, концентрується на її ефективному вирішенні, так як паралельне виконання завдань не допускається. Після закінчення цієї роботи, він перетягує її в колонку «Ready for deploy».

Kanban-дошка - обов'язковий елемент гнучкої технології для відстеження оперативної ситуації всередині IT-компанії, яка застосовує ітеративну модель. Кожен член команди отримує доступ до дошки у будь-

який час і бачить, на якому етапі перебуває завдання.

Kanban має і свої недоліки:

- команди повинні бути чисельністю не більше 5 осіб;
- не призначений для довгострокового планування.

Особливості Kanban подано на рис. 1.5.

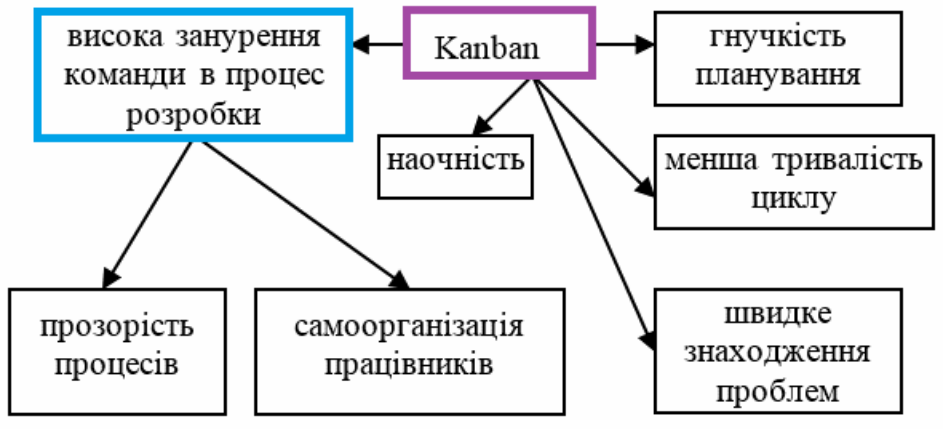


Рис. 1.5. Основні особливості методології Kanban

Методологія Lean використовує концепції зменшення витрат та ощадливого виробництва, допомагає знаходити проблеми в процесі з подальшим вирішенням за допомогою різних практик та інструментів з постійним вдосконалення на всіх етапах реалізації. Основні принципи Lean корелюють із завданнями IT- проєкту [34]:

- визначити цінність конкретного продукту;
- визначити потік створення цінності для цього продукту;
- забезпечити безперервне створення цінності продукту;
- дати змогу споживачеві тестувати продукт;
- прагнути досконалості.

Методологія передбачає за японською традицією, постійне вдосконалення процесу через недопущення непотрібних витрат, виправлення попередніх помилок в управлінні (муда); нерівномірності та невідповідності навантаження (мура); недоцільності, складності в роботі команди (мурі).

Недоліками Lean є:

- необхідний постійний контроль управління з метою виявлення потенційних проблем та підтримки ефективності проєкту;

- на відміну від Scrum, не пропонує чіткого робочого процесу для реалізації частин проєкту, що веде до збільшення строку виконання проєкту;

- модифікація може привести до нерівномірності навантаження на команду, що в свою чергу веде до невдоволення співробітників.

На рис. 1.6 надана схема основних особливостей методології Lean.

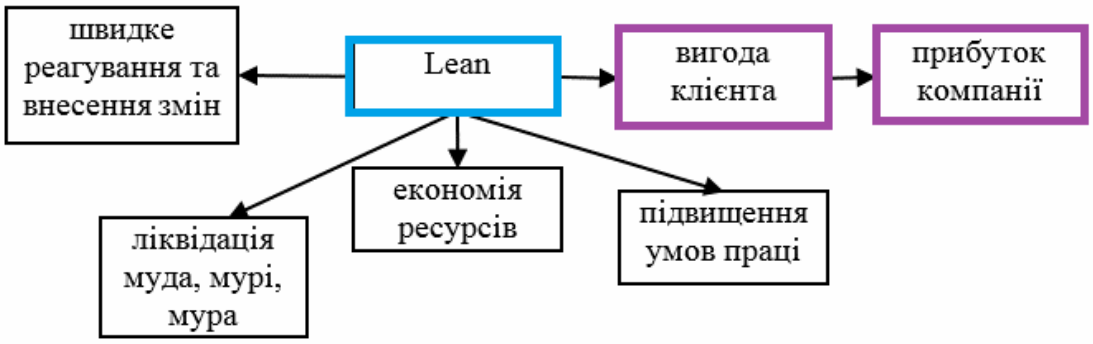


Рис. 1.6. Основні особливості методології Lean

XP (Extreme Programming), або екстремальне програмування має ітеративний фактор прискорення розробки: розробка ведеться короткими ітераціями при наявності активної взаємодії із замовником. Використовується індивідуальний процес розробки, що спрямований на покращення власної продуктивності членів команди IT-проєкту. Використовує також принципи зрілості процесів до практики одного розробника.

Всі гнучкі технології розробки ПЗ ґрунтуються на ітеративності, інкрементності та самокерованості команди IT-проєкту.

1.3. Особливості управління командами проєктів розробки ПЗ

Проекти розробки ПЗ (IT-проєкти), безумовно, характерні для компаній з різноманітними організаційними структурами управління, кількістю

працюючих, потенційного масштабу проєкту та профілем діяльності, але всі вони мають деякі загальні риси. ІТ-проєкти найбільш ризикові, динамічні проєкти, що характеризуються значною кількістю проблем, високим рівнем напруженості та конфліктів, часто вимагають рішення нетипових завдань, і за статистикою, лише третина з них завершується цілком успішно. Успіх впровадження та управління ІТ- проєктами напряму залежить від команди виконавців, їх інформативності, розподілу ролей, повноважень, відповідальності.

Процес створення команди проєкту ускладняється кваліфікацією та компетентностями спеціалістів, участь яких протягом проєкту буде необхідна. Створення команди ІТ-проєкту може ускладнюватися і тем фактором, що є випадки, коли команда не має прямого контролю над всіма спеціалістами, в роботі яких є потреба. Ще одним з факторів, що впливає на успішність виконання ІТ-проєктів, є відсутність ідеальної системи управління проєктами, яка б на 100 % була прийнятною і керівникам і всім членам команди. Тому, питання формування команди є досить актуальним задля забезпечення завершення проєкту в межах терміну і бюджету, та задоволення очікувань замовника.

Розглянемо визначення команди проєкту. Так PMBoK, визначає команду проєкту як «групу осіб, яка підтримує керівника проєкту у виконанні проєкту для досягнення цілей проєкту». Там же, окремо виділяють «команду управління проектом (Project Management Team). Це члени команди проєкту безпосередньо зайняті в операціях з управління проєктом». В наступній версії PMBoK, сьомій, команда проєкту інтерпретується, як «набір осіб, які виконують роботу над проєктом для досягнення його цілей».

Команда створюється на період реалізації проєкту та поєднання її членів повинно формувати синергічний ефект. Особливості роботи в команді допускають, що кожен окремо взятий учасник команди не обов’язково повинен мати повний набір навичок та досвіду для виконання проєкту. В індивідуальних особливостях виконавців є причина створення команд, де

сильні сторони кожного співробітника доповнюють слабкі сторони колег. Обмеженість ресурсів може компенсуватися за рахунок професіоналізму та ефективності сумісних дій усіх членів.

Життєвий цикл команди, за класичним підходом (PMBoK) послідовно проходить декілька етапів, які змінюють один одного:

- формування команди (постановка цілей, розподілення ролей в команді);

- притирання (осмислення цілей, визначення спільного вектору руху);

- нормалізація (досягнення цілей внаслідок компромісів та налагодження комунікацій);

- функціонування (збільшення продуктивності за рахунок оптимізації процесів розробки і самоорганізації членів команди);

- розформування команди (отримання результатів, завершення).

Однак, спочатку слід визначитися з методологією розробки продукту проєкту, а вже потім обирати варіанти створення команди IT-проєкту. В управлінні командами проєктів можуть бути різні підходи до цього процесу: жорсткі (орієнтовані за задачу), гнучкі (орієнтовані на команду) та незалежні команди з високим ступенем незалежності.

Класичний Scrum містить три ролі, що відображено на рис. 1.7.

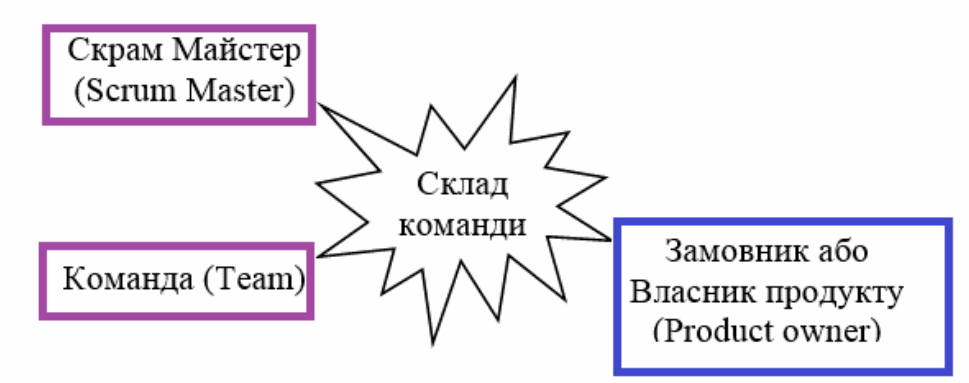


Рис. 1.7. Склад команди в методології Scrum

Scrum Майстер, найважливіша роль (менеджер проєкту) в методології, є з'єднуючою ланкою між замовником та командою. Основні обов'язки

пов’язані із створенням робочої атмосфери довіри, ліквідації перепонів, дотримання практик і процесу в команді. Замовник (власник продукту) відповідає за розробку продукту. Як правило, це менеджер продукту або представник замовника, якщо це аутсорс продукт. Замовник приймає кінцеві рішення для команди в проєкті, відповідає за формування і бачення продукту, керує рентабельністю та очікуваннями замовників.

В методології Scrum команда є самоорганізованою та самокерованою, бере на себе обов’язки виконання задач перед Замовником. Основні обов’язки команди відображені на рис. 1.8.

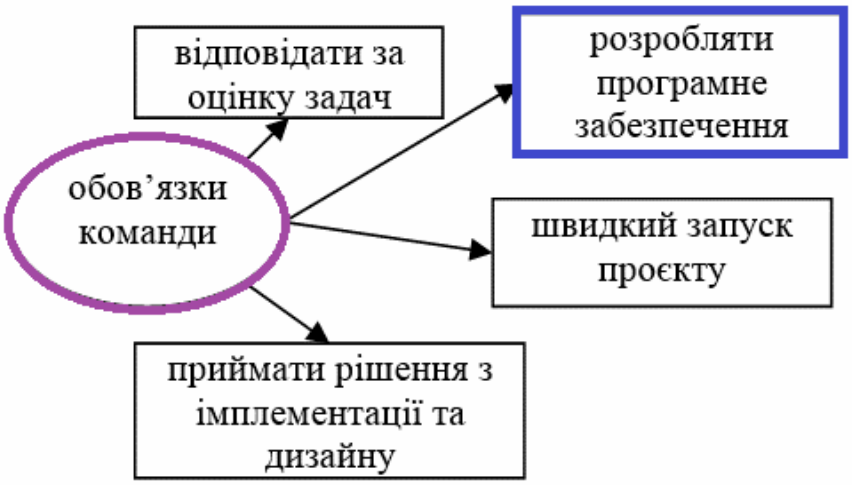


Рис. 1.8. Схема основних обов’язків команди

Kanban працює з командами підтримки, серед яких:

- групи підтримки програмного забезпечення, де важлива швидкість реагування на зміни;

- групи тестування, що працюють окремо від груп розробки;

- служби підтримки;

- інші групи «неосновних виробництв».

Порівняння методологій Scrum та Kanban з точки зору вимог до команди проєкту подано у табл. 1.1.

Таблиця 1.1.

Порівняльна характеристика методологій Scrum та Kanban

Характеристика	Kanban	Scrum
Темп	Повторювані спрінти фіксованої тривалості	Безперервний процес
Випуск релізу	В кінці кожного спринту після схвалення проєктним менеджером (власником товару)	Потік триває без перерв або на розсуд команди
Ролі	Власник продукту, Scrum-майстер, команда розробників	Команда під керівництвом проєктного менеджера; в деяких випадках залучаються тренери по Agile / Kanban
Головні показники	Швидкість команди	Провідний час
Прийнятність змін	У ході спринту зміни небажані, так як можуть привести до невірної оцінки задач	Зміни можуть трапитися в будь-який момент

Інформованість членів команди про прогрес проєкту та його проблеми здійснюється через їх доступ до звітів про стан проєкту та до відгуків стейкхолдерів через: дошки оголошень, поштову розсилку, веб-сайт, наради та презентації найкращих практик. Другий тип спілкування рекомендовано здійснювати через: свята для членів команди, святкування днів народжень, цікаві вікторини, неформальні вечері, тощо. Крім того, до задач управління командою ІТ-проєкту відносять удосконалення професійних навичок молодих членів команди, і це завдання всієї команди.

В якості інструментів відбору до команди проєкту застосовують: індикатор типу особистості Майєрса-Бріггса; модель FIRO-B, що визначає та аналізує фундаментальні міжособистісні відносини; модель сортування темпераментів Кірсі; модель міжпроцесної взаємодії Келера.

Визначення 1.1. Координатор команди ІТ-проєкту (ККІТП) - особа, яка використовує стиль керівництва проєктом «Servant leadership», дозволяє проєктним командам самоорганізовуватися, підвищувати рівень автономії, передаючи прийняття рішень членам команди ІТ-проєкту.

Лідерська поведінка включає в себе:

- усунення перешкод, які можуть заважати роботі команді ІТ-проєкту;

- створення «антидиверсійного щита» від внутрішніх і зовнішніх чинників, які відволікають команду проєкту від поточних цілей;

- застосовує інструменти та підтримку, щоб команда проєкту залишалася задоволеною і продуктивною, вивчає, що мотивує членів проєктної групи та які способи винагороди за якісну роботу допомагають зберегти задоволеність членів проєктної групи.

Динаміку розвитку команди впродовж ЖЦП описують дві моделі: «Сходи Такмана» та «Модель ефективності команди Drexler/Sibbet».

Модернізована модель Брюса Такмана з доданою п'ятою стадією, включає в себе етапи: формування, штурму, нормування, виконання та перерви (рис. 1.9).

Рис. 1.9. Модель розвитку команди «Сходи Такмана»

В моделі виділено психо-емоційні стани особи, яка попадає до проєктної команди. Модель відповідає предмету магістерського дослідження, але в ній відсутні механізми формування команди, управління під час штурму, та покрокового переходу до фази нормування та виконання. Крім того, що модель припускає кооперацію, взаємодію членів команди проєкту задля отримання синергії, вона не враховує особливості управління IT-проєктами, де команда працює на постійній основі, не розформовується після закінчення проєкту, та відповідно не збирається «з нуля». Можливо, за

особливостями ІТ-проєкту можуть бути додані відповідні ролі для врахування саме цих особливостей. Така ситуація має місце, коли команда (можливо команди) є постійними співробітниками ІТ-компанії. Тому доробку та вдосконалення моделі потрібно направити на згадані вище особливості через механізми формування культури та зрілості команди ІТ-проєкту.

Друга модель - це модель ефективності команди Drexler-Sibbet. Автори розробили модель командної роботи, що складається з семи кроків та описує поведінковий стан команди з прив'язкою до етапів ЖЦП. Перші кроки (1 - 4) описують етапи створення проєктної групи, останні (5 - 7) описують стан команди при створенні продукту проєкту (рис. 1.10).

Рис. 1.10. Модель ефективності команди Drexler-Sibbet

До обрання засобів формування команди ІТ-проєкту слід визначитися із загальною формою організації команди, яка буде найкращою для нового проєкту та оптимальною в межах корпоративної культури ІТ-компанії.

Software Extension to the PMBoK [15] рекомендує такі форми організації команди - спеціалізована або універсальна, в першому випадку члени

команди мають вузьку спеціалізацію та працюють над одним, своїм, проєктом. В другому - члени команди мають універсальні широкі компетентності та працюють одночасно над декількома проєктами. Вузька спеціалізація та окремі функціональні підрозділи додають складності в площині економічних рішень управління командою та проєктом. Тому менеджер проєкту повинен знаходити компроміс задля забезпечення успішного завершення проєкту.

