**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**
**НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ**

**Кафедра комп'ютеризованих систем управління**

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри

_____Олександр ЛИТВИНЕНКО

«___» _____2023 р.

# КВАЛІФІКАЦІЙНА РОБОТА

**(ПОЯСНЮВАЛЬНА ЗАПИСКА)**

**ЗДОБУВАЧА ВИЩОЇ ОСВІТИ**
**СТУПЕНЯ «МАГІСТР»**

Тема: Програмний модуль прогнозування часу виконання проєктів в компанії

Виконавець: Денис САВЧЕНКО

Керівник: Ольга КРАВЧЕНКО

Нормоконтролер: Євгеній ТУПОТА

**Київ 2023**

**НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ**

Факультет комп'ютерних наук та технологій

Кафедра комп'ютеризованих систем управління

Спеціальність 123 «Комп'ютерна інженерія»

Освітньо професійна програма «Системне програмування»

Форма навчання денна

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____Олександр ЛИТВИНЕНКО

«_____»_____2023 р.

**ЗАВДАННЯ**

**на виконання кваліфікаційної роботи**

Савченко Денис Русланович

**1.Тема роботи:** «Програмний модуль прогнозування часу виконання проєктів в компанії»

затверджена наказом ректора від «28» серпня 2023 року № 1494 /ст.

**2. Термін виконання роботи:** з 02.10.2023 до 31.12.2023

**3.Вихідні дані до проєкту (роботи):** системна програмна документація, документація до програмного забезпечення управління проєктами

**4.Зміст пояснювальної записки (перелік питань, що підлягають розробці):**

1) аналіз процесу реалізвції проєктів програмного забезпечення,
2) методи групової динаміки в глобальній розробці програмного забезпечення,
3) реалізація програмного модуля прогнозування часу виконання проєктів.

**5.Перелік обов'язкового графічного матеріалу:**

1) ієрархія проєктної групи;
2) вікно формування інформації про проєкт;
3) діаграма класів клієнтської частини;
4) діаграма класів серверної частини;
5) схема алгоритму роботи з програмою керівника компанії.

**6. Календарний план-графік**

| № п/п | Етапи виконання кваліфікаційної роботи | Термін виконання етапів | Примітка |
|---|---|---|---|
| 1 | Провести аналіз літератури за темою кваліфікаційної роботи та аналіз існуючих систем | 02.10.2023-12.10.2023 | |
| 2 | Оформлення розділу 1 | 13.10.2023-29.10.2023 | |
| 3 | Оформлення розділу 2 | 30.10.2023-06.11.2023 | |
| 4 | Оформлення розділу 3 | 07.10.2023-14.11.2023 | |
| 5 | Провести налаштування програмних засобів на сервері | 15.10.2023-26.11.2023 | |
| 6 | Пройти нормоконтроль | 27.10.2023-11.12.2023 | |
| 7 | Підготувати презентацію | 12.12.2023-18.12.2023 | |
| 8 | Оформити супрводжувальну документацію | 19.12.2023-23.12.2023 | |

**7. Дата видачі завдання** «02» жовтня 2023 р.

Керівник кваліфікаційної роботи_____Ольга КРАВЧЕНКО_____
(підпис)

Завдання прийняв до виконання_____Денис САВЧЕНКО_____
(підпис студента)

# РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи «Програмний модуль прогнозування часу виконання проєктів в компанії» складається з: 81 с., 20 рис., 2 таблиці, 27 літературних джерел, 1 додаток.

РОЗРОБКА ПЗ, ПРОЄКТ ПЗ, КОНТЕНТ ПЗ, АНАЛІЗ ПРОЄКТУ, УПРАВЛІННЯ ПРОЄКТУВАННЯМ, ТЕСТУВАННЯ

Мета дослідження – створення програмного модуля, який здатний аналізувати історичні дані про виконання завдань та проєктів та на їх основі робити прогнози часу виконання подібних завдань у майбутньому, що сприятиме кращому управлінню ресурсами та забезпечить вчасне завершення проєктів.

Об'єкт дослідження – процес прогнозування часу виконання завдань та проєкту в цілому в компаніях.

Предмет дослідження – програмний модуль прогнозування часу виконання проєктів в компанії.

Прогнози припущення щодо розвитку об’єкта дослідження – створення робочого зразка програми та використання його при розробці *web*-додатків розподіленими у просторі командами розробників.

Апробація результатів була проведена на міжнародній науковій конференції “Інтелектуальні технології лінгвістичного аналізу” (м. Київ, 23-24 жовтня 2023 р.), науковій практичній конференції “Сучасні тенденції розвитку системного програмування” (м. Київ, 23-24 листопада 2023 р.).

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

*API* – *Application Programming Interface* (програмний інтерфейс додатку)

*CGI* – *Common Gateway Interface* (загальний інтерфейс об’єднання)

*CRM* – *Customer Relationship Management*

*DNS* – *Domain Name Service* (сервер доменних імен)

*FTP* – *File Transfer Protocol* (протокол передачі файлів)

*GSD* – *Global Software Development*

*SQL* – *Structured Query Language* (структурована мова запитів)

ГД – глобальна динаміка

# ВСТУП

У сучасному світі інформаційних технологій (ІТ) розробка програмного забезпечення та управління ІТ-проєктами стали необхідністю для більшості компаній та організацій. За останні роки ринок ІТ надзвичайно динамічний, і це призводить до зростання складності та обсягів проєктів, а також до появи нових викликів та проблем у сфері управління цими проєктами. Однією з ключових проблем є прогнозування часу виконання ІТ-проєктів.

У сучасному світі інформаційних технологій (ІТ) розробка програмного забезпечення та управління ІТ-проєктами стали необхідністю для більшості компаній та організацій. За останні роки ринок ІТ надзвичайно динамічний, і це призводить до зростання складності та обсягів проєктів, а також до появи нових викликів та проблем у сфері управління цими проєктами. Однією з ключових проблем є прогнозування часу виконання ІТ-проєктів.

Прогнозування часу виконання ІТ-проєктів є надзвичайно важливою задачею, яка впливає на багато аспектів підприємництва та управління. Від точності та надійності прогнозів залежить вчасне завершення проєктів, використання ресурсів компанії та задоволеність клієнтів. Неточні або недостовірні прогнози можуть призвести до перевищення бюджету, прострочки проєкту та втрати довіри замовника.

Наукова складова цієї роботи полягає у розробці та вдосконаленні системи прогнозування часу виконання ІТ-проєктів. Для досягнення цієї мети використовуються передові методи та технології з областей машинного навчання, статистики, оптимізації та аналізу даних. Система, яку розглядає ця робота, створена з метою підвищення точності та надійності прогнозів часу виконання ІТ-проєктів, що є важливим внеском у покращення управління проєктами та ефективності ІТ-компаній.

Розробка програмних систем і проєктний менеджмент в сфері інформаційних технологій (ІТ) постійно розвиваються і стають все складнішими завданнями. Важливим етапом у процесі розробки проєкту є прогнозування часу виконання завдань та проєкту в цілому. Неточні або недостовірні прогнози можуть призвести до перевищення бюджету, прострочки проєкту та незадоволеності клієнтів. Тому розробка програмного модуля для прогнозування часу виконання проєктів є актуальною задачею, яка допомагає покращити управління проєктами в ІТ-компаніях.

Прогнозування часу виконання ІТ-проєктів є надзвичайно важливою задачею, яка впливає на багато аспектів підприємництва та управління. Від точності та надійності прогнозів залежить вчасне завершення проєктів, використання ресурсів компанії та задоволеність клієнтів. Неточні або

недостовірні прогнози можуть призвести до перевищення бюджету, прострочки проєкту та втрати довіри замовника.