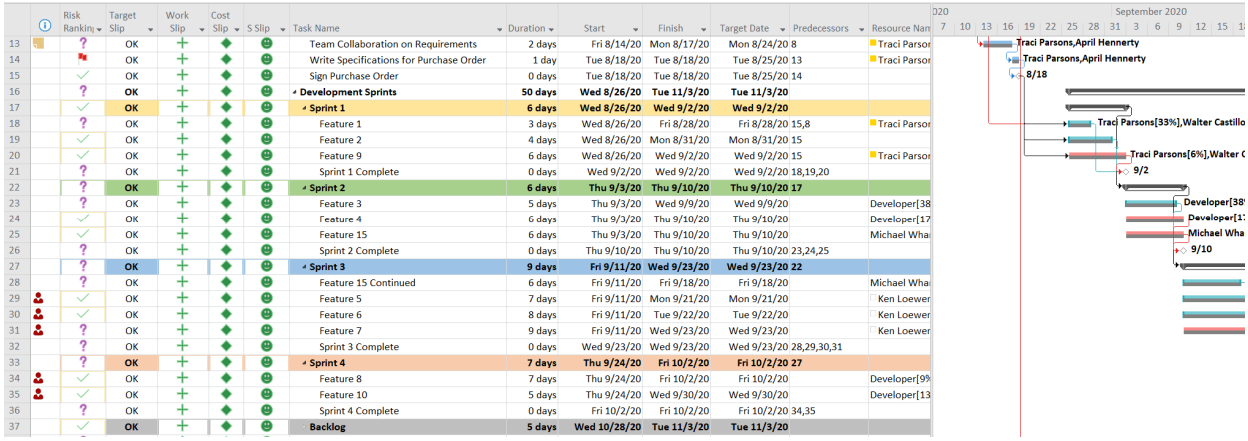
1.4 Аналіз інформаційних технологій управління проєктами розробки ПЗ

Сьогодні ефективне управління проєктами неможливе без використання сучасних програмних засобів, оскільки зростають розміри проєктів, частота їх виконання, обсяги інформації. Автоматизовані системи управління проєктами містять такі структурні елементи:

- засоби для календарно-сіткового планування;
- засоби для вирішення окремих завдань (розробка бюджетів, аналіз ризиків, управління контрактами, часом тощо);
- засоби для спрощення і обмеження доступу до проєктних даних;
- засоби для організації комунікацій;
- засоби для інтеграції з іншими прикладними програмами.

Найбільш відомий та доступний програмний засіб з управління проєктами є наступні.

MS Project. Цей програмний продукт призначений для менеджерів проєктів для розробки планів проекту, розподілу ресурсів і фінансів відповідно до поставлених завдань. Додаток відрізняється простотою і зручністю, дозволяє відстежити хід виконання конкретного проєкту з детальним аналізом обсягу виконаної роботи, та можливістю формування необхідних звітів.

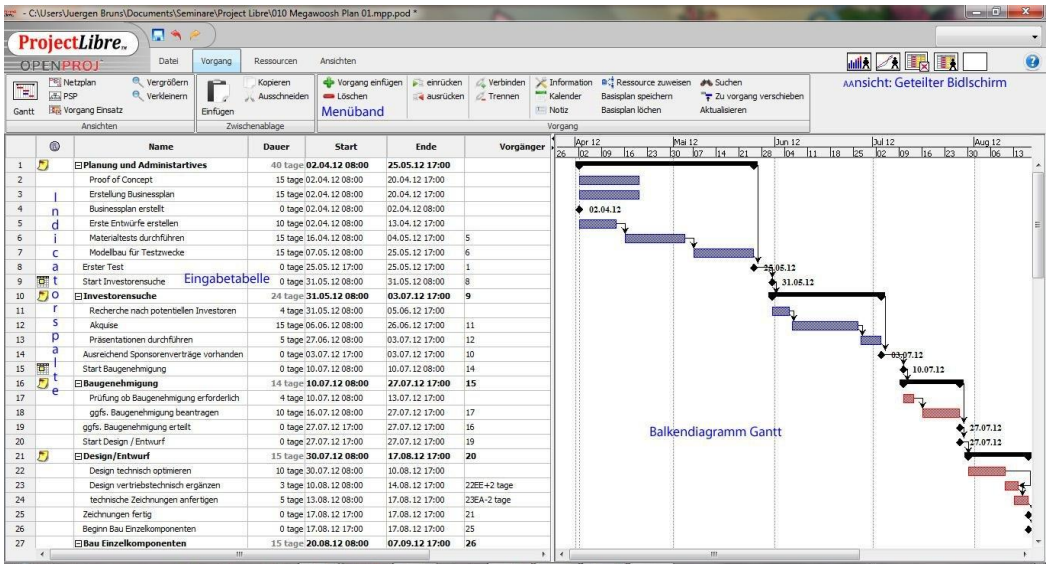


	Risk Ranking	Target Slip	Work Slip	Cost Slip	S Slip	Task Name	Duration	Start	Finish	Target Date	Predecessors	Resource Nam
13		OK				Team Collaboration on Requirements	2 days	Fri 8/14/20	Mon 8/17/20	Mon 8/24/20	8	Traci Parsor
14		OK				Write Specifications for Purchase Order	1 day	Tue 8/18/20	Tue 8/18/20	Tue 8/25/20	13	Traci Parsor
15		OK				Sign Purchase Order	0 days	Tue 8/18/20	Tue 8/18/20	Tue 8/25/20	14	
16		**OK**				**Development Sprints**	**50 days**	**Wed 8/26/20**	**Tue 11/3/20**	**Tue 11/3/20**		
17		**OK**				**Sprint 1**	**6 days**	**Wed 8/26/20**	**Wed 9/2/20**	**Wed 9/2/20**		
18		OK				Feature 1	3 days	Wed 8/26/20	Fri 8/28/20	Fri 8/28/20	15,8	Traci Parsor
19		OK				Feature 2	4 days	Wed 8/26/20	Mon 8/31/20	Mon 8/31/20	15	
20		OK				Feature 9	6 days	Wed 8/26/20	Wed 9/2/20	Wed 9/2/20	15	Traci Parsor
21		OK				Sprint 1 Complete	0 days	Wed 9/2/20	Wed 9/2/20	Wed 9/2/20	18,19,20	
22		**OK**				**Sprint 2**	**6 days**	**Thu 9/3/20**	**Thu 9/10/20**	**Thu 9/10/20**	**17**	
23		OK				Feature 3	5 days	Thu 9/3/20	Wed 9/9/20	Wed 9/9/20		Developer[38
24		OK				Feature 4	6 days	Thu 9/3/20	Thu 9/10/20	Thu 9/10/20		Developer[17
25		OK				Feature 15	6 days	Thu 9/3/20	Thu 9/10/20	Thu 9/10/20		Michael Wha
26		OK				Sprint 2 Complete	0 days	Thu 9/10/20	Thu 9/10/20	Thu 9/10/20	23,24,25	
27		**OK**				**Sprint 3**	**9 days**	**Fri 9/11/20**	**Wed 9/23/20**	**Wed 9/23/20**	**22**	
28		OK				Feature 15 Continued	6 days	Fri 9/11/20	Fri 9/18/20	Fri 9/18/20		Michael Wha
29		OK				Feature 5	7 days	Fri 9/11/20	Mon 9/21/20	Mon 9/21/20		Ken Loewer
30		OK				Feature 6	8 days	Fri 9/11/20	Tue 9/22/20	Tue 9/22/20		Ken Loewer
31		OK				Feature 7	9 days	Fri 9/11/20	Wed 9/23/20	Wed 9/23/20		Ken Loewer
32		OK				Sprint 3 Complete	0 days	Wed 9/23/20	Wed 9/23/20	Wed 9/23/20	28,29,30,31	
33		**OK**				**Sprint 4**	**7 days**	**Thu 9/24/20**	**Fri 10/2/20**	**Fri 10/2/20**	**27**	
34		OK				Feature 8	7 days	Thu 9/24/20	Fri 10/2/20	Fri 10/2/20		Developer[9
35		OK				Feature 10	5 days	Thu 9/24/20	Wed 9/30/20	Wed 9/30/20		Developer[13
36		OK				Sprint 4 Complete	0 days	Fri 10/2/20	Fri 10/2/20	Fri 10/2/20	34,35	
37		**OK**				**Backlog**	**5 days**	**Wed 10/28/20**	**Tue 11/3/20**	**Tue 11/3/20**		

Рис. 1.11. Вигляд системи управління проєктами MS Project

Кожен член команди проєкту в режимі онлайн може вносити власні пропозиції і бачити отримані зміни. Результатом впровадження даного рішення є робота, виконана в повному обсязі і в точно встановлені терміни.

В процесі роботи програми, призначеної для управління окремими проєктами, менеджеру доступний широкий функціонал, заснований на роботі з екранами, які містять фільтри, таблиці і угруповання. У програмі Microsoft Project є кілька десятків єкранів, здатних організувати відомості, призначені для різних цілей.



		Name	Dauer	Start	Ende	Vorgänger
1		**Planung und Administartives**	40 tage	**02.04.12 08:00**	**25.05.12 17:00**	
2		Proof of Concept	15 tage	02.04.12 08:00	20.04.12 17:00	
3		Erstellung Businessplan	15 tage	02.04.12 08:00	20.04.12 17:00	
4		Businessplan erstellt	0 tage	02.04.12 08:00	02.04.12 08:00	
5		Erste Entwürfe erstellen	10 tage	02.04.12 08:00	13.04.12 17:00	
6		Materialtests durchführen	15 tage	16.04.12 08:00	04.05.12 17:00	5
7		Modellbau für Testzwecke	15 tage	07.05.12 08:00	25.05.12 17:00	6
8		Erster Test	0 tage	25.05.12 17:00	25.05.12 17:00	1
9		Start Investorensuche	0 tage	31.05.12 08:00	31.05.12 08:00	8
10		**Investorensuche**	24 tage	**31.05.12 08:00**	**03.07.12 17:00**	**9**
11		Recherche nach potentiellen Investoren	4 tage	31.05.12 08:00	05.06.12 17:00	
12		Akquise	15 tage	06.06.12 08:00	26.06.12 17:00	11
13		Präsentationen durchführen	5 tage	27.06.12 08:00	03.07.12 17:00	12
14		Ausreichend Sponsorenverträge vorhanden	0 tage	03.07.12 17:00	03.07.12 17:00	10
15		Start Baugenehmigung	0 tage	10.07.12 08:00	10.07.12 08:00	14
16		**Baugenehmigung**	14 tage	**10.07.12 08:00**	**27.07.12 17:00**	**15**
17		Prüfung ob Baugenehmigung erforderlich	4 tage	10.07.12 08:00	13.07.12 17:00	
18		ggfs. Baugenehmigung beantragen	10 tage	16.07.12 08:00	27.07.12 17:00	17
19		ggfs. Baugenehmigung erteilt	0 tage	27.07.12 17:00	27.07.12 17:00	16
20		Start Design / Entwurf	0 tage	27.07.12 17:00	27.07.12 17:00	19
21		**Design/Entwurf**	15 tage	**30.07.12 08:00**	**17.08.12 17:00**	**20**
22		Design technisch optimieren	10 tage	30.07.12 08:00	10.08.12 17:00	
23		Design vertriebstechnisch ergänzen	3 tage	10.08.12 08:00	14.08.12 17:00	22EE+2 tage
24		technische Zeichnungen anfertigen	5 tage	13.08.12 08:00	17.08.12 17:00	23EA-2 tage
25		Zeichnungen fertig	0 tage	17.08.12 17:00	17.08.12 17:00	21
26		Beginn Bau Einzelkomponenten	0 tage	17.08.12 17:00	17.08.12 17:00	25
27		**Bau Einzelkomponenten**	15 tage	**20.08.12 08:00**	**07.09.12 17:00**	**26**

Рис. 1.12. Вигляд системи управління проєктами ProjectLibre

ProjectLibre призначений для управління проєктами та має відкритий код. Додаток працює на платформі Java, що дозволяє його запускати в різних операційних системах. Вважається, що ProjectLibre є основною альтернативою з відкритим кодом для Microsoft Project, та повністю сумісна з версіями 2003, 2007 та 2010.

Trello - багатоплатформна система з управління проєктами, яка є безкоштовною. Управління проєктами в ній основане на технології Kanban. Проєкти розміщуються на типових Kanban-дошках зі списками. Списки містять карти, або завдання, які перетягуються за технологією Kanban, відображаючи формування продукту проєкту (або його окремої функції) від ідеї до тестування.

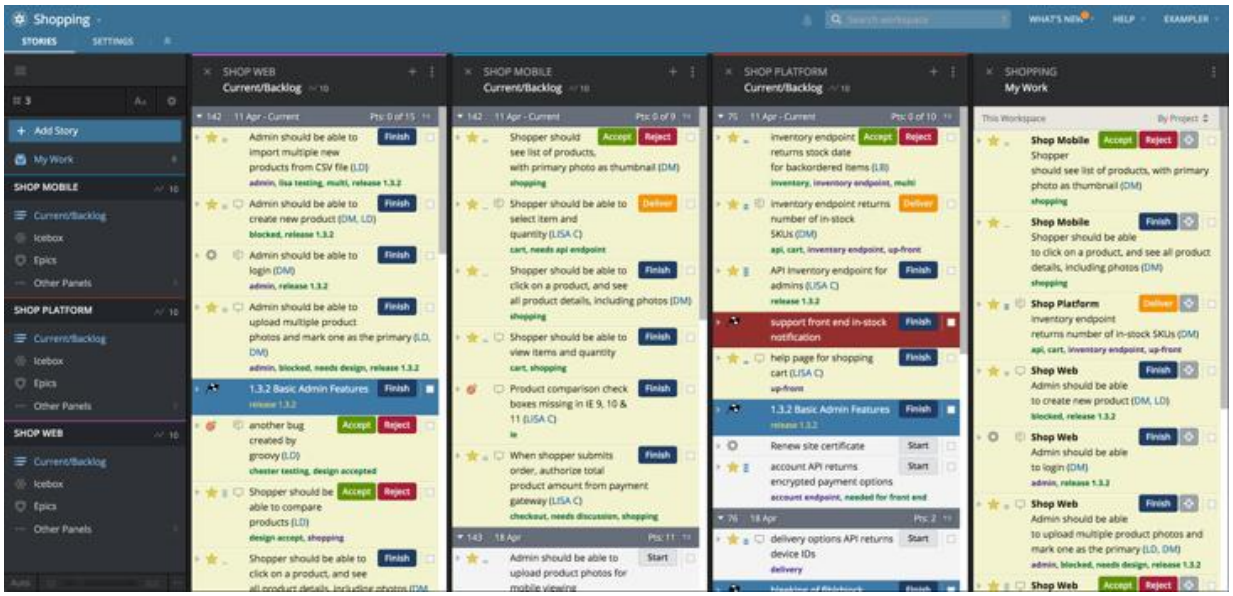


Рис. 1.13. Система Pivotal Tracker

Pivotal Tracker позиціонують як перевірений засіб управління проєктами. Додаток забезпечує спільний погляд на пріоритети команди, процеси співпраці, та інструменти для аналізу прогресу. Tracker візуалізує сферу діяльності, чітко визначає пріоритети та дозволяє команді зосередитись на можливих змінах. Pivotal Tracker має інструменти для адаптації та розвитку, через формування пріоритетів на керовані фрагменти та інше.

Gantter - це додаток з управління проєктами на основі діаграм Ганта, реалізує створення та редагування планів проєктів, які можуть бути повністю інтегровані з Google. Має всі можливості настільних програмних продуктів з управління проєктами, таких як MS Project, та, разом з тим, має всі переваги хмарного сховища. Gantter був розроблений для управління проєктами в Інтернеті. Користувачам Google надається двонаправлена синхронізація завдань із розкладів Gantter в їх календарі Google, зберігання файлів на Google Drive і Google Team Drive, спільне редагування в режимі реального часу, вбудовані коментарі Google, можливість запуску чату Google з членами команди з їх розкладу Gantter.

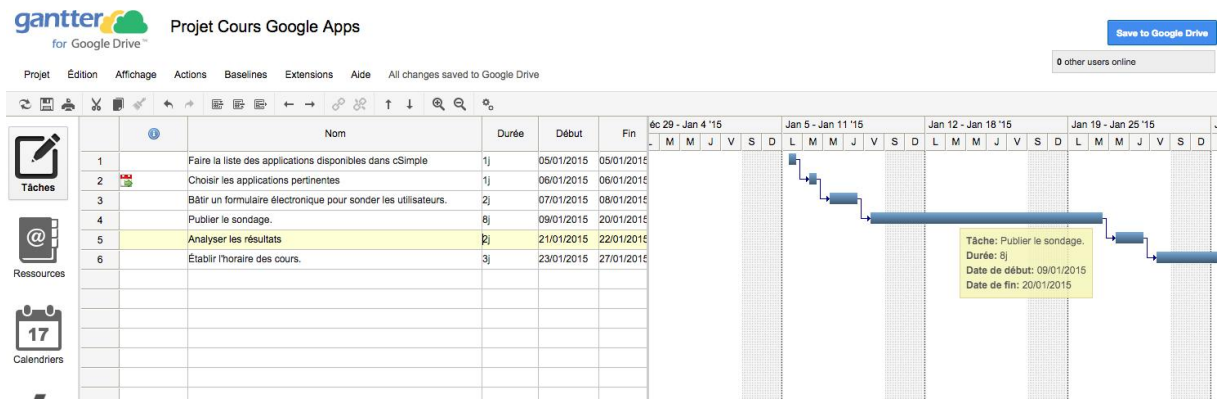


Рис. 1.14. Додаток управління проєктами Gantter

Worksection - український онлайн сервіс з управління проєктами, мобільна та багатомовна система управління. Серед функціональних можливостей слід відзначити вбудований тайм-трекінг з розрахунком витраченого часу у форматі час/гроші, або лише у годинах, що ідеально для компаній з погодинними ставками. Worksection може бути застосований як для невеликих команд, так і великого бізнесу без прив'язки до сфери діяльності.

Для проєктів, в яких для розробки застосовано гнучкі технології, в якості засобів автоматизації управління необхідно обирати відповідне програмне забезпечення. Розробником Jira є компанія Atlassian (Австралія).

Система призначена для відстеження помилок, управління завданнями, управління agile-проєктами з хмарною і серверною версіями. Вихідний код є відкритим для скачування.

Youtrack призначена для гнучкого управління проєктами для розробників ПЗ. Розробник — JetBrains. Доступна хмарна та десктопна версії. Вартість користування зростає пропорційно кількості користувачів, для команди до 10 осіб - безкоштовна, також як і Jira.

Сучасні програмні засоби управління проектами стрімко розвиваються з врахуванням тенденцій ведення проєктів сьогодні - розподіленої роботи над проектами, віртуальних та міжкультурних команд, високої нестабільності середовища проєктів.

1.5 Висновки до розділу

В першому розділі було проаналізовано особливості сучасних проєктів розробки програмного забезпечення (ІТ-проєктів), досліджено специфіку управління ІТ-проєктами з врахуванням їхньої інноваційності та відповідної участі членів команди проекту, проаналізовано засоби управління такими проєктами. Проведено огляд підходів до управління командами ІТ-проєктів та подано результати аналізу управління командами, що використовують гнучкі технології розробки програмного продукту. Подано огляд сучасних технологій управління ІТ-проєктами, що мають у собі інноваційну складову та застосовують гнучкі технології. Проведено аналіз інформаційних технологій управління проєктами.

РОЗДІЛ 2. ІННОВАЦІЙНІ МОДЕЛІ УПРАВЛІННЯ КОМАНДАМИ ПРОЄКТІВ РОЗРОБКИ ПЗ

2.1. Дослідження та опис технологій управління проєктами

2.1.2 Технологія дизайн-мислення

Однією з особливостей ІТ-проєктів можна вважати те, що більшість з них розробляються «на замовлення», тобто ІТ-проєкти є суто унікальні. Це означає, що рівень новизни продукту достатньо високий, він перевищує середнє значення. Мається на увазі, що продукт проєкту є новим, або порівняно з діючими аналогами, в ньому суттєво змінені його властивості та/або функціонал, суттєво змінена сфера застосування, тощо. Мова йде про те, що користувачам пропонується новий продукт, з яким вони не знайомі, не використовували, не припускали навіть можливості його існування.

Для виокремлення таких продуктів проєктів можна запропонувати для них назву «Три «НІ»» = «не знайомі, не використовували, не припускали навіть можливості його існування».

Просуненням подібних продуктів на ринок збуту зараз займається так званий агресивний маркетинг, який нав'язує користувачеві продукти, які йому в принципі і не потрібні. Це виконується на основі різних психологічних технік.

Але, серед продуктів «З-НІ» обов’язково існують продукти, що будуть корисні для споживачів, та змінять їх життя на краще. Для того, щоб з’ясувати структуру такого продукту, використовують технології дизайн-мислення. Спочатку технологія дизайн-мислення застосовувалася лише для start-up проєктів. Але пізніше, цю технологію стали застосовувати й для загального управління проєктами. Часто розглядають можливість застосування дизайн-мислення для управління ІТ-проєктами. Вказується на те, що клієнт-орієнтована парадигма створення продукту ІТ-проєкту вимагатиме й відповідних змін в технологіях управління такими проєктами.

Показано, що впровадження дизайн-мислення в методологію управління проєктами ІТ-компанії створить такі умови, що члени команди стануть частиною цієї компанії, а такі співробітники вже будуть перейматися потребами замовників і користувачів.

Технологія дизайн-мислення структурована (рис. 2.1), має властивість системності і має взаємопов'язані етапи [1, 13]. Перший етап - збір і аналіз вимог від стейкхолдерів, які прямо або опосередковано будуть втілені у майбутньому продукті проєкту. У класичних термінах дизайн-мислення, перший етап то є емпатія.

Рис. 2.1. Технологія дизайн-мислення

Другий етап - формулювання концепції майбутнього продукту з точки зору його призначення, або фокусування. Третій - генерація можливих рішень того, яким може бути майбутній продукт та вибір одного з них для подальшого створення прототипу. Четвертий етап - прототипування одного з відібраних варіантів для його перевірки, п’ятий - тестування, отримання зворотного зв'язку від користувачів та замовника, внесення змін (доробка) прототипу, або повернення до фокусування та генерації ідей, якщо користувач незадоволений продуктом.

Однак, такий розподіл на етапи дещо умовний, і дана технологія — це щось більше, ніж процес з п'яти або семи кроків. Вказані на рисунку 2.3 лінії зворотного зв’язку передбачають повернення від процедури розробки продукту на попередні етапи, за умови незадоволеності замовником поточної версії продукту.

Проте такий схематизм дозволяє виділити деякі ролі, правила умовності для членів команди, моделі поведінки і формальні очікування, які у підсумку повинні сформувати принципи і концепцію управління командою проєкту.

Дизайн-мислення починається з емпатії — глибокого розуміння людей, яким вона призначена. Команда проєкту, яка думає як дизайнери, ставить себе на місце замовників та користувачів. Передбачається, що управління проєктом вже клієнто-орієнтоване, але тут ще необхідно розуміння стейкхолдерів, з їх реальними проблемами, а не враховувати їх тільки як джерело власного доходу, або як набір демографічних даних про вік, рівень доходу та сімейного статусу і т.і. Використання дизайн-мислення передбачає глибоке осмислення їх емоційного статусу, потреб і побажань щодо майбутнього продукту проекту.

Дизайн-мислення передбачає ітераційний процес. Ухвалення рішення про те, «що рішення знайдено», буде прийматися на основі зворотного зв'язку від замовника, а саме про те, що поточна вдосконалена версія продукту проєкту задовольняє очікування замовника. Це висуває додаткові вимоги до команди проєкту, а саме необхідність постійного навчання або підвищення своїх компетентностей і навичок.

Класичний менеджмент передбачає прямий, лінійний метод вирішення: визначення завдання/проблеми, пошук декількох варіантів рішень і вибір одного з них.

Дизайн-мислення для успішних рішень пропонує експериментувати, причому експериментувати з емпатією, тобто експеримент проводиться

виключно з метою задоволення очікувань замовника. А для цього знову потрібно постійно вчитися.

Ще одна відмінність між класичним менеджментом і дизайн-мисленням, яке повинно бути враховано в інструментах управління командою ІТ-проєкту, - це відмінності в базових передумовах і факторах, на основі яких приймаються рішення. Класичний менеджмент передбачає раціональність і об'єктивність. Рішення приймаються на основі економічної або технологічної логіки. Реальність точна і вимірювана кількісно.

У практиці дизайн-мислення постійно відбуваються ітерації — не тільки в часі, але і між рівнями абстракції, між загальною картиною і конкретними елементами. Команда проєкту, що використатвує технологію дизайн-мислення виготовляє моделі і прототипи, для візуалізації і матеріалізації ідей для замовника.

2.1.2 Технологія створення мінімально життєздатного продукту

В комбінації з технологією дизайн-мислення доречним є використання концепції створення мінімально життєздатного продукту, як засобу для створення «прототипу» продукту ІТ-проєкту.

Тому наступним кроком необхідно формалізувати термін «прототип» - як поточну версію продукту проєкту, яка існує на поточному етапі життєвого циклу проєкту. В якості такого терміну пропонується використовувати «Мінімально життєздатний продукт» (МЖП). Цей термін найбільш повно та ємно розкриває результат процесів управління за технологією дизайн-мислення. Коротко його суть можна викласти наступним чином: команда ІТ-проєкту повинна у найкоротші строки подати замовникові прототип продукту проєкту. Зрозуміло, якщо строки мінімальні, то він володітиме мінімальним функціоналом, але ідея його така, щоб він продемонстрував працездатність майбутнього продукту проєкту.

В англомовних дослідженнях МЖП називають Minimum Viable Product (MVP) - перша пропозиція клієнтам з мінімальним набором функцій, яка

представляє максимальну цінність. Як можна помітити з назви MVP, існує дилема між "величиною мінімальності" в продукті проєкту і "кількістю його готовності" (життєздатністю). Ітеративний процес створення MVP передбачає, що реально повернутися у вихідну точку або на певний етап розробки.

Слово «viable» (життєздатний) в абревіатурі MVP можна трактувати по-різному у різний спосіб: придатний для тестування; придатний для використання.

Для створення MVP застосовують алгоритм з 4 етапів:

- Think it - поміркуй;
- Build it - побудуй;
- Ship it - розповсюджуй;
- Tweak it - налаштуй те, що побудував.

Етап обмірковування передбачає висування гіпотези, створювання опису та начерки прототипу, які стануть в нагоді для розробки. По завершенні гіпотез команда будує MVP - створює простий функціонал, щоб визначитися, чи відповідають його функції потребам замовника та користувачів, чи ні. Якщо стейкхолдери не зацікавлені, від ідеї відмовляються. Коли задумка знаходить відгук у майбутніх користувачів продукту проєкту - її тестують і переробляють, поки не буде отриманий готовий продукт, який можна поширити на всіх користувачів.

Такий підхід дозволяє знизити ризики і економно витратити гроші на розробку продукту проєкту. Тобто, головною задачею використання ідеї створення MVP є якомога раніше визначити, чи потрібен продукт проєкту замовнику, та чи представляє він собою цінність.

2.1.3 Методика Proof of concept

Перевірка концепції (PoC) — це демонстрація практичної здійсненності будь-якого методу, ідеї, технології, з метою доведення факту, що метод, ідея або технологія працює. Також помилково приймати PoC як "чернетку"

проєкту, яку перед завершенням треба доопрацювати. Вважається, що PoC набагато ближче до дослідження, ніж до розробки працюючого продукту.

«Прототип» з етапу технології дизайн-мислення потрібен для перевірки ідеї у «фізичному розумінні», тобто у розумінні продукту проєкту, у розумінні MVP. У випадку інноваційних проєктів, стає за потребу перевірка ідеї у сенсі технологічного здійснення певних розрахунків, спроектованих систем тощо. Якщо ключовою функцією планованого продукту проєкту є, наприклад, розпізнавання образів з похибкою менш 1%, то в межах PoC ми перевіримо технічну можливість таких обчислень, а при створенні прототипу змовнику буде продемонстровано розташовування відповідних полів введення, та інтерфейсу роботи із зовнішніми камерами стеження.

Для неінноваційних проєктів перевірка концепції є необов'язковим етапом проєкту, але для інноваційних проєктів відсутність PoC у рази збільшує невизначеність вже на початку проєкту, бо вона визначає загрозу безпеки, непередбачувану складність проєктів, дефіцит кваліфікованих кадрів. Тому застосування PoC з’ясовує не тільки працездатність майбутніх рішень, але й пропорційність витрат та ризиків проєкту.

2.1.4 Креативне управління проєктами

Управління проєктами приймає невизначеність, зміни та непередбачуваність як норму. В цьому випадку застосування традиційних методів управління проєктами, які спираються на ретельне планування, майже неможливе. Збільшення прогнозованості інноваційних проєктів може забезпечити використання креативного потенціалу команди. Креативність має стає ключовою навичкою команди для забезпечення успішного завершення IT- проєкту.

Креативний («creative») означає «творчий», як пошук неординарних підходів, розкриття інтелектуальної складової особостості. Під креативністю розуміється універсальна творча здатність, рівень творчої обдарованості, який є однією з характеристик особистості.

Визначення креативності має різне забарвлення та наголоси в залежності від сфери застосувння. Креативність позиціонується як «здатність до конструктивного, нестандартного мислення та поведінки, усвідомлення і розвитку свого досвіду». Автор описує прояв креативності як швидкість та оригінальність мислення, схильності до естетичних цінностей, що дає можливість знаходити ефективні рішення та переборювати стереотипність мислення та поведінку у звичних ситуаціях, приймати нові, оригінальні рішення.

В креативному управлінні творчість виступає як засіб підвищення ефективності та джерело інновацій. Необхідність креативного управління виникає тоді, коли стандартні принципи та засоби управління не дають результату. Креативне управління - це сукупність засобів впливу задля забезпечення творчого підходу, формування індивідуальних здібностей та підвищення імовірності успішного завершення ІТ-проєктів з інноваційною складовою.

Креативне управління містить у собі:

- оцінку творчого потенціалу особистості;
- створення творчої атмосфери в команді ІТ-проєкту, яка працює певний час разом, має свої традиції, неформальних лідерів та розподіл функціональних ролей;
- формування тимчасових творчих колективів задля участі в груповій динаміці;
- оцінка мотиваційних установок у найближчій перспективі завдань, що вимагають застосування творчих здібностей членів команди.

Окремо виділяють «продуктивну креативність», яка заснована на позитивній мотивації, заохоченні спілкування, на тренінгах та позитивній взаємодії членів команди. Така креативність проявляється через самореалізацію, підвищення потреб особистостей, надбання знань та навичок, появу нових ідей. Продуктивну креативність та позитивну

мотивацію пов'язують із самореалізацією та підвищенням рівня потреб згідно з пірамідою А. Маслоу.

2.2 Концептуальна модель інноваційного управління командою ІТ-проєкту

Головна ідея полягає у тому, що застосування креативних підходів до управління, а саме, технології дизайн-мислення, концепції MVP та PoC, при розробці та реалізації ІТ-проєктів потребує відповідних знань та навичок, носіями яких є безпосередньо члени згаданих проєктних команд. Тому здається логічним застосувати ті ж самі принципи до створення та управління такими командами, тим паче, що відповідні знання вже будуть присутніми в таких командах на початковій фазі проєкту.

Ця ідея дала змогу сформулювати концептуальну модель інноваційного управління командою ІТ-проєкту, графічне представлення якої подано на рис. 2.2.

Початковими даними в моделі є концепція ІТ-проєкту, яка затверджена його замовником. У часовому просторі це відповідає ситуації за PMBoK, коли вже затверджений устав, сформовано бачення продукту проєкту та очікування його користувачів (або їх проблеми, які має вирішити цей продукт). Початковими умовами до моделі є сформовані вимоги до продукту, на основі яких мають бути висунуті вимоги до виконавців та ролей команди проєкту. Початкові умови формуються у блоках 1-3, але ці дії не входять безпосереднє до управління командою.

Слід зауважити, що розроблена концептуальна модель повністю відповідає загальному підходу управління командою проєкту, а саме, складається з наступних етапів: формування команди, розвиток команди, моніторинг та контроль команди.