Актуальність цієї проблеми стає ще більш очевидною в контексті зростання обсягів даних, змінності умов, розподіленості команд розробників, а також зі зростанням числа проєктів, які виконуються одночасно. Управління такими складними завданнями стає справжньою викликом для керівництва IT-компаній та їхніх розробників.

Науковою новизною даної роботи є порівняльний аналіз різних факторів глобальної розробки програмного забезпечення, проведений якісними та кількісними методами дослідження.

Основні завдання дослідження включають:

1. Аналіз та обробка історичних даних про виконання проєктів в IT-компанії.
2. Розробка математичних моделей та алгоритмів для прогнозування часу виконання проєктів.
3. Інтеграція розробленого програмного модуля з існуючими системами управління проєктами.
4. Проведення тестів та експериментів для оцінки ефективності та точності прогнозування.

Мета дослідження – створення програмного модуля, який здатний аналізувати історичні дані про виконання завдань та проєктів та на їх основі робити прогнози часу виконання подібних завдань у майбутньому, що сприятиме кращому управлінню ресурсами та забезпечить вчасне завершення проєктів.

Об'єкт дослідження – процес прогнозування часу виконання завдань та проєкту в цілому в компаніях.

Предмет дослідження – програмний модуль прогнозування часу виконання проєктів в компанії.

**Апробація результатів** була проведена на науковій практичній конференції “Сучасні тенденції розвитку системного програмування” (м. Київ, 23-24 листопада 2023 р.).

За результатами опубліковано тезу доповідей: Савченко Д.С. Програмний модуль прогнозування часу виконання проєктів в *IT*-компанії // Тези доповідей наук.-практ. конф. “Сучасні тенденції розвитку системного програмування” (23-24

листопада 2023 р.). – К.: НАУ, 2023. – С. 69.

# РОЗДІЛ 1
# АНАЛІЗ ПРОЦЕСУ РЕАЛІЗАЦІЇ ПРОЄКТІВ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## 1.1. Аналліз особливостей розробки програмного забезпечення

У сучасному світі програмне забезпечення (ПЗ) відіграє надзвичайно важливу роль у різних галузях, включаючи інформаційні технології, медицину, фінанси, освіту та багато інших. Розробка програмного забезпечення є складним і багатоетапним процесом, який включає в себе багато технічних, організаційних та людських аспектів. В цьому підрозділі ми розглянемо основні проблеми, що можуть виникати під час розробки програмного забезпечення, і шляхи їх вирішення.

Однією з найбільших проблем, з якими стикаються розробники програмного забезпечення, є складність проектів. Програмні проекти можуть бути великими та складними, з великою кількістю функціональності, завдань та вимог. Ця складність може призводити до збоїв у розробці, затримок та перевищення бюджету.

Вирішення: Для вирішення цієї проблеми розробники повинні ретельно планувати проекти, визначати реалістичні терміни та бюджети, використовувати методи проектного управління, такі як методології Agile або Waterfall. Також важливо ретельно документувати вимоги та використовувати сучасні інструменти розробки та тестування.

Часто вимоги до програмного забезпечення можуть змінюватися в процесі розробки через різні причини, такі як зміна бізнес-потреб, виявлені помилки або нові вимоги користувачів. Це може призводити до збільшення затрат і затримок у розробці.

Вирішення: Для вирішення цієї проблеми важливо мати ефективні системи управління вимогами та контролю змінами. Також корисно використовувати

підходи до розробки програмного забезпечення, які сприяють гнучкості та адаптації, такі як Agile.

Якість програмного забезпечення є критично важливою, оскільки низька якість може призводити до виявлення помилок після випуску та негативно впливати на користувачів.

Вирішення: Для вирішення проблеми якості ПЗ, розробники повинні використовувати тестування, контроль якості та інші методи перевірки програмного забезпечення. Також важливо враховувати якість від початкових етапів розробки і використовувати кращі практики управління якістю.

Ефективна комунікація та співпраця між членами команди розробки, клієнтами та іншими стейкхолдерами є важливими для успіху проектів ПЗ. Недоліки у цих аспектах можуть викликати непорозуміння, невдачі та конфлікти.

Вирішення: Для вирішення проблеми комунікації і співпраці важливо створити ефективну комунікаційну стратегію, використовувати спеціалізовані інструменти для спільної роботи та спілкування, а також підтримувати відкрите та довірливе середовище у команді.

Програмна розробка є складним та важливим процесом, і вона стикається з різними проблемами. Однак застосування правильних методів та підходів може допомогти вирішити ці проблеми та забезпечити успіх проектів програмного забезпечення. Зазначені вище проблеми є лише частиною викликів, які можуть виникнути, і вирішення їх вимагає багато уваги, ресурсів та професіоналізму від розробників та команди.

## 1.2. Аналіз поняття життєвий цикл у розробці програмного забезпечення

Життєвий цикл програмного забезпечення (ЖЦ ПЗ) є однією з ключових складових процесу реалізації проєктів у сфері розробки програмного забезпечення. Він представляє собою набір фаз та етапів, які здійснюються від початку розробки ПЗ до його завершення та обслуговування. В даному підрозділі буде проведений аналіз ЖЦ ПЗ, розглянуті основні фази та їх характеристики, а

також обговорені сучасні методи управління ЖЦ ПЗ та їх вплив на результативність проєктів.

Життєвий цикл програмного забезпечення

ЖЦ ПЗ може бути поділений на кілька основних фаз, кожна з яких включає в себе специфічні завдання та процеси. Основні фази ЖЦ ПЗ включають:

1. Аналіз та планування (Analysis and Planning): Ця фаза визначає основні цілі проєкту, визначається обсяг та вимоги до ПЗ, розробляється план проєкту. Аналіз вимог дозволяє зрозуміти, яким має бути результат проєкту.

2. Проектування (Design): На цьому етапі визначається архітектура ПЗ, вибираються технології та інструменти розробки. Результатом цієї фази є деталізований план програмного продукту.

3. Розробка (Implementation): Фаза розробки включає у себе написання, тестування та оптимізацію коду програми. Це найбільш відомий етап для програмістів, де вони перетворюють плани та дизайн у функціональний код.

4. Тестування (Testing): Після розробки ПЗ проходить черговий етап перевірки, де виявляються помилки, проводяться тести на відповідність вимогам та якості.

5. Впровадження (Deployment): На цьому етапі програмне забезпечення впроваджується в реальне середовище та починає використовуватися кінцевими користувачами.

6. Підтримка та обслуговування (Maintenance): Після впровадження ПЗ продовжує підтримуватися, виправляються помилки, вдосконалюються функції та реагується на нові вимоги.

Основні характеристики фаз ЖЦ ПЗ

- Залежність від фази: Кожна фаза ЖЦ ПЗ залежить від попередньої і наступної. Це означає, що неправильні рішення або помилки на ранніх етапах можуть призвести до серйозних проблем на пізніших етапах.

• Ітеративність та інкрементальність: У сучасних методологіях розробки ПЗ використовуються ітерації та інкременти, що дозволяють покращувати та доповнювати програмне забезпечення послідовно.

• Керованість та контроль: Кожна фаза ЖЦ ПЗ підлягає керуванню та контролю з метою забезпечення виконання завдань в обумовлені строки та бюджетом.

Сучасні методи управління ЖЦ ПЗ

Сучасні підходи до управління ЖЦ ПЗ включають у себе Agile, DevOps, Lean та інші. Ці методи дозволяють зменшити час розробки, підвищити якість програмного забезпечення та забезпечити більшу гнучкість у відповіді на зміни вимог.

Вплив ЖЦ ПЗ на результативність проєктів

Правильний вибір та ефективне управління ЖЦ ПЗ може суттєво покращити результативність розробки програмного забезпечення. Ітеративність, контроль та залежність від фази можуть сприяти вчасному виявленню та виправленню помилок, а також підвищити якість та стабільність ПЗ.