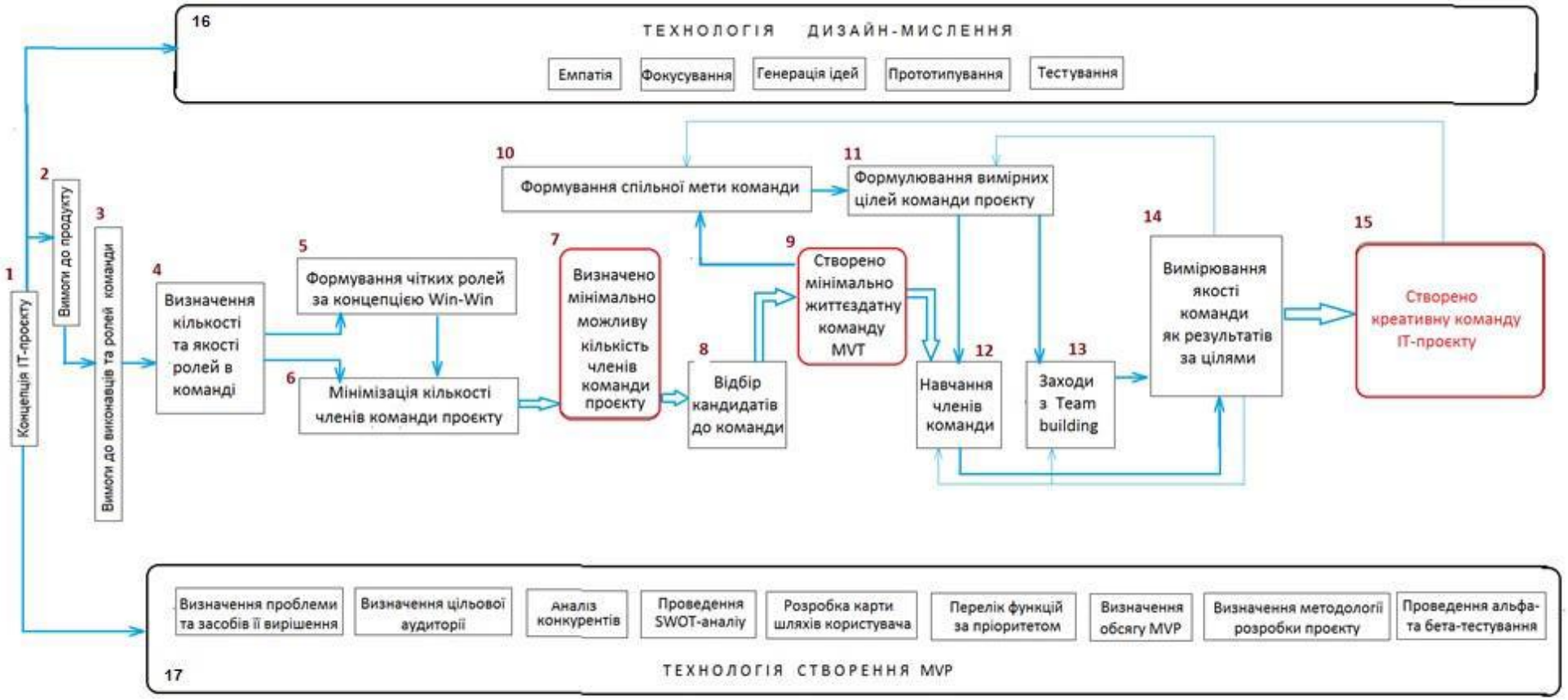


Рис. 2.2. Концептуальна модель інноваційного управління командою проєкту розробки ПЗ

Застосування креативних інструментів до управління командою, обумовлено вимогами сучасних ІТ-проєктів та рекомендацій, що подані у новітній редакції РМВоК 7. Результатом моделі є сформована креативна команда ІТ-проєкту (блок 15).

Таким чином, представлена модель є відображенням процесу створення інноваційної команди ІТ-проєкту. Тут слід зауважити, що «створення такої команди» не є першим етапом управління командою, за РМВОК, а саме її «формуванням», мається на увазі, що в результаті інноваційного управління командою ІТ-проєкту за час його життєвого циклу буде створена креативна за визначенням команда.

Визначення 2.1. Інноваційна команда ІТ-проєкту - це самокерована команда, що сама зорганізується, в якій відсутні централізовані (зовнішні) засоби управління.

Така команда сама бере на себе відповідальність, сама визначає проблеми/завдання та приймає рішення щодо їх вирішення. ККІТП формує загальні власні цілі (які не є цілями ІТ-проєкту) з урахуванням кожного учасника (члена команди) та досягає їх відповідними шляхами впродовж проєкту.

Обмеження процесу: до команди не можуть бути відібрані особи, що не мають досвіду командної розробки ІТ-проєктів, які мають інноваційну складову.

Допущення процесу: вважається, що кандидати в команду мають навички розробки програмного продукту вище середнього рівня; приймали участь в ІТ- командах, що самокерувалися; мають здатність до емпатії, або мають властивості емпатії.

Емпатія має бути направлена у бік команди, тобто керівник проєкту має визначити усі проблеми та цілі команди та забезпечити їх досягнення, через формування ідей, фокусування, створення прототипу команди, який задовольнить очікування її членів.

Таким чином, першим елементом концептуальної моделі є визначення кількості та якості ролей у команді (блок 4). Як було зазначено вище, даний елемент спирається на вимоги до виконавців та ролей команди. Відмінністю зазначених сутностей є те, що перша, тобто вимоги, формується за технологією створення програмного продукту, а друга — приймається рішення про кількість з урахуванням наявних ресурсів та корпоративної культури ІТ-компанії, що реалізує проєкт.

Наступний елемент моделі направлений на мінімізацію кількості команди з урахуванням техніки MVP - мінімально життєздатного продукту, а саме направлений на створення мінімально життєздатної команди (блок 6). Але ця процедура доповнена використанням стратегії Win-Win, тобто вона направлена не на зменшення команди будь-якою ціною, можливо за рахунок якості розробки, або виснаження сил та особистого часу членів команди, а з урахуванням тих критеріїв, коли кожен з членів команди, має залишитися «у виграшу». Ця процедура відбувається у блоці 5.

В результаті кроків 5 та 6 отримано перший проміжній результат, а саме, визначено мінімально можливу кількість членів команди проєкту з урахуванням елементів корпоративної культури організації та поточного стану трудових ресурсів в проєкті на даний момент. Цей результат на моделі зображено елементом червоного кольору (блок 7).

Після того, як мінімально можливу кількість визначено, слід переходити до безпосереднього відбору кандидатів до команди (блок 8). Метод відбору буде подано пізніше.

Після відбору отримаємо другій проміжний результат управління командою ІТ-проєкту, а саме, мінімально життєздатна команда (MVT) створена (блок 9).

Цей факт ініціює дві групи процесів: перший - починається формування спільної мети та вимірних цілей безпосередньо для обраних членів команди (блоки 10-11), а другий - навчання команди (блок 12).

На відміну від існуючих алгоритмів управління командою ІТ-проєктів, розроблена модель позиціонує членів команди як особистостей, які саме як особистості повинні зростати під час свого життя, і проєкт не має цьому заважати, а навпаки, сприяти цьому. Це означає, що застосована вище стратегія Win-Win залишається активованою впродовж усього життєвого циклу проєкту, тобто у блоці 10 реалізується алгоритм створення спільної мети команди на основі стратегії Win-Win.

Далі, спільна мета має бути трансформована у вимірні цілі команди проєкту (блок 11). Друга група процесів, а саме навчання команди, розподіляється на два вектори. Перший - це професійне зростання (професійне включає в себе три складові:

1) вдосконалення навичок з програмування,

2) навчання технікам дизайн-мислення та MVP,

3) набуття та розширення властивостей емпатії.

Другий вектор - соціально-психологічний розвиток, що пов’язаний із створенням позитивного внутрішнього середовища проєкту, побудови команди, її згуртованості та ін. Ці процеси реалізовано у блоках 12 та 13, відповідно.

Заходи з навчання постійно перевіряються шляхом вимірювання якості команди (блок 14).

Одним з показників якості є перелік компетентностей членів команди, які можуть бути виміряні стандартними, або спеціально розробленими засобами. Ці компетентності набуваються членами команди за раніше сформованими цілями (блок 11). Результати вимірювання формують петлі зворотного зв'язку до блоків 11 та 12, які дозволяють вносити корективи в перелік вимірних цілей команди проєкту (блок 11), або безпосередньо у заходи навчання (блок 12).

Якщо в результаті вимірювання показники якості відповідатимуть очікуванням команди, то можна вважати, що процес управління командою ІТ- проєкту завершено, і команду - створено (блок 15).

Але зауважимо, що цей процес динамічний і необхідно перевіряти, чи не змінилися очікування та цілі членів команди впродовж ЖЦП, що реалізовано стрілкою зворотного зв’язку до блоку 10.

Крім того, зауважимо, що існують випадки, коли необхідно перевіряти спільну мету та вимірні цілі команди проєкту, які можуть зхмінюватись в зв’язку із зміною складу команди або із зміною або додаванням нової предметної галузі у проєктне середовище, яке потребує вивчення.

Таким чином, процес створення команди відтворюється не один раз впродовж проєкту, а має відбуватися постійно, або із заданою циклічністю, в залежності від особливостей конкретного ІТ-проєкту.

Впродовж всього процесу управління командою згідно концептуальної моделі застосовується технологія дизайн-мислення (блок 16) та концепція MVP (блок 17).

2.3 Семіотична модель управління командою ІТ-проєкту

Семіотичне управління найчастіше використовують як узагальнене управління. "Узагальнення" необхідне при складних ситуаціях, коли має місце великий обсяг вихідних даних і велика кількість залежностей та зв'язків. Прийняття рішень у складних ситуаціях потребує узагальнення та редукції, проте при цьому значні параметри об'єкту управління та поточної ситуації вважаються незмінними. Тому, у складних ситуаціях семіотичне управління є альтернативно - значимим рішенням.

Вважається, що семіотичні моделі — це моделі верхнього рівня абстракції по відношенню до ситуаційних моделей. Які, в свою чергу, є моделями верхнього рівня по відношенню до технологічних моделей, які зазвичай і використовують в управлінні проєктами, спираючись на технологію розробки продукту проєкту.

Такий «верхній» рівень семіотичних моделей значно скорочує час та зменшує складність аналізу проєкту та логіку проєктних рішень, а

застосування семіотичного управління підвищує швидкість та якість управління проєктами. Семіотичне управління є ключовим при створенні систем управління складними ситуаціями, якими, безумовно, є управління командою ІТ- проєкту з інноваційною складовою. В даному випадку складність проявляється у вигляді відсутності однозначної залежності результатів від вхідних параметрів.

Наприклад, якщо замовник (керівник ІТ-компанії/команди/проєкту) хоче створити команду ІТ-проєкту, то як система управління реагуватиме на таку команду на вході? Можливо, йому спочатку необхідно просто створити команду, яка буде зацікавлена в управлінні та реалізації даного проєкту. Однак, така інформація на вході відсутня, і система управління ухвалить неправильне рішення.

Ще однією проблемою є те, що команда управління інноваційними ІТ-проєктами не є статичною системою. Протягом життєвого циклу команди можуть змінюватися як набір вхідних вимог (мається на увазі вимоги до компетентностей членів команди з боку проєктних задач, які вони мають виконувати), так і вихідні очікування самої команди та/або її керівника, або «слугуючого лідера». Це потребує зміни критеріїв управління впродовж ЖЦП.

В таких випадках при управлінні об'єктом з динамічною та унікальною структурою застосовують метод ситуаційного управління. Цей метод заснований на понятті ситуації, її класифікації та зміні.

Зазначимо спільність ситуаційного та семіотичного управління. Обидва підходи спираються на використанні великої кількості правил, що описують поведінку об'єкта управління, в даному випадку команди управління як частини інноваційних ІТ-проєктів, а також описують множину правил, які пов'язують ситуації, пов'язані з об'єктом, і правило прийняття рішення.

Завданням управління в обох випадках є вибір правила із заданого переліку для використання у конкретній ситуації. При цьому модифікація, тобто зміна семіотичної моделі управління - допускається, тобто, правила

змінюються на основі аналізу турбулентного оточення проєкту, очікування команди та результатів попереднього управління.

В ситуаційному управлінні сукупність відомостей про поточну структуру та стан об'єкта називають поточною ситуацією.

Крім того, вводять поняття повної ситуації SCi (*Complete situation*), яка включає: поточну ситуацію SRi (*Current situation*), знання про стан системи управління, а також знання про технологію управління.

І третьої значної сутністю системи управління називають обмежену множину можливих керуючих впливів на об'єкт, які називають однокрокові рішення OSSi (One step solution).

Тоді дію управління, який переводить систему з поточної ситуації *SRi* у поточну ситуацію *SRm* з урахуванням повної ситуації *SCi*, можна записати в такому вигляді:

$$\{SC_i; SR_i\} \underset{OSS_i}{\Longrightarrow} SR_m$$

Наведене правило означає наступне: якщо поточна ситуація і повна ситуація в команді проєкту і в системі управління проєктом дозволяють застосувати якийсь однокроковий вплив, і він застосовується, то це призводить до появи нової поточної ситуації *SRm*.

Можливість системи управління впливати на об'єкт визначається набором логіко-трансформаційних правил.

На рисунку 2.3 наведено структуру управління командою IT-проєкту на основі ситуаційної моделі.

Рис. 2.3. Ситуаційна модель управління командою інноваційного проєкту

Процес інноваційного управління командою ІТ-проєкту з урахуванням поданої ситуаційної моделі представимо у вигляді наступного алгоритму:

1. У блоці 1 відбувається формалізація поточної ситуації, в якій знаходиться команда ІТ-проєкту у теперішній час. Сформований опис поточної ситуації команди, як об'єкта управління, передається на блок 2.

2. В блоці 2 оцінюється необхідність втручання системи управління у процес. Якщо необхідність втручання існує, формалізований опис поточної ситуації передається у класифікатор (блок 3).

3. У блоці 3 виконується порівняння поточної ситуації з одним або декількома класами, яким відповідає одне однокрокове рішення.

Оскільки кількість можливих керуючих впливів на об'єкт завжди обмежено, а кількість існуючих повних ситуацій значно більша за кількість можливих рішень, то створюється класифікатор, який розділяє згадані повні ситуації на класи.

4. У розв’язувачі (блок 4) на основі логіко-трансформаційних правил, обирається правило, яке має використовуватися в даній ситуації. Якщо таке правило одне, то для об'єкту управління формується відповідний керуючий вплив (блок 7).

5. Якщо в розв’язувачі виявлено кілька можливих до застосування правил, то ці попередні рішення обробляються і серед них обирається найкраще (блок 5), на підставі якого формується керуючий вплив (крок 4).

6. Якщо ж в результаті проведеного аналізу, кращого рішення знайдено не було (тобто воно відсутнє для даної ситуації, блок 6), то обирається одне з рішень, яке відповідно до теорії ситуаційного управління надасть команді ІТ-проєкту, як об'єкту управління, незначний вплив (блок 7).

Або, через недостатність даних, система прийняття рішення відмовиться від будь-якого впливу. Тобто, у будь-якому випадку, система має рекомендувати рішення про ситуативний вплив на команду: ординарне рішення; найкраще рішення з можливих; незначне рішення із мінімальним впливом; рішення про відсутність дії.

Формальний опис моделі креативного управління командою IT-проєкту, або формальна модель, має ґрунтуватися на чотирьох множинах:

$$Y = \langle BE, R, A, U \rangle$$

Множина BE формальної моделі містить усі базові елементи, які є основою інших елементів системи. В цій множині відсутні будь-які обмеження, при цьому кожен з елементів повинен відрізнятися від інших. Базовими елементами команди IT-проєкту можна вважати: імена членів команди; назви їх ролей; їхні компетентності і т.і.

Множина R — це множина синтаксичних правил. На їх основі формуються сукупності з базових елементів, які вважатимуться синтаксично правильними. На цю множину також не накладається спеціальних обмежень, за винятком цільового призначення: «чи є побудована на основі правил сукупність базових елементів синтаксично правильною». У термінах проєктної команди синтаксично правильними будуть вважатися всі поєднання, наприклад, імен членів команди та назв ролей, імен членів команди та їхніх особистих компетентностей тощо.

Множина аксіом, або множина A, утворює будь-яку множину синтаксично правильних сукупностей. У нашому випадку це можуть бути поєднання, наприклад, імені члена команди, назви його ролі та його компетентності або результати навчання, при цьому вони повинні утворювати повний стан системи. Однак, якщо після цього до нього додати, наприклад, номер телефону або будь-який інший ідентифікатор, отримана сукупність вже не буде вважатиметься аксіомою.