Стандарт *ISO* / *IEC* 12207:1995 "*Information Technology – Software Life Cycle Processes*" є основним нормативним документом, який регламентує склад процесів життєвого циклу ПЗ. Він визначає структуру життєвого циклу, що містить процеси, дії і завдання, які повинні бути виконані під час створення ПЗ[1].

Процеси життєвого циклу ПЗ [10]:

1) основні:

– придбання (дії і завдання замовника, що здобуває ПЗ);

– поставка (дії і завдання постачальника, який постачає замовника програмним продуктом або послугою);

– розробка (дії і завдання, що виконуються розробником: створення ПЗ, оформлення проєктної та експлуатаційної документації, підготовка тестових та навчальних матеріалів і т. д.);

– експлуатація (дії і завдання оператора – організації, що експлуатує систему);

– супровід (дії і завдання, що виконуються супроводжує організацією, тобто службою супроводу). Супровід – внесень змін в ПЗ з метою виправлення помилок, підвищення продуктивності або адаптації до нових умов роботи або вимогам.

2) допоміжні:

– документування (формалізований опис інформації, створеної протягом ЖЦПЗ);

– управління конфігурацією (застосування адміністративних і технічних процедур на всьому протязі ЖЦПЗ для визначення стану компонентів ПЗ, управління його модифікаціями);

– Забезпечення якості (забезпечення гарантій того, що ІС і процеси її ЖЦ відповідають заданим вимогам та затвердженим планам);

– верифікація (визначення того, що програмні продукти, які є результатами певної дії, повністю задовольняють вимогам або умовам, обумовленим попередніми діями);

– атестація (визначення повноти відповідності заданих вимог і створеної системи їх конкретному функціональному призначенню);

– спільна оцінка (оцінка стану робіт по проєкту: контроль планування та управління ресурсами, персоналом, апаратурою, інструментальними засобами);

– аудит (визначення відповідності вимогам, планам і умов договору);

– дозвіл проблем (аналіз і рішення проблем, незалежно від їх походження чи джерела, які виявлені в ході розробки, експлуатації, супроводу або інших процесів).

3) Організаційні:

– управління (дії і завдання, які можуть виконуватися будь-якою стороною, що управляє своїми процесами);

– створення інфраструктури (вибір та супровід технології, стандартів та інструментальних засобів, вибір і установка апаратних і програмних засобів, що використовуються для розробки, експлуатації чи супроводу ПЗ);

– удосконалення (оцінка, вимір, контроль та вдосконалення процесів ЖЦ);

– навчання (початкове навчання і подальше постійне підвищення кваліфікації персоналу);

Кожен процес включає ряд дій. Наприклад, процес придбання охоплює наступні дії:

– ініціювання придбання;

– підготовка заявочних пропозицій;

– підготовка та коригування договору;

– нагляд за діяльністю постачальника;

– приймання та завершення робіт.

Кожна дія включає ряд завдань. Наприклад, підготовка заявочних пропозицій повинна передбачати:

– формування вимог до системи;

– формування списку програмних продуктів;

– встановлення умов і угод;

– опис технічних обмежень (середовище функціонування системи і т. д.).

Життєвий цикл програмного забезпечення є ключовою складовою процесу розробки програмного забезпечення. Він включає в себе фази від аналізу та планування до підтримки та обслуговування. Правильне управління ЖЦ ПЗ може суттєво покращити результативність проєктів та забезпечити високу якість програмного забезпечення.

## 1.3. Аналіз принципів керування проєктами

Керування проєктом є однією з найважливіших складових процесу реалізації проєктів програмного забезпечення (ПЗ). Відправною точкою будь-якого

успішного проєкту є ефективне управління його ресурсами, часом та завданнями. В даному підрозділі проведемо аналіз основних аспектів керування проєктом у контексті розробки ПЗ, розглянемо методи та інструменти, які використовуються для забезпечення успішного виконання проєктів.

Основні аспекти керування проєктом ПЗ

Керування проєктом у сфері розробки ПЗ передбачає ряд ключових аспектів:

1. Постановка мети та завдань: Визначення конкретної мети проєкту та завдань, які потрібно виконати для її досягнення. Це включає в себе аналіз вимог та очікувань стейкхолдерів.

2. Планування проєкту: Розроблення докладного плану проєкту, який включає в себе розклад завдань, розподіл ресурсів, бюджетування та інші параметри. Планування допомагає забезпечити ефективне виконання завдань та дотримання графіку.

3. Управління ресурсами: Ефективне використання ресурсів, таких як люди, обладнання та фінанси, є ключовим аспектом управління проєктом. Керування ресурсами включає в себе розподіл завдань між командами, контроль над бюджетом та взаємодію з постачальниками.

4. Ризиковий менеджмент: Ідентифікація та оцінка ризиків, які можуть вплинути на проєкт, та розроблення стратегій їх запобігання та/або управління. Ризиковий менеджмент допомагає зменшити негативні наслідки непередбачених обставин.

5. Звітність та комунікація: Встановлення системи звітності та ефективної комунікації зі стейкхолдерами проєкту. Це включає в себе регулярні звіти, засідання, статус-онлайн інформацію та інші комунікаційні інструменти.

6. Контроль та моніторинг: Постійний контроль за виконанням завдань, ресурсами та графіком проєкту. Моніторинг дозволяє вчасно виявляти відхилення від плану та приймати коректівні заходи.

Методи та інструменти керування проєктом ПЗ

Управління проєктом ПЗ включає в себе використання різноманітних методів та інструментів:

1. Методології розробки ПЗ: Agile, Waterfall, Scrum та інші методології надають рамки для організації та виконання проєктів.

2. Інструменти управління проєктом: Програми для створення графіків, планування завдань, звітності та моніторингу, такі як Microsoft Project, JIRA, Trello.

3. Системи версійного контролю: Допомагають в управлінні кодом та його версіями, такі як Git, Subversion.

4. Керування вимогами: Інструменти для збору, аналізу та відстеження вимог до ПЗ.

5. Інструменти для тестування: Забезпечують автоматизоване та ручне тестування ПЗ.

Вплив керування проєктом на успіх розробки ПЗ

Ефективне керування проєктом має безпосередній вплив на успішність розробки програмного забезпечення. Відсутність чіткого плану, поганий контроль та неефективне управління ресурсами можуть призвести до затримок, вибуху бюджету та низької якості ПЗ.

Керування проєктом у розробці програмного забезпечення є невід'ємною частиною успішної реалізації проєктів. Воно включає в себе планування, розподіл ресурсів, моніторинг та контроль за виконанням завдань. Використання правильних методологій та інструментів сприяє підвищенню ефективності та якості програмного забезпечення.

Керування проєктом у розробці програмного забезпечення вимагає інтегрованого підходу та уваги до деталей на всіх етапах життєвого циклу проєкту. Воно охоплює ряд критичних аспектів та процесів, які разом забезпечують успішну реалізацію проєктів. Далі наведено детальний опис основних компонентів керування проєктом у розробці програмного забезпечення:

1. Планування: Ефективне планування визначає мету, обсяги, ресурси, часові рамки та витрати проекту. Воно включає розробку плану проєкту, графіку виконання, бюджету, а також стратегії управління ризиками. Планування також

передбачає встановлення цілей та вимог до продукту, організацію робочих процесів та визначення критеріїв успішності.

2. Розподіл ресурсів: Керування ресурсами включає ідентифікацію, розподіл та оптимізацію всіх ресурсів, необхідних для виконання проекту. Це охоплює людські ресурси (команду проекту), технічні ресурси, матеріальні засоби та фінанси. Ефективне управління ресурсами забезпечує, що кожен елемент проекту має необхідні ресурси в потрібний час.