Множина U складається із семантичних правил, які розширюють множину аксіом, через додавання нових синтаксично правильних сукупностей.

У термінах проєктної команди будь-яке поєднання з іменем члена команди та його обов'язкових параметрів є семантичною правильною

сукупністю. Однак, при додаванні будь-яких параметрів, які не належать об'єкту, створюється прецедент, в якому така сукупність вже не буде семантично правильною.

Однак, як було сказано вище, команда проєкту, як і сама система «Проєкт», є відкритою, нелінійною, динамічною. Для таких систем ефективніше використовувати семіотичні моделі, які беруть за основу, описану вище, формальну модель.

Суть семіотичної моделі полягає в тому, що для кожної з множин формальної моделі системи застосовується набір правил, якими вона змінюється, тобто можуть змінюватись базові елементи, семантичні та синтаксичні правила, аксіоми. Семіотичну модель представимо у вигляді залежності:

$$S = \langle Y, \varepsilon^{BE}, \varepsilon^{R}, \varepsilon^{A}, \varepsilon^{U} \rangle,$$

де Є— правило зміни множин: BE, R, A, U.

В семіотичній моделі можливо змінювати прагматику базових елементів, їхню семантику та синтаксис. На рисунку 2.4 показана семіотична модель управління командою IT-проєкту у вигляді мережі.

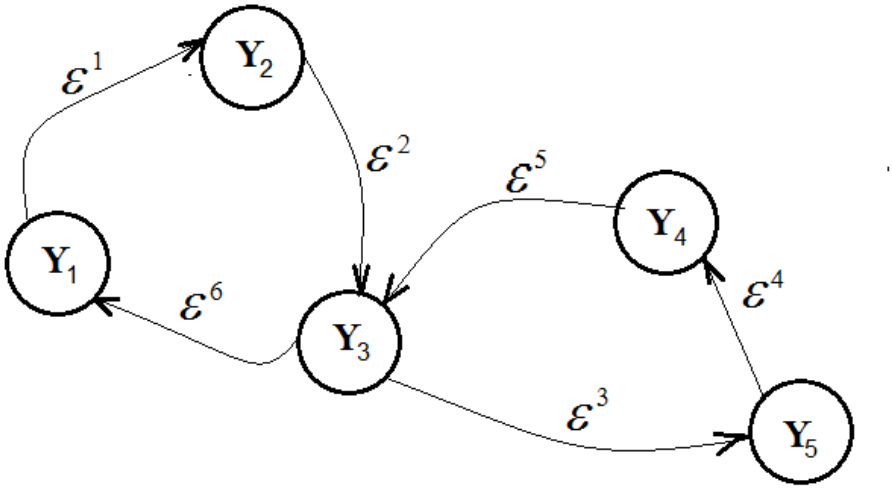


Рис. 2.4. Семіотична модель управління командою IT-проєкту

На рис. 2.4 показано, що модель містить в собі 5 варіантів формальної системи Y_i та правила зміни базових елементів:

Y1 – команда формується;

Y2 – команда визначає індивідуальні та групові цілі;

Y3 – тестування команди;

Y4 – команда формує технічні компетентності;

Y5 – команда формує психологічні навички.

Але команда проєкту є динамічною системою і формальні системи можуть змінюватися. Наприклад, по завершенню етапу формування команди та адаптування один до одного, почнеться етап її становлення та набуття креативних компетентностей. Варіанти формальних систем можуть додаватися нові, а можуть і зникати попередні. Саме ці динамічні перетворення відображаються у змінах до правил . У ситуативній моделі правила не змінюються, а залишаються впродовж усього життєвого циклу проєкту.

З рисунку 2.4 видно, що за рахунок введення правил став можливий перехід від однієї формальної системи до іншої і це пов'язано із зміною ε^i, яке може збігатися з ε^{BE}, ε^R, ε^A, ε^U або з будь-яким їх поєднанням. Тобто, на відміну від знакової семіотичної системи у випадку складної системи, до типу яких належить проєкт, зміну правил прив’яжемо до поточного стану системи. Тоді зміна правил, наприклад, для базових елементів (повноти даних про членів команди проєкту) може мати такий вигляд :

1) визначати лише роль за функціональним обов’язком;

2) визначати прізвище, ім’я, по батькові;

3) визначати прізвисько членів команди.

Другим прикладом аксіоми може бути визначення мети, яку ставлять перед собою претенденти до команди, а потім вже і члени команди:

1) збільшення заробітної плати та зменшення часового навантаження;

2) підвищення рівня власних технічних компетентностей;

3) підвищення рівня власної емпатії;

4) формування загальнокомандних цілей, тощо.

Як було зазначено вище, команда, як система, є динамічною. Будь-якого ідеального стану за досяжності якого робота над командою буде припинена - не існує, тільки закриття проекту. Будь-які зміни, наприклад, додавання нового стану команди, потребує внесення змін у множину базових елементів {BE}. Тоді, набір правил ε^{BE}, по-перше, має дозволяти робити такі зміни, а подруге - має існувати процедура оновленння множини базових елементів.

Правила зміни аксіом має відображати поточний стан речей в системі управління для забезпечення повної ситуації. Наприклад, може скластися ситуація, коли до одного члена команди у конкретний момент може бути застосоване більш одного управлінського впливу: навчання для підвищення технічних компетентностей та психологічних навичок. Така присутність декількох керуючих впливів призведе до небажаного результату.

Для моделі, що приведена на рис. 2.4, інтерпретація змін правил може бути наступною:

ε^1 – система дозволяє визначати та збирати цілі команди;

ε^2 – система дозволяє вносити зміни до тестів в залежності від встановлених цілей команди;

ε^3 – система дозволяє вносити нові аксіоми задля формування психологічних навичок;

ε^4 – система дозволяє вносити нові заходи із формування технічних компетентностей;

ε^5 – правило обмеження одночасного впливу на суб’єкт управління;

ε^6 – система дозволяє змінити загальну кількість членів команди.

2.4 Висновки до розділу

В даному розділі описані клієнт-орієнтовні техніки гнучкого управління ІТ-проєктами: технологія дизайн-мислення, технологія створення

мінімально-життєздатного продукту MVP, методика Proof of concept. Показано, що саме ці техніки є найбільш ефективним середовищем для розробки ІТ-проєктів. Показано, що головним критерієм та складовою успішного завершення ІТ-проєкту є команда проєкту. Тому її стан та розвиток впродовж проєкту напряму впливатиме на показники успішності. За результатами аналізу властивостей подібних команд було запропоновано концептуальну модель інноваційного управління командами ІТ-проєктів.

РОЗДІЛ 3. ІННОВАЦІЙНІ МОДЕЛІ ТА МЕТОДИ УПРАВЛІННЯ КОМАНДАМИ ПРОЄКТІВ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Метод формування інноваційної команди

Стандартний підхід до формування команди проєкту містить у собі аналіз задач, які необхідно виконати для отримання продукту проєкту. Саме кількість цих задач, їх обсяг та тривалість визначають фаховий та кількісний склад команди проєкту. Крім того, методи формування проєктних команд різняться за галузевою направленністю проектів.

Agile practice guide визначає, що ефективні agile-команди, як правило, складаються з трьох-дев'яти осіб, які повинні розміщуватися в єдиному просторі і на 100% повинні бути віддані команді. Agile команда має бути самоврядною, її члени самостійно вирішують, хто і як виконуватиме роботу наступного періоду. Agile-команди замість керівника мають слугуючого лідера, який підтримує свою команду. Така команда обов'язково має бути крос-функціональною, за рахунок цього вона колективно володіє роботою і всі разом мають всі необхідні навички для завершення проєкту. Це формує синергетичний ефект, з допомогою якого можливий функціональний розвиток продукту.

Стандарт з управління проєктами рекомендує такий перелік питань, які далі зможуть сформувати вимоги та критерії для обрання кандидата до команди проєкту.

Перша вимога - це місце фізичного знаходження команди проєкту. Зазвичай кросфункціональні компетентності зможуть проявитися лише за умови, коли команда проєкту разом перебуває в одному місці. Друга вимога - це культурні погляди команди, культура, традиції, ментальність. Наступним є визначення, як здійснюється управління розвитком команди проєкту, за рахунок яких інструментів.

Рис. 3.1. Основні ознаки формування команди

В результаті формування команда має набути наступних ознак (рис. 3.1):

- відкриті комунікації всередині команди;

- спільне розуміння мети проєкту та здобутків від його успішного завершення;

- спільна відповідальність за кінцевий результат;

- довіра між членами команди;

- співпраця всередині команди, генерація ідей;

- адаптивність до середовища та ситуації;

- стійкість команди як можливість швидкого відновлення у разі виникнення проблем або збоїв;

- розширення прав, можливостей та повноважень кожного члена команди задля прийняття рішень щодо способу своєї роботи;

- отримання визнання та вдячності за виконану роботу, що збільшить імовірність продовжувати роботу з більшим натхненням та наснагою.

Таким чином, команда, що формується, має відповідати таким ознакам. Вона має бути:

- самокерованою (без зовнішнього централізованого керування);
- такою, самоорганізується;
- з феноменом групової емпатії;
- мінімально-життєздатною.

Застосування техніки дизайн-мислення до управління інноваційними ІТ-проєктами вимагає виділення додаткового типу ролей, а саме роль емпата. Усі члени команди, що формується, мають володіти здібностями емпатії у більшому чи меншому ступені. Але ця роль має бути додатковою до означених вище. Роль емпата за своїми властивостями входить до категорії «соціальних», але відомо, що для успішного завершення проєкту бажана перевага ролей дії над ролями аналізу та соціальними ролями.

Відомо, що попередній досвід взаємодії членів команди з іншими учасниками проекту (замовником / користувачем / підрядником / постачальником/ спонсором/ ініціаторром тощо) позитивно впливає на результативність команди. У зв’язку із поставленим критерієм мінімізації кількісного складу команди необхідно відстежувати доступність членів команди один для одного та для лідера, щоб мінімізація кількісного складу не призвела до небажаних зворотних процесів від самоорганізації та самокерованості.

Передумовою взаємодії є вірхний розподіл завдань у проєкті, що може вирішуватися через делегування. У цьому випадку знову треба розглянути роль лідера у команді у сенсі контролю ступеня делегування та завантаженості членів команди. Тому можемо стверджувати про необхідність підвищення доступності членів команди один для одного через підвищення значущості лідера.

Інтелектуальний капітал команди - це сумарний набір компетенцій, знань, умінь, навичок, здібностей команди [2, 13]. Інтелектуальний капітал залежить від рівня освіти, таланту, наявності обов'язкових та бажаних

компетенцій, у тому числі: навчально-пізнавальних, для ІТ-проєктів – технічних: знання технічних аспектів, навичок програмування, комунікативних: уміння слухати, співпереживати, емпатувати, експертизи, швидкості та обсягу виконуваної роботи.

Задачу формування мінімально-життєздатної команди ІТ-проєкту [21] сформуємо через залежність між компетентностями проєктної команди за умови мінімізації кількості її членів.

Для команди ІТ-проєкту введемо такі позначення:

– множина кандидатів до проєктної команди ІТ-проєкту:

$$X = \{x_1, x_2, \dots, x_n\},$$

де n – кількість претендентів в проектну команду,

$$1 \le n \le N_{\text{обм}},$$

де Nобм – максимальна кількість претендентів в команду проєктів, яких запрошують, обмежується керівником або замовником проекту;

– множина компетентностей, якими має володіти кожен і-ий претендент:

$$K_i = \{k_{i1}, k_{i2}, \dots, k_{iMi}\},$$

де i=(1,n), Mi – кількість компетенностей, якими володіє і-ий претендент.

Задачу формування команди ІТ-проєкту зведемо до задачі математичного програмування. Кожному претендентові з першої рівності можна поставити у відповідність певний вектор у багатовимірному просторі з координатами.

Кандидати заздалегідь будуть поділені на групи, що відповідають різним психологічним типам особи. Вектор з останнього виразу для кожного претендента можна отримати в результаті його експертного оцінювання.

Процедура створення МЖК актуалізує завдання оптимізації організаційної структури та методів управління командою ІТ-проєкту. Загальний підхід заснований на формуванні команди, яка об'єднана однією метою, здатна досягати мети автономно і злагоджено за мінімальних управлінських впливів.

Тоді задачу формування МЖК сформулюємо наступним чином:

$$\begin{cases} \sum_{i=1}^{n} x_i \to min \\ \sum_{j=1}^{n} E(K_j) \to max, \end{cases}$$

де x_i — приймає значення 1 (якщо претендента зарахували до складу команди) або 0 (якщо претендента не зарахували до складу команди),

$E(K_j)$ — ефективність команди від множини компетеностей K_j, яку можна оцінювати в балах або в кількості успішно виконаних задач згідно плану проекту (в разі застосування даної моделі не на етапі формування команди, а перегляду складу команди впродовж ЖЦП).

Причому

$$1 \leq \sum_{i=1}^{n} x_i \leq n$$

Тобто, задача формування МЖК сформована як набуття командою максимальної сумарної ефективності від компетентностей, якими володіють усі члени команди, за умови мінімізації кількості членів.

Підсумовуючи все сказане вище, тобто всі умови, передумови, обмеження, сформулюємо узагальнений метод формування мінімально-життєздатної команди ІТ-проєкту з інноваційною складовою, яка застосовуватиме технології дизайн-мислення та МЖП у якості провідних технологій розробки програмного продукту. Блок-схема алгоритму методу наведена на рис. 3.2.

Таким чином, представлений метод формування ефективної інноваційної команди ІТ-проєкту.

Така команда буде самокерованою та такою, що само організується та з стратегією Win-Win всередині команди.

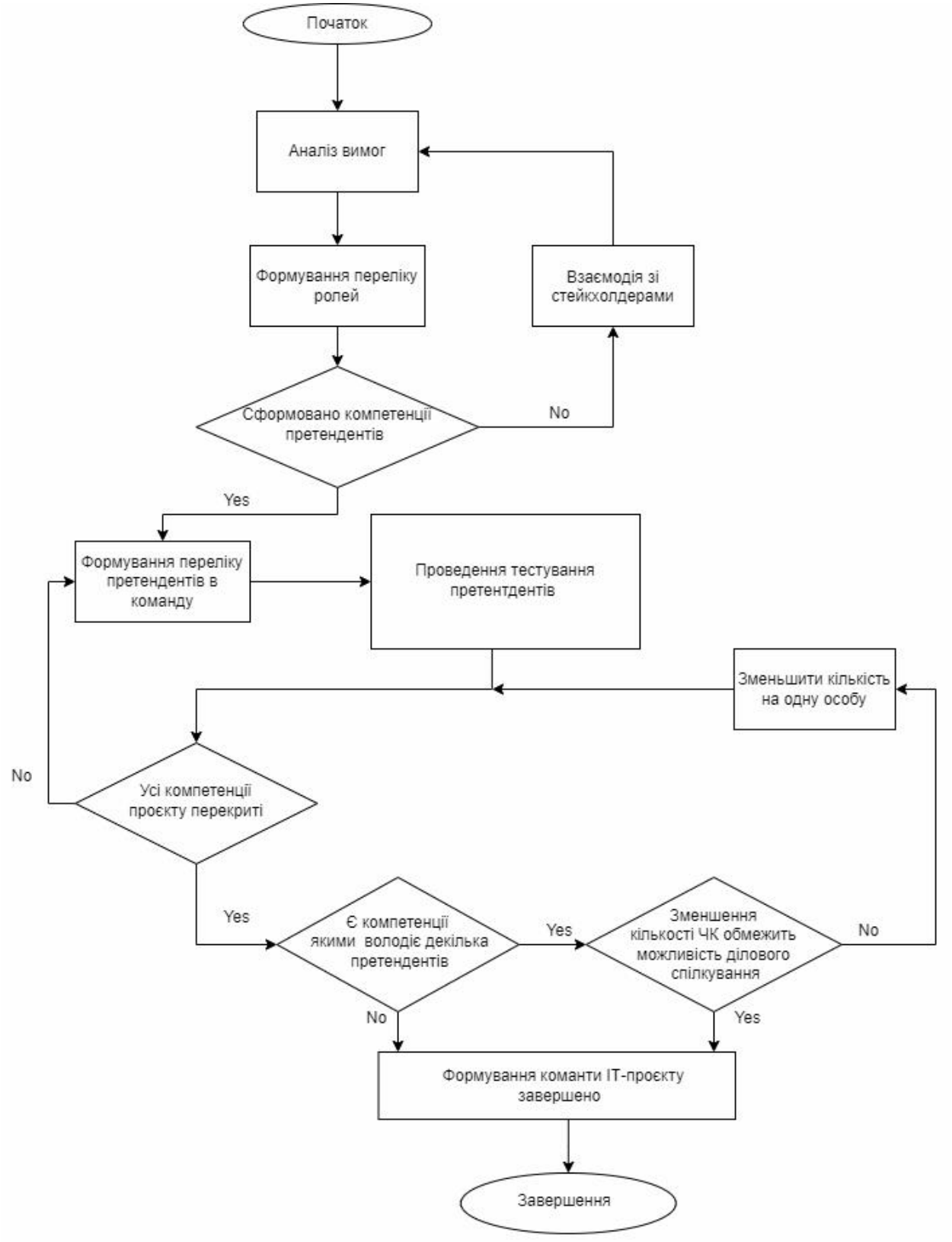


Рис. 3.2. Метод формування команди ІТ-проєкту

3.2 Використання генетичного алгоритму для оптимізації складу команди проєкту розробки ПЗ

Представлений в попередньому підрозділі метод формування команди ІТ-проєкту ставить за мету створення мінімально-життєздатної команди, яка за визначенням має забезпечити перелік необхідних компетентностей та навичок у членів команди, при мінімальній кількості її членів, при цьому такий мінімальний склад не повинен обмежувати час на спілкування як всередині команди, так і зовні (із замовником, кінцевими споживачами продукту проєкту).

Ще одним обмеженням може виступати наявність фінансових ресурсів на оплату праці відповідних категорій працівників. Таким чином, успіх реалізації проекту залежить не тільки від висококваліфікованої професійної проектної команди, але й від однозначного розуміння усіх процесів проекту його зацікавленими сторонами. Таке комплексне бачення має бути інтегроване в єдину систему термінів, понять, проектних дій.

В процесі управління командою проєкту, як складною системою, має місце науково-технічне протиріччя між необхідністю застосування засобів автоматизації складних динамічних об’єктів та процесів та відсутністю універсальних методів та засобів, які б мінімізували вплив рівня кваліфікації і досвіду фахівців-експертів та низки інших суб’єктивних чинників на процес проектування, а потім і на процес управління. Це протиріччя має бути розв’язане через зниження суб’єктивного впливу експертних знань на процес формування та управління командою ІТ-проєкту. Попередня формалізація процесів формування команди виявила багатокритеріальність такої задачі, а функція управління матиме не один, а декілька екстремумів, які не можуть бути зведені до одного - інтегрального. Крім того, функція управління є складною та динамічною, тобто буде змінюватися в часі впродовж ЖЦП.

Метою цього підрозділу є обґрунтування та розробка методу оптимізації кількості членів команди, за рахунок динамічного планування

виконання задач проєкту, яке враховуватиме не лише характеристики самих задач, але й прогнози потреб у нових ресурсах та/або нових компетентностей ресурсів та вартості цих ресурсів із заданими компетентностями. Крім того, як було зазначено вище, обмеженням є обов'язкове виділення часу на спілкування/обговорення членами команди поточної ситуації в проєкті, задля створення можливостей для прояву синергетичного ефекту та групової емпатії.

Мінімізація кількості членів команди проводиться за умови обмежень на крайній термін виконання кожного із завдань з урахуванням технології створення мінімально-життєздатного продукту та з урахуванням створення команди управління проєктом, яка буде незмінною впродовж усього ЖЦ.

Метод формування команди базується на зменшенні кількості членів команди за рахунок розширення діапазону компетентностей кожного з учасників відбору. При цьому обмеженнями є:

1) наявність додаткового часу (не пов'язаного з програмуванням):

- на комунікації всередині команди та із замовником;

- на тренінги/семінари для збільшення професійних компетенцій;

2) умова наявності попереднього досвіду спільної роботи та/або досвіду роботи в команді;

3) фінансові обмеження із боку замовника на оплату робіт проєкту.

Тоді для забезпечення всіх необхідних робіт проєкту доцільно вибрати невелику кількість типів ресурсів, які будуть задіяні в команді. Ці типи ресурсів мають бути обрані з урахуванням типів завдань, які потрібно вирішувати. Ми називатимемо ресурси претендентами, учасниками конкурсного відбору, до того, як вони будуть рекомендовані для участі в проєкті. Після рекомендації та згоди претендента на участь у команді проєкту, називатимемо його членом команди.

Далі процедура відбору буде базуватися на наступних припущеннях: чим більше номер типу претендента, тим менший його компетентнісний діапазон і менше його вартість, а також, якщо завдання може ефективно

вирішуватися і-м претендентом, значить вона не гірше може бути виконана і претендентами з меншими номерами. Ще однією умовою роботи і навіть існування команди ІТ-проєкту є те, що кількісний склад не може сильно змінюватись в межах ЖЦП, особливо в бік зменшення. Мається на увазі, що після закінчення періоду завантаженості даного ресурсу, наприклад, протягом двох тижнів або місяців, він видаляється з команди.

При такому підході однозначно буде втрачено синергетичний ефект (або він і не встигне сформуватися в таких умовах) і вся перевага самоорганізованої та самоврядної мінімально-життєздатної команди буде втрачена.

Разом з тим, технологія створення мінімально-життєздатного продукту передбачає поступове розширення функціоналу продукту в його наступних прототипах, що вимагатиме і збільшення ресурсного забезпечення.

Тому даний підхід враховує необхідність зміни чисельного складу команди, залежно від поточних умов та формування пропозицій щодо прийняття рішень, а самоорганізована та самоврядна команда на основі цих даних прийматиме остаточне рішення. Зважаючи на складність і багатокритеріальність розв'язуваної задачі, її пропонується вирішувати з використанням засобів інтелектуального аналізу, а саме в системі, яка побудована на генетичному алгоритмі.

Генетичні алгоритми зазвичай - це алгоритми глобального пошуку. Вони проводять оптимізацію, спираючися на принципи еволюції та природнього відбору. В роботі був використаний генетичний алгоритм з бінарним кодуванням. Він використовує бінарні рядки як хромосоми - носії генетичної інформації, що представляють можливі рішення.

Цей тип генетичного алгоритму складається з таких основних етапів (рис. 3.3):

1. Ініціалізація популяції. На цьому етапі створюється випадкова популяція бінарних рядків.

2. Оцінка. Кожен бінарний рядок (хромосома) оцінюється за допомогою функції пристосування, яка визначає якість рішення, що він представляє.

3. Селекція. Хромосоми обираються для відтворення на основі їхньої пристосованості.

4. Кросовер (схрещування). Пари обраних хромосом обмінюються бітами для створення нових хромосом.

5. Мутація. Випадкові біти у хромосомах змінюються для внесення випадкових змін.

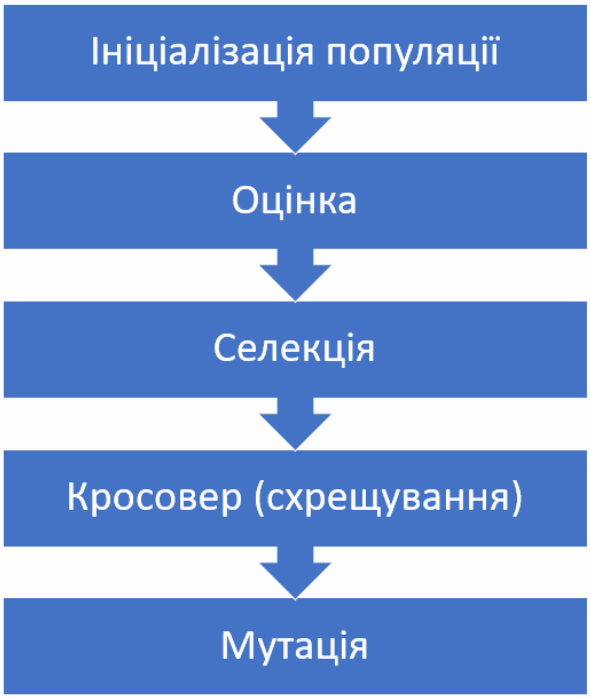


Рис. 3.3. Основні етапи генетичного алгоритму

Генетичний алгоритм з бінарним кодуванням використовується для розв'язання широкого спектру проблем оптимізації, включаючи задачі розміщення, планування, проектування та інших. Він є ефективним і гнучким інструментом, який може адаптуватися до різних типів проблем і обмежень.

Через свою відносну простоту і ефективність, генетичний алгоритм з бінарним кодуванням може бути реалізований за допомогою стандартних бібліотек та інструментів.

Команда проєкту, яка використовує для розробки технології створення мінімальної команди і дизайн-мислення має чотири можливі типи трудових ресурсів:

R1 – зарезервовані ресурси (Reserved Instances) – ресурси з постійними навантаженнями;

Цей чисельний склад є на постійній основі в команді, тому динамічно він змінюватися не може. При цьому можуть бути ситуації, коли ці ресурси не повністю задіяні в запланованих завданнях. В цьому випадку зарезервовані ресурси зможуть виконувати додаткові завдання.

Reserved Instances повинні забезпечувати наявність усіх технічних компетентностей, що вимагаються проєктними роботами, а також обов'язкове вміння емпатувати проблемам замовника, демонструвати психологічну сумісність та мати досвід/можливість роботи в команді.

Наступними за пріоритетом йдуть ресурси **R2** із меншим набором технічних компетенностей. Однак до них, як і раніше, висуваються такі ж вимоги - вміння емпатувати проблемам замовника та психологічну сумісність. Досвід роботи в команді не є обов'язковим.

Ресурси цього мають пріоритети другого порядку, їх вартість менша від вартості ресурсів R1.

Ресурси третього типу (**R3**) повинні володіти технічними компетенностями мінімум двох різних типів та володіти здатністю емпатії. Відповідно цей тип ресурсів має ще нижчі пріоритети і вартість.

Найнижчий пріоритет призначимо ресурсам **R4,** які володіють лише однією технічною компетентністю з необхідного в проєкті діапазону.

Для динамічного управління командою на кожному етапі ЖЦП буде вказано:

– тривалість виконання завдання у дискретах часу, що відповідають інтервалу часу доступності ресурсу;

– мінімальний набір компетентностей, який відповідає цьому завданню;

– крайній термін завершення виконання поточного завдання;

– наявність фінансових можливостей у замовника для покриття витрат на роботу сформованої мінімально життєздатної команди проєкту.

Відповідно до цього, кожному ресурсу при запуску завдання формується вектор стану команди. Поточний час приймемо за Δ, (t+1) – тривалість виконання завдання, (Δ+t+1) – крайній термін завершення завдання.

Початок вектора стану відповідає етапу чи задачі, які виконуються першими. Якщо під час Δ це завдання чи етап (які можуть відповідати, наприклад, одному спринту) було виконано, то вектор стану зменшується на один елемент. Якщо цей етап за час Δ не виконаний, то всі одиниці зсуваються праворуч і вектор коротшає на один елемент зліва.

Структурно, система управління міститиме блок прогнозування, який визначатиме найближчим часом чисельно-компетентнісний склад проєктної команди.

Якщо в момент часу Δ блок прогнозування видав прогноз щодо появи завдання, виконання якого можливо наявними в команді ресурсами, то всі одиниці у векторі стану зсуваються праворуч на інтервали часу, коли прогнозується виконання відповідних етапів цими ресурсами. Осередки ліворуч заповнюються нулями, що означає, що завдання очікує виконання зазначеними вище ресурсами. При скасуванні прогнозу стан вектора відновлюється.

Іншою можливою ситуацією вектора станів може бути склад із одних одиниць, що означає відсутність вільних членів команди. У цьому випадку відбувається пошук претендента за нижчою категорією.

Для формування команди з претендентів ресурсів істотним моментом є рівень запропонованої замовником ціни за необхідний вид ресурсу.

Тоді, власне, процедура формування мінімально-життєздатної команди алгоритмічно відповідатиме мінімізації цільової функції, яка розраховує

необхідну кількість членів команди за необхідними компетентностями від поточного моменту часу на D дискретів вперед, які обмежуються проєкт.

$$K(\Delta) = K_1(\Delta) + K_2(\Delta) + K_3(\Delta) + K_4(\Delta)$$

де К(Δ) – сумарна кількість членів команди, що покривають усі необхідні компетенції та вимоги на дискрет часу Δ;

$К_1(\Delta)$ – сумарна кількість членів команди, які відповідають типу ресурсів R1 на дискрет часу Δ;

$К_2(\Delta)$ – сумарна кількість членів команди, які відповідають типу ресурсів R2 на дискрет часу Δ;

$К_3(\Delta)$ – сумарна кількість членів команди, які відповідають типу ресурсів R3 на дискрет часу Δ;

$К_4(\Delta)$ – сумарна кількість членів команди, які відповідають типу ресурсів R4 на дискрет часу Δ;

Сумарна кількість членів команди типу ресурсу Ri на дискрет часу Δ залежатиме від таких складових:

$$K_i(\Delta) = \sum_{n=1}^{N} \sum_{m=1}^{M} s_m^i(\Delta) \times d_{nm}^i(\Delta) \times T_{nm}^i(\Delta),$$

де i = [1,4] - номер типу ресурсу;

N – кількість робіт проєкту;

M – кількість претендентів у команду проєкту;

$S_m^i(\Delta)$ [1,0] – фінансова доступність забезпечення і - ого типу ресурсу в дискрет часу (1 – ресурс доступний, 0 - недоступний);

$d_{nm}^i(\Delta)$ [1,0] – плановане використання m - ого претендента в n - й задачі в дискрет часу (1 – ресурс запланований для використання, 0 - незапланований);

$T_{nm}^{i}(\Delta)$ [1,0] – планована затримка реалізації n-ї задачі з m-м претендентом в дискрет часу Δ (1 – присутня планова затримка завдання, 0 - відсутня).

Мінімізація чисельного складу команди проєкту проводиться кожен поточний дискрет часу Δ за рахунок вибору відповідних значень коефіцієнтів генетичним алгоритмом при зазначених вище обмеженнях.

Довжина хромосоми для кожного розрахунку визначається кількістю завдань та кількістю відповідних типів ресурсів. Кількість хромосом, що піддавалися мутаціям, а також коефіцієнт мутацій визначається при проведенні експериментів, результати якого будуть представлені в четвертому розділі.

В результаті, даний метод формування команди ґрунтується на генетичному алгоритмі зменшення чисельного складу команди з повним забезпеченням компетентнісного набору ресурсів для успішного завершення проєкту. Процедура враховує динаміку завдань, що запускаються, а також зміну компетентностей і навичок членами команди в межах ЖЦ.

3.3 Практичний приклад формування команди проєкту розробки ПЗ

Нехай проєкт передбачає розробку модуля для системи IoT. Для успішного досягнення цілей проєкту було ідентифіковано склад учасників проєкту, визначені їх ролі, порядок взаємодії учасників проєкту, сформована команда проєкту.

Команда проєкту складається з: проєктного менеджера, тімліда, програмістів (1-3), тестувальників (1-2), інженера та science-інженера.

Склад команди визначено за обсягом робіт, що виконувалися за гнучкою технологією. Простий аналіз складу команди показав наявність 22% управлінців та 78% виконавців (розробників). Подальшу розробку плану

управління командою IT-проєкту виконано з використанням RACI-матриці, як одного з різновидів матриці відповідальності.

На ранніх стадіях життєвого циклу проєкту зазвичай будують укрупнену матрицю відповідальності, більш пізніх - детальну.

Задача	Члени команди								
	Проектний менеджер	Тім лід	Програміст 1	Програміст 2	Тестувальник 1	Програміст 3	Тестувальник 2	Інженер	Science інженер
1.1.1 Узгодження потреби в проекті з замовником	A, R								
1.1.2. Визначення основних параметрів і обсягів робіт		A, R							
1.2. Формулювання основної мети і завдання	A, R								
1.3.1. Формування функціональних вимог		A, R							
1.3.2. Формулювання не функціональних вимог	A, R								
1.4. Формування команди проекту	A, R								
2.1.1. Укладення контрактів з замовником	A, R								
2.1.2. Укладення контрактів з командою	A, R								
2.2.1. Створення календарного плану	A, R								

Рис. 3.4. Частина матриці відповідальності проєкту

В проекті були застосовані класичні значення матриці:

– accountable (відповідальний) – відповідає за виконання завдання, і має право приймати рішення, пов'язані зі способом виконання. За одну задачу проєкту відповідає тільки одна особа;

– responsible (виконавець) – виконує завдання, але не відповідає за вибір методу її вирішення;

– consulted (консультант) – консультує щодо вирішення питань проєкту та веде контроль якості виконання;

– informed (спостерігач) – його інформують про вже прийняте рішення, взаємодія з ним має односторонній характер.