3. Моніторинг та контроль: Неперервний моніторинг ходу реалізації проекту та контроль за виконанням завдань є життєво важливими для забезпечення його успіху. Це включає відстеження прогресу щодо плану, ідентифікацію відхилень та негайне вживання заходів для коригування ситуації. Використання KPIs (ключових показників ефективності), регулярні зустрічі команди та оцінка якості роботи допомагають підтримувати контроль над проектом.

4. Комунікація: Ефективна комунікація всередині команди та зі стейкхолдерами є критичною для успішної реалізації проекту. Вона включає регулярний обмін інформацією, звітність, вирішення конфліктів та забезпечення, що всі учасники проекту мають актуальну інформацію та розуміють свої обов'язки.

5. Управління змінами: Проекти рідко проходять без змін. Управління змінами включає процеси ідентифікації, оцінки та апробації або відхилення змін у проекті. Ефективне управління змінами забезпечує, що всі модифікації добре документовані, оцінені та інтегровані з мінімальним розладом для загального плану проекту.

6. Контроль якості: Підтримка високого рівня якості - ключовий аспект управління проектами. Контроль якості охоплює розробку та виконання процедур перевірки та тестування продуктів та процесів на всіх етапах розробки для забезпечення відповідності вимогам та стандартам.

7. Ризик-менеджмент: Ідентифікація, оцінка та управління ризиками є невід'ємною частиною керування проєктом. Це включає розробку стратегій для

мінімізації або нейтралізації потенційних проблем, що можуть негативно вплинути на успіх проекту.

Використання правильних методологій, таких як Agile, Scrum, або Waterfall, та інструментів, залежно від специфіки проекту, може значно підвищити його ефективність та якість кінцевого продукту. Кожна методологія має свої сильні сторони та може бути адаптована до конкретних вимог та умов проекту. Правильний вибір та застосування цих методологій та інструментів є ключовим для успішної реалізації проєктів у сфері розробки програмного забезпечення.

## 1.4. Основні функції системи керування контентом на різних етапах розробки програмного забезпечення

Системи керування контентом (СКК) є незамінною складовою процесу реалізації проєктів програмного забезпечення (ПЗ). Вони дозволяють ефективно керувати всією інформацією та ресурсами, які використовуються на різних етапах розробки ПЗ. В даному підрозділі розглянемо основні функції СКК та їх роль на кожному етапі проєкту.

Функції СКК на початковому етапі проєкту

На початковому етапі проєкту ПЗ основною функцією СКК є організація та збереження початкової інформації про проєкт. Це включає в себе:

1. Збереження вимог та специфікацій: СКК дозволяють структуровано зберігати всі вимоги до ПЗ, які складаються на основі взаємодії з клієнтом та стейкхолдерами. Ця інформація стає основою для подальшого проектування.

2. Версійний контроль документів: СКК забезпечують можливість відстеження версій документів та їх змін, що робить процес роботи зі специфікаціями більш систематичним та структурованим.

3. Управління доступом: Системи керування контентом дозволяють обмежувати доступ до конфіденційної інформації та документів лише визначеним користувачам, забезпечуючи безпеку даних.

4. Комунікація та співпраця: За допомогою СКК можна організовувати комунікацію та співпрацю між членами команди, що сприяє ефективному обміну інформацією.

Функції СКК на етапі розробки ПЗ

Під час розробки ПЗ, СКК виконують наступні функції:

1. Управління вихідним кодом: СКК дозволяють зберігати та відстежувати версії вихідного коду програмного забезпечення. Це допомагає розробникам спільно працювати над кодом та контролювати його розвиток.

2. Автоматизація збірки та розгортання: СКК можуть бути інтегровані з системами автоматизації збірки та розгортання ПЗ, що полегшує процес розробки та тестування.

3. Спільна робота над задачами: СКК надають можливість створювати задачі, призначати їх виконавцям, відстежувати статус та прогрес вирішення завдань.

Функції СКК на етапі тестування та випуску

На завершальних етапах проєкту, СКК допомагають у:

1. Управлінні тестуванням: СКК дозволяють створювати тестові набори, відстежувати результати тестування та оформлювати звіти.

2. Документуванні та збереженні знань: СКК дозволяють створювати та зберігати документацію, яка необхідна для підтримки та обслуговування ПЗ після випуску.

3. Управлінні змінами та оновленнями: СКК допомагають відстежувати зміни та оновлення, які вносяться після випуску продукту, та забезпечують контроль за процесом.

Системи керування контентом відіграють ключову роль на різних етапах розробки програмного забезпечення. Вони забезпечують ефективне управління інформацією, ресурсами, комунікацією та процесами на проєкті. Використання СКК сприяє підвищенню продуктивності, зменшенню ризиків та підвищенню якості програмного забезпечення.

Весь набір прийомів групової динаміки можна розділити на дві групи: методи дослідження та прийоми на рівнях аналізу. У методах дослідження варіюються методи збору інформації та методи лікування. Методи даних часто не виділяються в спеціальній одиниці, оскільки більшість з них не є специфічними, але використовують загальнонаукові методи.

Методи збору даних:

1. Спостереження: Цей метод передбачає систематичний аналіз та фіксацію подій, які відбуваються в групі. Спостереження дозволяє отримати об'єктивну інформацію про взаємодію членів групи та їхню поведінку.

2. Експертиза документів (аналіз змісту): Цей метод використовується для аналізу текстової інформації, такої як документи, звіти, листування тощо. Аналіз змісту дозволяє виявити ключові теми, патерни та тенденції в комунікації.

3. Дослідження (анкетування, інтерв’ю): Дослідження засновані на опитуванні членів групи за допомогою анкет або інтерв'ю. Цей метод дозволяє зібрати детальну інформацію про думки, думки та почуття учасників групи.

4. Тест та експеримент: Тести та експерименти можуть бути використані для вимірювання певних характеристик або вивчення реакцій учасників групи на певні стимули або умови.

Рівні методів аналізу:

1. Індивідуальний аналіз: Цей рівень передбачає вивчення індивідуальних характеристик та особливостей членів групи, їхніх навичок, досвіду та мотивації. Індивідуальний аналіз може бути корисним для розуміння внеску кожного учасника у груповий процес.

2. Груповий аналіз: На цьому рівні вивчаються взаємодія та відносини між членами групи, групова динаміка, рольова структура та спільні цілі. Груповий аналіз спрямований на виявлення факторів, які впливають на ефективність групи.

"Людський капітал" – один із головних активів будь-якого виду бізнесу сьогодні. І цей капітал створити непросто (підготувати до необхідної кваліфікації), дуже важко знайти готовий (необхідний рівень), його важко підтримувати і, отже, грамотне управління ними – завдання з багатьма змінними.

Компетенції персоналу – це сукупність конкретних характеристик (знань, умінь та особистих якостей) кандидата, необхідних для успішного виконання своїх обов'язків. Компетенції визначають, чи відповідає кандидат вимогам посади та чи може він внести вклад у досягнення цілей компанії.

Центр оцінювання – це один із найбільш дієвих методів оцінки персоналу сьогодні. Він базується на моделюванні ключових моментів професійної діяльності та оцінці навичок, необхідних для роботи на певній посаді. Цей метод дозволяє отримати об'єктивну інформацію про кандидата і призначити його на посаду, на якій він буде найбільш ефективним.

Проте, центр оцінювання не завжди отримує широке застосування через свою вартість та необхідність висококваліфікованого технічного персоналу для проведення оцінки.

При вивченні проблем глобального розвитку було виявлено основні відмінності між розподіленими та реальними командами.

1. Для розподілених команд характерні особливості інструменту та послаблення міжособистісної взаємодії. Розподілені команди часто використовують онлайн інструменти та платформи для комунікації та співпраці, що може впливати на спосіб взаємодії між їхніми членами.