Навіть за таким складом матриця RACI є зручним інструментом візуалізації та проєктування будь-якого процесу управління.

Далі було спроєктовано матрицю, в якої задіяні наступні ролі: R – виконавець, A – відповідальний, C – контролюючий; I – особа, яку інформують про прийняті рішення та поточний стан проєкту. Фрагмент матриці відповідальності подано на рис. 3.4.

З урахуванням доданих ресурсів та відповідальних була сформована сітьова діаграма, яка в управлінні проєктами відображає повний комплекс робіт із встановленими між ними залежністями. Графічно вони виглядають як множина вершин, відповідних завданням, пов'язаних лініями, що представляють взаємозв'язки між роботами. Такий граф ще називають мережею типу вершина-робота чи діаграмою попередження і для даного проєкту він поданий на рис. 3.5.

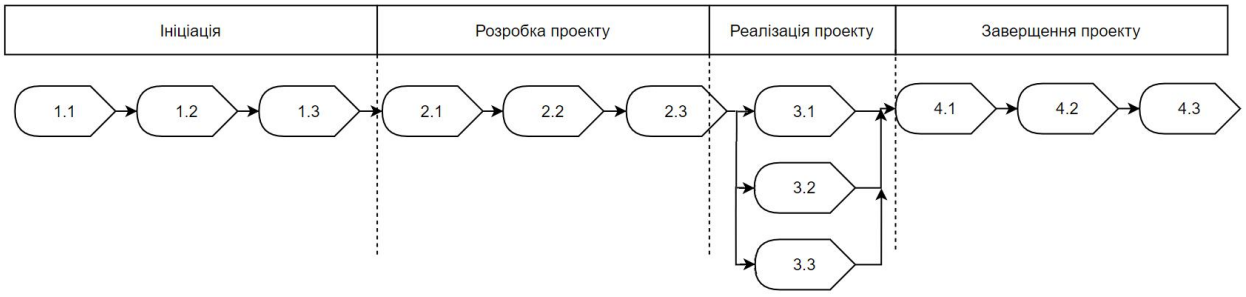


Рис. 3.5. Мережева діаграма проєкту

Подальші пропозиції щодо формування команди управління для команди розробників, ґрунтується на вертикальному та горизонтальному аналізі отриманої матриці (рис. 3.4). Матриця RACI дозволяє наочно контролювати виконання завдань та швидко виявляти перевантажених чи недовантажених виконавців, або таких, які не задіяні у виконанні завдань.

Дешифровка вертикального аналізу означає:

– багато "R". Не завжди людина здатна контролювати настільки багато дій;

– немає порожніх комірок. Слід звернути увагу, що людина навряд чи здатна ефективно працювати над великою кількістю завдань;

– немає "R" або "A". Необхідно звернути увагу на доцільність цієї ролі в команді проєкту;

– багато "A". Варто замислитись над розподілом відповідальності.

Горизонтальний аналіз:

– немає "R". Слід звернути увагу на пошук виконавця завдання, і навпаки, занадто велика кількість «R» може говорити про дублювання виконання;

– багато "A". Може виникнути плутанина через те, що багато виконавців відповідає за одну дію, згідно принципу відповідальності, така ситуація неможлива;

– всі комірки заповнені. Варто звернути увагу, що одним завданням може займатися занадто багато виконавців, що уповільнить процес;

– багато "C". Може зашкодити швидкості реалізації дії чи завдання.

За результатом вертикального аналізу найбільшу кількість R має Проєктний менеджер - 14 робіт на виконання, що показує його велику завантаженість. Також, велику кількість робіт на виконання має Тімлід, що також є високим навантаженням. Це веде до низької продуктивності, що збільшить заплановані строки та бюджет. І хоча, чим більше завантажений робочий час, тим менше простоїв та інших втрат робочого часу, але й тим вище рівень екстенсивного використання праці і, відповідно, продуктивності праці. Проте екстенсивне зростання має чіткі законодавчі межі тривалості робочого дня й тижня.

Результати горизонтального аналізу. Усі завдання мають виконавців. Кожна задача має не більше однієї «A», що свідчить про те, що не буде проблем з перевіркою робіт, а значить й менше часу на вирішення питань між тими, хто затверджує виконання, тобто задачі розподілені вірно.

Таким чином, застосування методики RACI дозволяє не тільки організувати ефективне виконання робіт та поділ відповідальності, а й створити систему відстеження результативності управлінської роботи.

Основні проблеми розподілення робіт у команді, які було виявлено під час аналізу матриці відповідальності – два робітника мають високу завантаженість робіт управляючого характеру (планування попереднього бюджету, оцінка ризиків, формування команди проєкту та інші).

Для забезпечення більш продуктивної та швидкої роботи, пропонується додати у команду проєкту ще одного менеджера, що дозволить проводити деякі роботи паралельно. Фактично, ці зміни ґрунтуються на компетентнісному підходу, який став невід'ємною частиною системи управління персоналом організацій за останні роки.

Також, було виявлено, що керівник проєкту відповідає за усі роботи постановки задач різних областей проєкту. Це веде до того, що менеджер повинен бути залучений в усі тонкощі реалізації. Це також знижує ефективність його роботи, а також може вести за собою недостатню компетентність у багатьох питаннях, та ризики невиконання цих робіт. Для вирішення цієї проблеми запропоновано включити до проєкту команду управління, яка буде безпосередньо залучена до управління проєктом та прийняття управлінських рішень. Менеджери та члени команди (виконавці) звітують перед менеджером проєкту та несуть відповідальність за реалізацію запланованих робіт та результатів (відповідальність може варіюватися від окремого виділеного результату (документу, рішення) до завершеного етапу). До команди управління проєктом було додано п'ять менеджерів: проєктний менеджер, менеджер бекенд розробки, менеджер фронтенд розробки, менеджер апаратної частини та менеджер з машинного навчання (ML розробки).

З урахуванням цього було побудовано нову матрицю відповідальності для команди управління проектом після переформування команди, яку подано на рис. 3.6.

Задача	Ресурси					
	Проектний менеджер 1	Проектний менеджер 2	Менеджер Бекенд розробки	Менеджер фронтенд роз-ки	Менеджер апаратної частини	Менеджер ML розробки
1	2	3	4	5	6	7
1.1.1 Узгодження потреби в проекті з замовником	A, R					
1.1.2. Визначення основних параметрів і обсягів робіт						
1.2. Формулювання основної мети і завдання		A, R				
1.3.1. Формування функціональних вимог						
1.3.2. Формулювання не функціональних вимог	A, R					
1.4. Формування команди проекту		A, R				

Рис. 3.6. Фрагменти матриці відповідальності для команди управління проєктом

Проведений аналіз проєкту з врахуванням матриці відповідальності RACI показав можливість виявляти недоліки сформованої команди IT-проєкту, в плані виконання членами команди проєкту різних типів робіт, за різними компетентностями, що може знижувати продуктивність праці. На основі цього було запропоновано алгоритм формування команди управлінців, які мають володіти високим рівнем емпатії, чисельний склад цієї команди має бути незмінним продовж ЖЦП, а зміна чисельності команди, яку вимагатиме застосування технології MVP, буде проводитися під управлінням саме цієї команди управлінців. Це дало кращі результати по вартості та тривалості виконання проєкту. Нова сформована команда має більшу продуктивність, оскільки канали взаємодії з менеджерами проєкту налаштовані правильно. Таким чином, команда управління проєктом, що додалась до існуючого проекту, сприяла покращенню усіх показників реалізації проєкту. Щодо проєкту, який розглядається як приклад у роботі, було зафіксовано таки показники: зменшення бюджету на 7,3%; тривалості на 14 календарних днів.

Завдяки команді управління проєктом вдалося розпаралелити задачі етапу ініціації та розробки, що відображено у оновленій мережевй діаграмі (рис. 3.7).

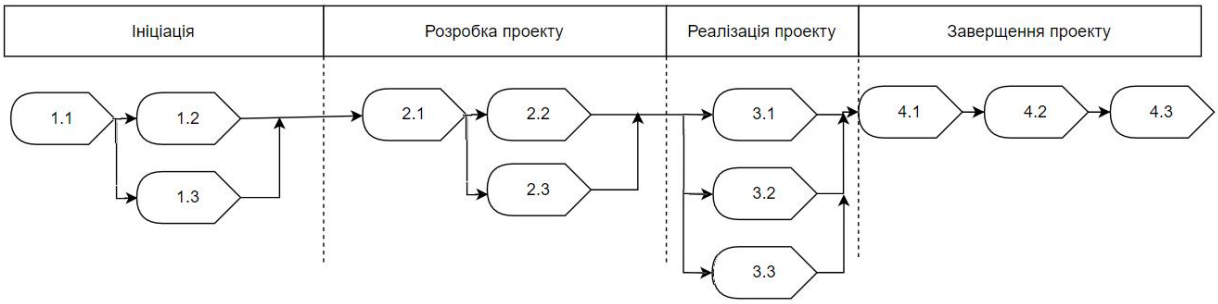


Рис. 3.7. Оновлена сітьова діаграма проєкту

Таким чином, застосування команди управління ІТ-проєктом буде важливим та необхідним для команд, які використовують при розробці: технологію дизайн-мислення та технологію MVP.

В даному випадку, недоліком команди управління проєктом буде збільшення кількості осіб, якщо критерієм була їх мінімізація, або ж збільшення бюджету, якщо він розраховуватиметься за місячною заробітною платою виконавців. Крім того, якщо критерієм оцінки є оптимізація процесів управління командою, застосування компетентнісного підходу, підвищення кваліфікації, то команда управління проєктом, як додаткова організаційна структура проєкту, безумовно має переваги, та рекомендується до застосування.

3.4 Інформаційна технологія інноваційного управління командою розробки ПЗ

Основою системи є генетичний алгоритм, що використовується для розв'язання оптимізаційної проблеми - вибору найкращих (за заданими

критеріями) членів команди для досягнення максимальної ефективності - яка проявляється лише в успішному завершенні проєкту.

Генетичний алгоритм наслідує процеси природного відбору та дозволяє ІТ здійснювати ітеративний пошук оптимальних комбінацій співробітників на основі ряду критеріїв, таких як досвід, навички, вартість, сумісність та емпатія.

Ця система також автоматично враховує складність проєкту, яка розраховується на основі факторів, таких як терміни, потреби у навичках та обмеження бюджету. Визначення складності проєкту є цінним елементом, який дозволяє генетичному алгоритму точніше визначати оптимальні конфігурації команд.

Далі розглянемо технічну структуру системи, її ключові компоненти та особливості. Зазначимо також, що користувачі можуть взаємодіяти з системою та отримувати конкретні результати, що сприятиме ефективному управлінню проєктами та ресурсами. Діаграма акторів ІТ подана на рис. 3.8.

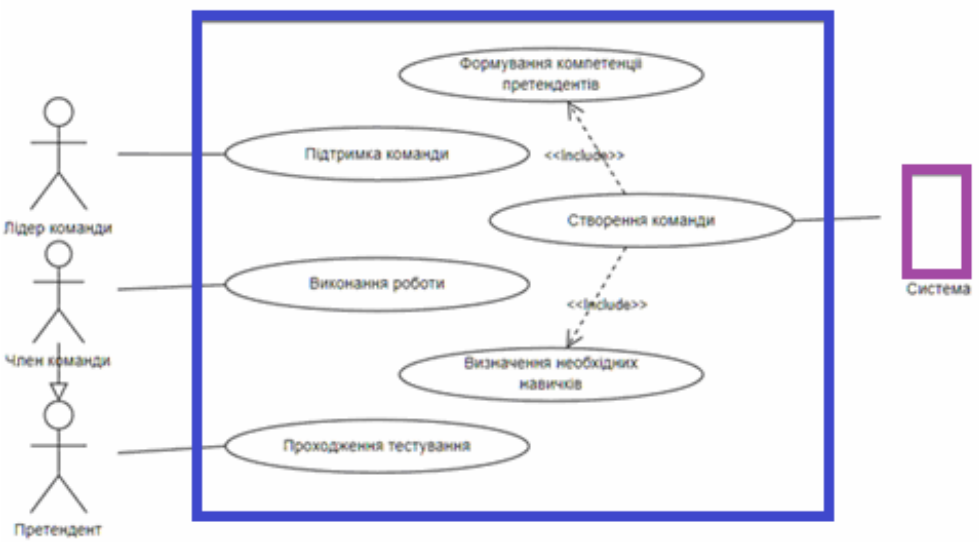


Рис. 3.8. Діаграма учасників інформаційної технології інноваційного управління проєктами розробки ПЗ

Структура бази даних ІТ подана на рис. 3.9.

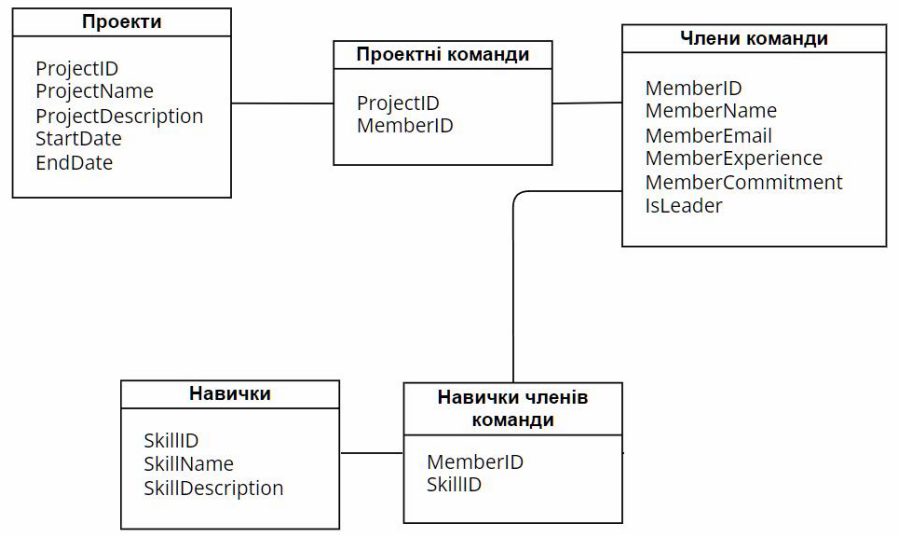


Рис. 3.9. Структура бази даних технології інноваційного управління проєктами

Система оптимізації команд проєктів складається з кількох основних компонентів, які взаємодіють між собою, щоб надати користувачам засоби для створення проєктів, додавання співробітників та оптимізації команд за допомогою генетичного алгоритму. У якості прикладу розроблених процесів операцій ІТ, рис. 3.10 подано діаграму діяльності прецедента, а саме процес відбору претендентів до команди проєкту.

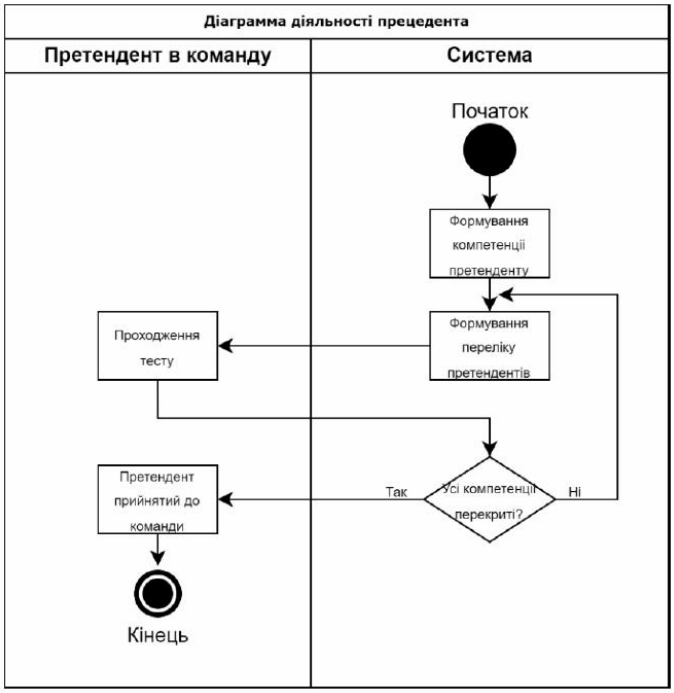


Рис. 3.10. Процес відбору претендентів до команди проєкту

Користувачі взаємодіють з системою через інтерфейс. За допомогою інтерфейсу користувач створює нові проєкти, а саме описує проєкт (рис. 3.11).