2. Етапи розвитку розподілених команд можуть бути скороченими в часі та стиснутими соціальним та психологічним змістом. У розподілених командах може бути обмежений розвиток міжособистісних відносин через фізичну віддаленість та обмежену особисту комунікацію.

3. Групові норми, що виникають у розподілених командах, спрямовані на включення всіх членів команди для обміну інформацією та оперування завданнями. У розподілених командах може бути складніше встановлювати загальні правила та норми взаємодії через віддалену природу комунікації.

4. Обмін інформацією в розподілених командах базується на принципі "кожен до кожного". Онлайн інструменти та платформи дозволяють членам команди обмінюватися інформацією безпосередньо, без посередника.

Тому методи групової динаміки до традиційних команд не застосовуються до команд у контексті глобального розвитку. У розподілених командах важливо враховувати особливості їхнього функціонування та знаходити ефективні методи співпраці та комунікації для досягнення успіху в проектах програмного забезпечення.

## 1.5. Висновки до розділу

У ході виконання першого розділу було проведено глибокий аналіз процесів, що лежать в основі створення програмного забезпечення. Огляд показав, що ефективність розробки значною мірою залежить від інструментарію та методів управління проектами, що використовуються. В умовах глобального розширення ринку програмних продуктів та зростаючої конкуренції, вибір адекватної моделі управління проектами є пріоритетним завданням для розробників, яке має відповідати їх амбіціям та унікальним вимогам.

Використання передових засобів розробки відіграє вирішальну роль у формуванні можливостей розробників, безпосередньо впливаючи на якість та швидкість створення програмних продуктів. Оптимальний вибір технологічного стеку може стати вагомим фактором у поліпшенні ефективності розробки.

Ефективне управління проектами є ключовим для успішної реалізації програмного забезпечення, починаючи з точного планування, ретельного аналізу ризиків, раціонального розподілу завдань, і закінчуючи суворим контролем за їх виконанням. Недоліки в управлінні можуть призвести до затягування термінів виконання, збільшення бюджету і, як наслідок, незадоволення замовника.

Тому, адекватний вибір моделей управління розробкою та ефективних інструментів є критичним для досягнення успіху в індустрії програмного забезпечення. У наступних розділах цієї роботи буде здійснено детальний огляд різноманітних методик управління проектами та підходів до групової роботи, що можуть внести значний вклад у розв'язання цього комплексного завдання.

# РОЗДІЛ 2
# ВИКОРИСТАННЯ МЕТОДІВ ГРУПОВОЇ ДИНАМІКИ ПРИ ПРОГНОЗУВАННІ ЧАСУ ВИКОНАННЯ ПРОЄКТІВ З РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## 2.1. Класифікація методів групової динаміки

В сучасному світі, де глобальна розробка програмного забезпечення стала стандартом, групова динаміка в командах розробників грає важливу роль. Зрозуміння і ефективне використання методів групової динаміки стає ключовим для досягнення успіху в глобальному програмному розробництві. У даному розділі ми розглянемо класифікацію методів групової динаміки, яка допоможе краще розуміти і використовувати їх у глобальному контексті.

Формалізовані методи групової динаміки базуються на точних наукових принципах та структурах. Вони часто використовуються в процесі прийняття рішень та аналізу великих обсягів даних у глобальних командах розробників. Основні підходи включають:

1. Математичне моделювання: Математичні моделі використовуються для аналізу та прогнозування різних аспектів групової динаміки, таких як розподіл завдань, часовий графік та ресурси. Вони дозволяють розробникам приймати обгрунтовані рішення на основі обчислень.

2. Теорія гри: Теорія гри використовується для моделювання взаємодії учасників команди та прийняття стратегічних рішень. Вона допомагає аналізувати ситуації конфлікту та співпраці та визначати оптимальні рішення.

3. Оптимізація та лінійне програмування: Ці методи використовуються для пошуку оптимальних рішень при обмежених ресурсах. Вони можуть бути застосовані для планування та розподілу завдань у глобальних командах розробників.

4. Системи підтримки прийняття рішень: Ці системи надають інструменти для збору, аналізу та обробки даних у реальному часі. Вони

допомагають розробникам приймати обгрунтовані рішення на основі актуальної інформації.

Неформалізовані методи групової динаміки базуються на соціальних та психологічних аспектах взаємодії учасників команди. Вони акцентують увагу на відносинах, спілкуванні та співпраці. Основні підходи включають:

1. Комунікація та співпраця: Важливим аспектом групової динаміки є здатність команди спілкуватися та співпрацювати. Неформальні методи спрямовані на покращення комунікації та розвиток ефективних співпрацюючих відносин.

2. Лідерство та мотивація: Лідерство грає важливу роль у формуванні динаміки команди. Методи лідерства та мотивації спрямовані на підтримку та мобілізацію команди для досягнення спільних цілей.

3. Конфлікти та їх управління: Конфлікти можуть виникати в груповій динаміці, і важливо вміти їх вирішувати та управляти. Методи управління конфліктами спрямовані на зменшення напруги та підтримку конструктивного вирішення проблем.

4. Розвиток особистості та навичок команди: Навички розвитку особистості та навички команди сприяють вдосконаленню учасників команди та підвищенню ефективності групової роботи.

Комбіновані методи групової динаміки поєднують у собі як формалізовані, так і неформалізовані підходи. Вони враховують як наукові принципи, так і психологічні аспекти взаємодії команди. Комбіновані методи надають можливість гнучко реагувати на різноманітні ситуації та досягати балансу між різними аспектами групової динаміки.

У глобальному розробництві програмного забезпечення важливо мати розуміння різних методів групової динаміки та вміти їх використовувати в залежності від конкретних завдань та контексту. Користування правильними методами може значно покращити ефективність та результативність глобальних розробочих команд, що впливає на якість та успіх програмних проектів.

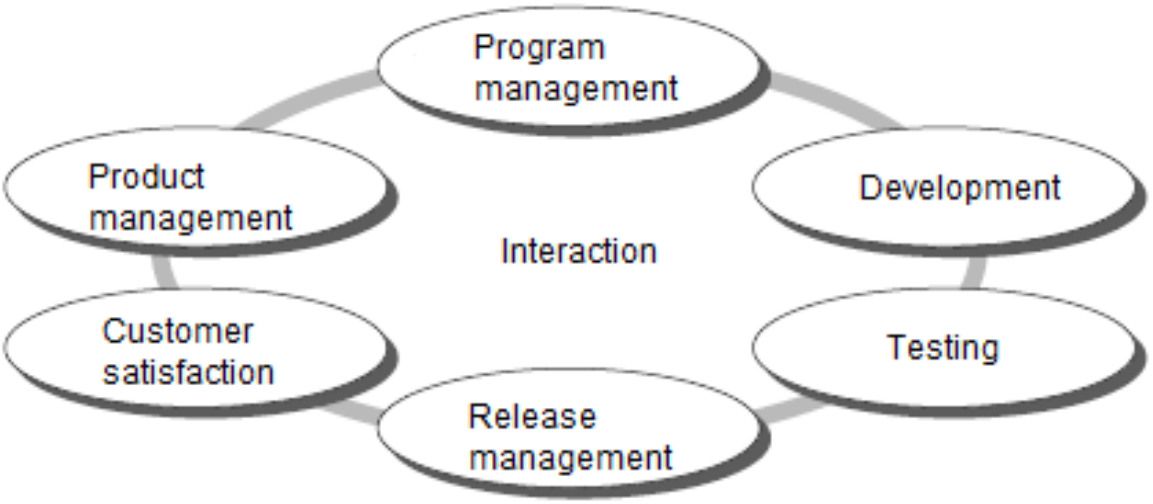


Рис. 2.1. Рольові кластери Модель команди *MSF*

Класифікація методів групової динаміки охоплює широкий спектр підходів і технік, кожен з яких спрямований на покращення взаємодії та продуктивності групи. Вони можуть бути розділені на кілька категорій залежно від цілей та контексту їх використання:

1. Комунікативні методи:

• Обмін інформацією: Вправи та техніки, які сприяють відкритому обміну інформацією та думками між учасниками.