Назва проєкту: Інтелектуальна система управління розумним будинком для людей з особливими потребами	
Проблематика	Визначити тип, пакети та джерела даних задля оптимізації системи управління розумним будинком для людей з особливими потребами
Мета проєкту	Мета проекту Підвищити ефективність управління розумним будинком для людей з особливими потребами з метою забезпечення їхньої безпеки
Опис продукту проєкту (до 200 слів)	Інтелектуальна система аналізу даних в системі IoT розумного будинку для людей з особливими потребами
Бачення замовника	Отримання даних для побудови моделей поведінки людей з особливими потребами
Очікування стейкхолдерів	Очікування стейкхолдерів Забезпечення зручних та комфортних умов проживання людей з особливими потребами через елементи та системи IoT
Анотація проєкту (до 200 слів)	Планування та розробка проєкту створення розумного будинку для людей з особливими потребами на основі системи IoT з інтелектуальним аналізом даних. Аналіз наявних датчиків та мереж для збору даних. Визначення структури додаткових датчиків для збору даних що забезпечать плановане ефективне функціонування системи.
Бачення розробників	Планування та реалізація апаратно-програмного комплексу IoT з інтелектуальним аналізом даних. Пошук оптимальної структури даних для забезпечення ефективного функціонування системи. Пошук оптимальної моделі аналізу даних. Розробки системи штучного інтелекту для забезпечення інтелектуального аналізу даних. Розробка та навчання нейронної мережі
Опис технології розробки продукту	Опис технології розробки продукту Використання водоспадної моделі для розробки апаратної частини продукту проєкту. Використання гнучких технологій розробки для створення програмної частини ПП. Застосування технології MVP з прототипуванням та дизайн-мислення.

Ступінь інноваційності ПП. Апаратна частина (від 1 до 10)	Ступінь інноваційності технології розробки проєкту. Апаратна частина (від 1 до 10)	Ступінь інноваційності технології управління проєктом. Апаратна частина (від 1 до 10)
5	4	2

Ступінь інноваційності ПП. Програмна частина (від 1 до 10)	Ступінь інноваційності технології розробки проєкту. Програмна частина (від 1 до 10)	Ступінь інноваційності технології управління проєктом. Програмна частина (від 1 до 10)
8	9	6

Рис. 3.11. Вікно введення для опису параметрів проєкту

Перш за все, система формує перелік вимог до команди проєкту на основі обробки даних про проєкт та про продукт проєкту. Приклад такого рішення приведений на рис. 3.12.

Вимоги до команди проєкту	
Загальні технічні	
1	Аналітик – 1
2	Архітектор – 1
3	Си # Сеньор – 1
4	Си # Мідл – 2
5	Си # Джун – 3
6	Тестувальник – 1
7	Дизайнер – 1
Загальні (ступінь від 1 до 10) (для всіх членів)	
8	Вміння працювати в команді (від 1 до 10)
9	Досвід роботи в команді проєкту (від 1 до 10)
10	Досвід роботи з конкретними особами в команді проєкту (від 1 до 10)
11	Необхідний рівень емпатії (від 1 до 10)
12	Володіння декількома компетенціями/навичками (від 1 до 10)

Рис. 3.12. Перелік вимог до команди проєкту

3.5 Висновки до розділу

Отже, в цьому розділі представлено методи управління командою ІТ-проєкту з інноваційною складовою, які спираються на технологію дизайн-мислення та концепцію створення мінімальної команди. Потреба у вдосконаленні та розробки нових методів з’явилася тому, що команди, які потрібні для розробки проєктів з інноваційною складовою, мають бути самокерованими та саморганізованими. Це суттєво відрізняє їх від класичних методів формування та управління командами ІТ-проєктів, в яких головними чинниками та критеріями успішності були: спланованість, керованість, підпорядкованість.

Також обґрунтовано застосування генетичного алгоритму в процесах формування команди для інноваційного проєкту. Показано, що задача формування є багатокритеріальною, а в рішенні відсутній головний критерій, який зазвичай вважається критерієм успішності проєкту. Тому, особа що приймає рішення, має орієнтуватися на власний досвід, уміння та компетенції. Представлено результати розробки інформаційної технології інноваційного управління командою проєкту розробки ПЗ.

ВИСНОВКИ

В магістерській роботі розглянуто інноваційні моделі, методи та методології управління розробкою програмного забезпечення. Визначено сутність та особливості сучасних проєктів розробки програмного забезпечення, досліджено специфіку управління проєктами. Подано результати аналізу управління командами, що використовують гнучкі технології розробки програмного продукту, та сучасні інноваційні технології управління ІТ-проєктами, такі, як технологія дизайн-мислення та концепція створення MVP. Досліджено, що оточення ІТ-проєктів має бути клієнт-орієнтованим та ціннісно-орієнтованим.

Проведений аналіз показав, що головним критерієм та складовою успішного завершення ІТ-проєкту є його команда. За результатами аналізу властивостей подібних команд було запропоновано та розроблено концептуальну модель інноваційного управління командою проєкту, яка дає змогу сформувати гнучку, мінімальну команду управління ІТ-проєктами.

Розроблено семіотичну модель управління командою ІТ-проєкту та формальну ситуаційну модель, що слугує їй базисом. Представлена модель на відміну від існуючих, дозволяє зміну набору правил, через базові елементи, семантичні та синтаксичні правила, аксіоми, що дозволить гнучко вносити зміни як в склад команди проекту, так і в процес прийняття рішень щодо управління командою на протязі ЖЦ.

Представлено метод інноваційного управління командою ІТ-проєкту який включає продукування професійного зростання членів команди; розвиток емоційного інтелекту та досягнення спільної командної мети, яка включає в себе індивідуальні цілі кожного члена команди. Розроблено інформаційну технологію інноваційного управління командою проєкту розробки ПЗ, за допомогою якої менеджерам проєкту пропонується проєктне рішення щодо формування команди ІТ-проєкту.

ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Danchenko O., Amalahu V., Zarntsky S., Bielova O. IT projects: features, characteristics and classification. Управління проектами: стан та перспективи : Матеріали ХУП Міжнародної науково-практичної конференції. Миколаїв : Видавець Торубара В.В., 2021. 130 с., с. 107-108.
2. Данченко О.Б., Тесленко П.О. Аналіз сучасних визначень ІТ-проєктів. Управління проєктами у розвитку суспільства. Тема: «Управління проєктами в умовах пандемії COVID-19»: тези доповідей. Київ : КНУБА, 2021. - С.100-104.
3. Коваленко О.О., Денисюк А.В., Бажан В.М. Порівняльний аналіз методологій розробки ІТ продукту. Вінницький національний технічний університет. URL: https://conferences.vntu.edu.ua/mdex.php/a-fitki/a-fitki-2021/paper/ download/12199/10302.
4. Bedrii D.I., Semko I., Krylov V. IT-projects in power engineering. Project, Program, Portfolio Management. Proceccing of the Fourth International Scientific and Practical Conference 06-07 December 2019. Book 2. Odesa, ONPU, 2019. p. 16-20. ISSN 2522-9435.
5. Богославец А.А. Классифікація ІТ-проєктів. Комунальне господарство міст. Х.: 2014. Випуск 118. С. 56-59.
6. Глушенкова А.А. Особливості управління інноваційними проєктами у сфері телекомунікацій та інформатизації. Економіка. Менеджмент. Бізнес, 2015. №4 (14). С. 72 - 77.
7. Катренко А.В. Управління ІТ-проєктами. [Стандарти, моделі та методи управління проєктами]: Львів: «Новий Світ-2000», 2011. 550 с.
8. Robert T. Futrell. Quality Software Project Management. Prentice Hall, 2002, 1639 p. ISBN 8177587536, 9788177587531.
9. Murray Cantor. Object-Oriented Project Management with UML. Publisher: John Wiley & Sons, Inc, 1998. - 388 p. ISBN: 0471253030.

10. Walker Royce. (2000) Software project management: a unified framework. The Addison-Wesley object technology series. Includes bibliographical references and index. ISBN 0-201-30958-0.
11. Dr. Winston W. Royce. Managing the development of large software systems. Proc. IEEE WESCON, Aug 1970. URL:http://www-scf.usc.edu/~csci201/lectures/ Lecture11/royce1970.pdf.
12. Прокопенко Т.О., Поволоцький Я.О. Система критеріїв оцінювання ефективності проєктів галузі інформаційних технологій. Вісник Черкаського державного технологічного університету, 2022. № 4. С.23-30. DOI: https://doi.org/10.24025/2306-4412.4.2022.271448.
13. Підходи до управління програмними проєктами у SWEBOK V3.: URL: https://www.researchgate.net/publication/316493834.
14. SWEBOK V3. Guide to the Software Engineering Body of Knowledge. Version 3.0. URL: https://cs.fit.edu/~kgallagher/Schtick/Serious/ SWEBOKv3.pdf.
15. Software Extension to the PMBOK® Guide Fifth Edition. Project Management Institute. Publ., 2013. www.PMI.org. 240 p.
16. Sarah K. White and Lynn Greiner. (2022). What is ITIL? Your guide to the IT Infrastructure Library URL: https://www.cio.com/article/272361/ infrastructure-it-infrastructure-library-itil-definition-and-solutions.html.
17. Scott Berkun (2005). The Art of Project Management. Released April 2005. Publisher: O'Reilly Media, Inc. ISBN: 9780596007867. URL: https://www.oreilly.com/ library/view/the-art-of/0596007868.
18. Данченко О.Б. Інструменти управління ІТ-проєктами з інноваціями. Збірка тез Vm Міжнародної НТК «Інформатика. Культура. Технології» ІКТ-2021. Одеса: ІКС, 2021. С. 84-86. URL: http://ics_conf.tilda.ws/ict_eng.
19. Robert C. Martin (2019) Clean Agile. Back to Basics. Boston. 2020. Pearson Education, Inc. 1st edition, 240 p. ISBN 0135781868.

20. Складові системи планування проекту. Електронний ресурс: https://buklib.net/books/23851/.
21. Software Engineering Body of Knowledge (SWEBOK). URL: https://www.computer.org/education/bodies-of-knowledge/software-engineering.
22. A Guide to the Project Management Body of Knowledge (PMBOK® Guide). Sixth Edition. USA. PMI, 2017. 756 p.
23. Управління IT проєктами. URL: http://dspace.wunu.edu.ua /retrieve/ 19638/%D0%9B%D0% B5%D0%BA%D1%86%D1%96%D1%97.pdf
24. Дослідження методів підтримки прийняття рішень при удосконаленні процесу розробки IT-проекту. URL: https://openarchive.nure.ua/ bitstream/document/19072/1/2021_M_IU S_Potehin_SV.pdf 6.
25. A Guide to the Project Management Body of Knowledge (PMBOK® Guide- Sixth Edition / Agile Practice Guide Bundle (HINDI). Project Management Institute. Publ., 2017. www.PMI.org. 115 p.
26. Schwaber K. and J. Sutherland The Scrum Guide. The definitive Guide to Scrum: The Rules of the Game. 2017, 19 p. URL: https://scrumguides.org/ docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf.
27. Schwaber, Ken; Beedle, Mike (2002). Agile software development with Scrum // Prentice HallPTR Upper Saddle River, NJ, USA, 2001.
28. J. Sutherland. (2019) Scrum. A revolutionary approach to building teams, beating deadlines and boosting productivity. Random House Business ISBN: 9781847941107.
29. Прокопенко Т.О., Підкуйко О.І. Розробка графоаналітичної моделі ситуацйного управління проектом в умовах SCRUM у сфері інформаційних технологій. Вісник Черкаського державного технологічного університету, 2022. №2. С 4-10. DOI: https://doi.Org/10.24025/2306-4412.2.2022.261704.
30. Семко І.Б., Кійко С.Г. Огляд сучасних методологій управління командами IT-проєктів. Управління розвитком складних систем. К.:

КНУБА, 2020. №43. С. 100-104. DOI: 10.32347/2412-9933.2020.43.60-66. URL: http://mdcs.knuba.edu.Ua/ article/view/219835.

31. Hamilton T. Scrum vs Kanban - Difference Between Them. URL: https://www. guru99.com/scrum-vs-kanban.html.
32. Джеймс Вумек, Дэниел Джонс. Ощадливе виробництво. Фабула, 2018. 448 с. ISBN978-617-09-3892-3.
33. Zanora V., Momot S., Bedrii D., Fonar L. Conflict management in enterprise development project teams. Academy review. Dnipro: Alfred Nobel Univ, 2023. Is. 1(58). P. 187-204. ISSN 2074-5354 (print), ISSN 2522-9745 (online). DOI: https://doi.org/10.32342/2074-5354-2023-1-58-14.
34. Новохацька Д.В. Особливості та проблеми реалізації ІТ-проєктів в Україні. Вісник ЧДТУ, 2016. № 2. С.72-77.
35. Laing Samantha and Hryvs Karen (2013). Growing agile: a coach's guide to training Scrum. Publisher: Growing Agile; 1st edition, 195 p. ASIN : B00E8DFMT4.
36. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) and the Standard for Project Management. Seventh Edition. USA. PMI, 2021. 274 p.
37. Івченкова О. Ю., Лях А. О. Аналіз моделей і методів розподілу трудових ресурсів в управлінні реалізацією портфеля ІТ-проектів. Вісник економічної науки України, 2016. № 2. С.87-91.
38. Шашкова, Н., Фадєєва, І., Казакова, Т. Управління проєктами в ІТ сфері: застосування гнучких методологій. Scientific Notes of Lviv University of Business and Law, 2021. 28. С. 166-172. Retrieved from URL: https: //nzlubp. org.ua/index.php/j ournal/article/view/402.
39. Bushuyeva N., Bushuiev D., Bushuieva V. Agile leadership of managing innovation projects. Сучасний стан наукових досліджень та технологій в промисловості, 2019. № 4. С. 77-84.

40. Jalote Pankaj (2002) Software project management in practice. Boston: Addison-Wesley. ISBN 9780201737219.
41. Добровська Л.М., Аверьянова О.В. Управління ІТ-проєктами в Microsoft Project: Комп'ютерний практикум URL: навчальний посібник для студентів спеціальності 122 "Комп'ютерні науки" для всіх спеціалізацій. Київ: КПІ ім. Ігоря Сікорського, 2020 - 152 с.
42. Використання програми Microsoft Project, Planner, To Do або програми "Завдання" в Teams. URL: https://support.microsoft.com/uk-ua/office.
43. ProjectLibre: програма управління проєктами з відкритим кодом. URL: https://www.linuxadictos.com/uk/projectlibre-un-programa-para-la-gestion-de- proyectos-de-codigo-abierto. html.
44. Бодненко Д. М., Котик В. В., Мишковець С. С., Соколовська Т. В. Використання сервісу trello в професійній діяльності. URL : https://elibrary.kubg.edu.ua/id/eprint/32911/1/1_Bodnwnko_Kotyk_Myshkovec_Sokolovska_IT-20.pdf.
45. Krigsman. M. CIO Playbook: IT value and the digital mindset URL : https://www.zdnet.com/artide/cio-playbook-it-value-and-the-digital-mindset.
46. Hasso Plattner Design Institute at Stanford University. URL : dschool.stanford.edu.
47. Ertel, C., Solomon, L. K. Moments of impact: how to design strategic conversations that accelerate change. New York: Simon & Schuster, 2014. 273 p.
48. Liedtka, J., Ogilvie, T. Designing for growth: A design thinking toolkit for managers. New York: Columbia University Press, 2011. 256 p.
49. Vasilieva, E. Developing the creative abilities and competencies of future digital professionals. Automatic Documentation and Mathematical Linguistics, 2018. No. 52 (5). P. 248-256.
50. A Guide to the Project Management Body of Knowledge (PMBOK® Guide). Sixth Edition. USA. PMI, 2017. 756 p.