• Активне слухання: Техніки, що покращують здатність учасників групи зосередитися на словах співрозмовника та розуміти його позицію.

• Ненасильницьке спілкування: Метод, що наголошує на важливості емпатії та вираження власних потреб без ворожості чи образ.

2. Рішення конфліктів:

• Медіація: Підхід, де третя сторона допомагає учасникам досягти взаємоприйнятного рішення.

• Конфліктологія: Вивчення конфліктів та методів їх вирішення для запобігання ескалації та знаходження ефективних рішень.

3. Прийняття рішень:

• Мозковий штурм: Генерація ідей у вільній формі, що дозволяє кожному учаснику внести свої пропозиції.

• Метод Дельфі: Систематичне опитування експертів із зворотнім зв'язком для досягнення консенсусу.

• Голосування: Демократичний метод прийняття рішень, де кожен учасник має один голос.

4. Лідерство та управління:

• Ситуативне лідерство: Адаптація стилю керівництва до конкретної ситуації та потреб команди.

• Емпауермент: Надання учасникам команди більше влади та відповідальності для підвищення мотивації та власної ефективності.

5. Психологічні методи:

• Динаміка ролей: Вивчення та оптимізація розподілу ролей в групі для ефективної взаємодії.

• Групова когезія: Стратегії зміцнення зв'язків між членами групи для підвищення згуртованості.

6. Організаційні методи:

• Тімбілдінг: Заходи, спрямовані на покращення взаєморозуміння та співпраці в команді.

• Скрам, Канбан: Адаптивні методи управління проектами, що сприяють швидкій реакції на зміни.

7. Тренінги та розвиток:

• Ролеві ігри: Симуляції реальних ситуацій для розвитку навичок і відпрацювання стратегій поведінки.

• Коучинг: Індивідуальна підтримка для розвитку особистих та професійних якостей членів команди.

8. Культурні та соціальні методи:

• Крос-культурне спілкування: Техніки та стратегії для ефективного спілкування в міжкультурному середовищі.

• Соціальні норми: Розробка та імплементація норм, що визначають прийнятну поведінку в команді.

Ці методи можна застосовувати індивідуально або в комбінації для створення ефективної робочої атмосфери та забезпечення успішного виконання проектів в рамках запланованих термінів.

Методи групової динаміки в контексті прогнозування часу виконання проєктів відіграють важливу роль, оскільки вони дозволяють врахувати людський фактор, що часто є непередбачуваним і складним для кількісного аналізу. Ці методи допомагають оцінити, як взаємодія між членами команди може впливати на продуктивність і терміни проектів. Нижче представлено кілька ключових аспектів, що відносяться до групової динаміки у цьому контексті.

Комунікація в команді: Ефективна комунікація є критичною для успішного прогнозування та виконання проєктів. Вона включає ясність в постановці задач, обмін знаннями та досвідом між членами команди, а також вчасне реагування на питання та проблеми.

Рішення конфліктів: Конфлікти можуть виникати з різних причин, включаючи несумісність цілей, методів роботи або особистісних характеристик. Методи групової динаміки допомагають ідентифікувати і розв'язувати конфлікти на ранніх стадіях, перш ніж вони вплинуть на терміни проекту.

Прийняття рішень: У груповій динаміці важливо визначити, як команда приймає рішення, особливо у відповідь на зміни в проекті. Методи, такі як консенсус або демократичне голосування, можуть використовуватися для досягнення єдності команди.

Лідерство та управління: Лідери проектів відіграють важливу роль у груповій динаміці, оскільки вони встановлюють напрямок, координують завдання та мотивують команду. Вони також відповідають за розподіл ресурсів та вирішення проблем, що можуть виникнути під час виконання проєкту.

Розвиток команди: Стадії розвитку команди, як-от формування, штормінг, нормування та виконання, мають істотний вплив на терміни проектів. Розуміння цих стадій може допомогти управлінню проектом краще спланувати і прогнозувати виконання робіт.

Мотивація та залученість: Мотивація команди і її залученість у проект є ключовими для досягнення цілей у встановлені терміни. Методи групової динаміки дозволяють аналізувати й оптимізувати ці аспекти, впроваджуючи такі засоби, як внутрішні стимули, визнання досягнень та кар'єрне зростання.

Культурні та соціальні фактори: Культурні різниці та соціальні аспекти в міжнародних командах можуть впливати на сприйняття роботи, ставлення до термінів та взаємодії в команді. Знання та повага до цих різниць можуть поліпшити групову взаємодію та співпрацю.

Гнучкість та адаптація: Спроможність команди адаптуватися до змін є важливою для забезпечення своєчасного завершення проектів. Методи групової динаміки дозволяють визначити, наскільки швидко та ефективно команда може адаптуватися до нових обставин.

Розуміння та застосування методів групової динаміки в контексті прогнозування часу виконання проєктів може значно підвищити точність планування, знизити ризики і підвищити якість роботи. Це вимагає не лише технічного розуміння процесів розробки, але й глибоких знань про людський фактор та взаємодію в команді.

## 2.2. Метод компетентності для підбору персоналу до команди

У глобальному розробництві програмного забезпечення ключовим фактором успіху є якісний та досвідчений персонал. Вибір правильних фахівців для глобальної розробочої команди має вирішальне значення, оскільки від цього залежить якість та швидкість розробки програмного продукту. У цьому підрозділі розглянемо метод компетентності, який використовується для підбору персоналу та формування ефективних розробочих команд у глобальному контексті.

Компетентність – це комплекс характеристик, які включають знання, вміння, навички, досвід та особистісні якості, що роблять можливим успішне виконання роботи або завдання. У контексті глобального розробництва програмного

забезпечення компетентність означає здатність розробника або команди до виконання завдань та досягнення поставлених цілей.

Метод компетентності використовується для визначення, які саме компетенції необхідні для успішної роботи на певній посаді або участі у глобальній розробочій команді. Цей метод передбачає такі основні кроки:

Спочатку важливо зрозуміти, які саме завдання або функції пов'язані з певною посадою чи роллю в команді. Цей аналіз допомагає визначити ключові аспекти роботи та очікувані результати.

На основі аналізу посади чи завдання визначаються необхідні компетентності. Це можуть бути технічні навички, мови програмування, знання спеціалізованих інструментів, але також і міжособистісні якості, такі як комунікабельність, лідерство, співпраця та інші.

Для визначення компетентностей кандидатів або членів команди можуть застосовуватися різні методи, включаючи спеціальні тести, співбесіди, портфоліо робіт, а також оцінку на практиці. Оцінка допомагає виявити, наскільки кандидати відповідають вимогам щодо компетентностей.

На основі результатів оцінки компетентностей відбираються кандидати, які найкраще відповідають вимогам. При формуванні глобальної розробочої команди важливо також враховувати комбінацію різних компетентностей для досягнення балансу та ефективності.

Метод компетентності має свої переваги та обмеження. До основних переваг відносяться:

- Об'єктивність: Визначення компетентностей базується на конкретних критеріях, що робить оцінку більш об'єктивною та спрямованою на результат.
- Вибір найкращих: Метод дозволяє відбирати кандидатів з наибільш відповідними навичками та якостями для конкретної ролі.
- Адаптивність: Метод можна адаптувати під конкретні вимоги та специфіку проекту.

Проте, метод компетентності також має обмеження:

- Відсутність комплексного підходу: Метод фокусується на конкретних компетентностях, але не завжди враховує загальну культуру та взаємодію в команді.
- Суб'єктивність оцінки: В процесі оцінки можуть виникати суб'єктивні судження, що може призвести до неточностей.
- Динамічність вимог: Компетентності можуть змінюватися з часом, і метод вимагає постійного оновлення та адаптації.

У глобальному розробництві програмного забезпечення метод компетентності має велике значення. Він дозволяє не тільки відібрати кандидатів з необхідними навичками для роботи у віддалених командах, але і формувати збалансовані та ефективні розробочі групи. Кожен член команди має внести свій унікальний вклад, що сприяє більш швидкому та якісному розвитку програмного продукту.

Загалом, метод компетентності є важливим інструментом для досягнення успіху у глобальному розробництві програмного забезпечення. Він дозволяє забезпечити належний рівень якості та ефективності проектів завдяки правильному вибору та формуванню персоналу у глобальних розробочих командах.

Метод компетентності для підбору персоналу до команди базується на ідеї, що для ефективної роботи команди кожен її член повинен мати відповідний набір знань, навичок і поведінкових характеристик, які відповідають вимогам ролі та цілям команди. Цей метод використовується для оцінки та відбору кандидатів на основі визначених компетентностей, які необхідні для досягнення конкретних бізнес-цілей.

Теоретичні основи методу компетентності:

1. Визначення компетентності: Компетентність — це комбінація знань, вмінь, досвіду та поведінки, яка дозволяє особі ефективно виконувати завдання. Вона може включати такі аспекти, як технічна експертиза, управлінські навички, емоційний інтелект та інші особистісні якості.

2. Модель компетентності: Модель компетентності — це структурований набір компетентностей, який необхідний для виконання конкретних видів робіт або ролей у проекті. Компетентності поділяються на кілька категорій, включно з базовими, професійними, менеджерськими та лідерськими.

3. Оцінка та відбір персоналу: Процес оцінки передбачає зіставлення компетентностей кандидатів з вимогами моделі. Це включає аналіз резюме, проведення співбесід, тестування та використання оціночних центрів.

4. Розвиток та навчання: Однією з ключових переваг методу компетентностей є зосередження на розвитку персоналу. Визначаються прогалини в компетентностях і розробляються індивідуальні плани розвитку, які можуть включати навчання, менторство та кар'єрне планування.

5. Робота з талантами: Метод компетентності також використовується для управління талантами в організації. Це допомагає ідентифікувати ключових працівників з необхідними навичками та якостями для керівництва та розвитку.

Практичне застосування методу компетентності:

1. Підбір команди проекту: При формуванні команди проекту важливо, щоб члени команди мали компетентності, які доповнюють одна одну. Це створює збалансовану команду, де кожен може внести свій вклад у вирішення різноманітних завдань.

2. Кар'єрне планування: Метод компетентності може застосовуватися для планування кар'єри співробітників, допомагаючи їм ідентифікувати області для розвитку та шляхи професійного зростання в компанії.

3. Оцінка продуктивності: Регулярна оцінка компетентностей співробітників дозволяє визначити ефективність їхньої роботи та встановити цілі для подальшого розвитку.

Виклики при застосуванні методу компетентності:

• Суб'єктивізм у оцінці: Оцінка компетентностей може бути суб'єктивною, і різні менеджери можуть інтерпретувати одну й ту ж компетентність по-різному.

- Зміна компетентностей: Компетентності можуть змінюватися з часом у відповідь на нові технології та бізнес-процеси, тому модель компетентностей потребує регулярного оновлення.
- Інтеграція з корпоративною культурою: Метод компетентності повинен відображати і підтримувати корпоративну культуру та цінності організації.

Метод компетентності є важливим інструментом для забезпечення того, щоб правильні люди працювали на правильних позиціях, що є ключовим для успішного виконання проектів та досягнення бізнес-цілей.

## 2.3. Методи групової динаміки при прогнозуванні часу виконання проєктів

Глобальне розробництво програмного забезпечення (ПЗ) є складним завданням, яке вимагає специфічного підходу до управління робочими групами та взаємодії між їх членами. У цьому підрозділі ми розглянемо методи групової динаміки, їхнє застосування та важливість у глобальній розробці ПЗ.

Групова динаміка - це область психології, що вивчає взаємодію та внутрішні відносини між членами групи та вплив цих відносин на колективну роботу та результати. У глобальній розробці ПЗ, де робочі групи часто розташовані на великих відстанях, групова динаміка відіграє ключову роль у досягненні успіху проекту.

У глобальній розробці ПЗ методи групової динаміки можна розділити на кілька основних категорій:

Комунікація є ключовим аспектом групової динаміки в глобальному розробництві ПЗ. Забезпечення ефективної комунікації між членами розробочої команди, які можуть бути розташовані у різних країнах та часових поясах, допомагає уникнути недорозумінь, зберегти єдність команди та спрямовує всі зусилля на досягнення загальних цілей. До методів комунікації в глобальній

розробці ПЗ включають віртуальні наради, спільні чати, використання спеціалізованих інструментів для спільної роботи тощо.

Формування ефективної розробочої команди важливо для досягнення успішних результатів у глобальному розробництві ПЗ. Методи групової динаміки допомагають збирати та вдосконалювати команду. Це включає в себе процес підбору спеціалістів, розподіл завдань, розвиток спільних цілей та створення позитивної корпоративної культури.

Конфлікти можуть виникнути в будь-якій команді, але вони особливо важливі у глобальному розробництві ПЗ, де команди можуть бути розташовані у різних культурних та часових контекстах. Методи групової динаміки допомагають вирішувати конфлікти та забезпечують сприятливий клімат для спільної роботи.

Методи групової динаміки також спрямовані на підвищення продуктивності команди. Це може включати в себе розробку спільних методів роботи, визначення ефективних процесів та створення мотиваційних систем.

Методи групової динаміки мають велике значення в глобальному розробництві ПЗ з наступних причин:

- Забезпечення ефективної комунікації та співпраці між членами команди.
- Формування та розвиток ефективних розробочих команд.
- Вирішення конфліктів та підтримка позитивного робочого середовища.
- Збільшення продуктивності та якості роботи.

У глобальному розробництві ПЗ, де команди можуть бути розпорошені по всьому світу, важливо використовувати методи групової динаміки для забезпечення успіху проекту та досягнення високих результатів.

Розглянемо конкретні приклади застосування методів групової динаміки в глобальному розробництві ПЗ:

Комунікація через спеціалізовані інструменти. У глобальних командах розвиток спеціалізованих інструментів для спільної роботи та комунікації є ключовим елементом успіху. Наприклад, використання інтернет-платформ для

спільної роботи, де можливо обмінюватися інформацією, завданнями, документами та коментарями, дозволяє команді бути завжди на зв'язку та зручно обговорювати питання.

Між цими двома варіантами структур існують, як правило, змішані форми (рис. 2.2).

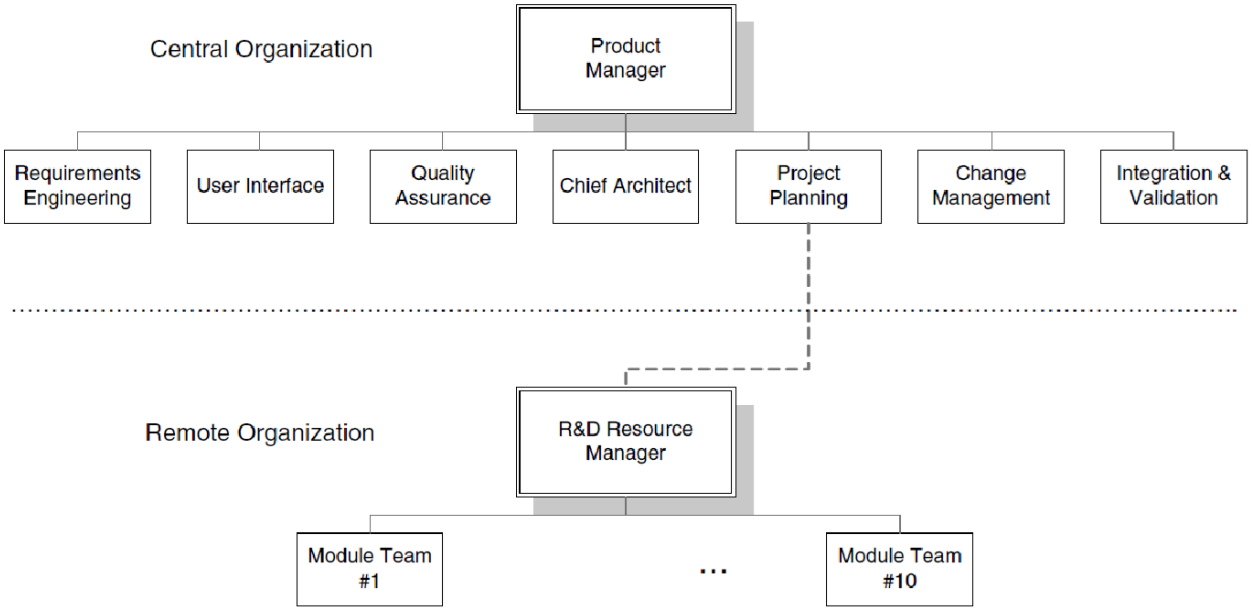


Рис. 2.2. Організація розподіленої діяльності [1]

## 2.4. Обгрунтування вибору клієнт-серверної архітектури

Обґрунтування вибору клієнт-серверної архітектури для будь-якого програмного проекту залежить від ряду технічних, економічних та стратегічних чинників. Цей вибір впливає на ефективність, масштабованість, надійність, доступність та безпеку системи. Далі наведено докладний аналіз цих чинників.

Технічні чинники:

1. Розділення обов'язків: Клієнт-серверна архітектура дозволяє чітко розділити обов'язки між клієнтом (кінцевим користувачем) та сервером (центром обробки даних). Це розділення сприяє спеціалізації та оптимізації виконання функцій, що покращує продуктивність системи.

2. Масштабованість: Системи на базі клієнт-серверної архітектури легко масштабуються за рахунок додавання або оновлення серверних ресурсів. Це важливо для зростаючих організацій та систем, що вимагають гнучкості у розширенні своїх можливостей.

3. Управління ресурсами: Сервери можуть ефективно управляти ресурсами та розподіляти їх між багатьма клієнтами, оптимізуючи загальну продуктивність системи та забезпечуючи краще використання апаратних ресурсів.

4. Централізоване оновлення та управління: Оновлення програмного забезпечення, застосування патчів безпеки та зміни в конфігурації можуть виконуватися централізовано на сервері, зменшуючи час та витрати на технічне обслуговування.

5. Безпека: Клієнт-серверні архітектури дозволяють краще контролювати доступ до даних та ресурсів. Централізоване управління безпекою на сервері полегшує реалізацію єдиної політики безпеки та захисту від несанкціонованого доступу.

Економічні чинники:

1. Зниження загальних витрат: Використання клієнт-серверної архітектури може знизити витрати на IT-інфраструктуру за рахунок централізації ресурсів та управління. Централізація допомагає оптимізувати використання апаратного забезпечення та програмного ліцензування.

2. Економія на технічному обслуговуванні: Централізоване обслуговування серверів знижує витрати на технічну підтримку, оскільки оновлення та ремонт можуть здійснюватися в одному місці, без необхідності фізичної присутності у всіх точках використання.

3. Оптимізація капіталовкладень: Організації можуть інвестувати в потужні сервери та високоякісне мережеве обладнання, оптимізуючи тим самим загальні капіталовкладення в IT-інфраструктуру.

Стратегічні чинники:

1. Гнучкість та адаптація до бізнес-змін: Клієнт-серверні системи можуть швидко адаптуватися до змін у бізнес-процесах та ринкових умовах, надаючи можливість швидкого реагування на нові вимоги та можливості.

2. Підтримка стратегічного планування: Централізоване управління даними та ресурсами дозволяє краще планувати і прогнозувати потреби в IT-ресурсах, сприяючи ефективному стратегічному плануванню.

3. Можливість інтеграції та розширення: Клієнт-серверна архітектура дозволяє легко інтегрувати нові програмні рішення та розширювати існуючі системи за рахунок додавання нових серверів або сервісів.

Висновок: Вибір клієнт-серверної архітектури має бути обґрунтованим на основі глибокого аналізу потреб бізнесу, технічних можливостей організації, економічних розрахунків та стратегічного бачення розвитку. Це забезпечить не тільки ефективність та надійність роботи системи, а й її масштабованість, гнучкість та здатність адаптуватися до змін у бізнесі та технологіях.

Вибір клієнт-серверної архітектури для системи прогнозування часу виконання проєктів обґрунтований низкою чинників, що відображають вимоги таких систем до обробки даних, доступності, масштабованості та взаємодії. Ось основні причини, чому клієнт-серверна архітектура є вибором для системи прогнозування часу виконання проєктів:

1. Централізоване управління даними: Системи прогнозування потребують доступу до великих обсягів історичних даних про проекти, включаючи терміни виконання, ресурси, ефективність команди та інші параметри. Клієнт-серверна архітектура дозволяє централізовано зберігати та управляти цими даними, забезпечуючи їх консистенцію, безпеку та легкий доступ для аналітики.

2. Масштабованість та розподіл навантаження: Прогнозування часу виконання проєктів може бути ресурсноємним, особливо для великих організацій з багатьма одночасними проектами. Клієнт-серверна архітектура дозволяє легко масштабувати систему, додаючи серверні ресурси, а також розподіляти навантаження між серверами для оптимізації продуктивності.

3. Висока доступність та надійність: Для критичних бізнес-систем, як-от системи прогнозування, важлива висока доступність та надійність. Клієнт-серверні архітектури можуть включати засоби для відновлення після збоїв, реплікації даних та балансування навантаження, що гарантує безперебійну роботу системи.

4. Централізоване оновлення та обслуговування: Системи прогнозування регулярно оновлюються з метою вдосконалення алгоритмів та врахування нових даних. Клієнт-серверна архітектура дозволяє централізовано оновлювати та обслуговувати систему, зменшуючи простої та гарантуючи, що всі користувачі працюють з останньою версією програмного забезпечення.

5. Інтеграція з іншими системами: Системи прогнозування часто потрібно інтегрувати з іншими корпоративними системами, такими як системи управління проектами, ERP та HRM. Клієнт-серверна архітектура спрощує цю інтеграцію, надаючи єдині точки доступу (APIs) для обміну даними між системами.

6. Гнучкість вибору клієнтських пристроїв: В організаціях, де команди мають різноманітні потреби та працюють на різних пристроях, клієнт-серверна архітектура дозволяє використовувати різноманітні клієнтські пристрої (ПК, мобільні телефони, планшети) для доступу до системи, забезпечуючи гнучкість і зручність для користувачів.

7. Безпека даних: Важливість захисту конфіденційних даних не можна недооцінювати. Клієнт-серверна архітектура забезпечує можливість впровадження цілісних стратегій безпеки, включаючи шифрування, контроль доступу та аудит, що гарантує безпеку чутливих даних та забезпечує відповідність нормативним вимогам.

Обґрунтування вибору гнучкої методології Agile для систем прогнозування часу виконання проєктів засноване на її принципах та перевагах, які відповідають динаміці та складності сучасних проектів. Ось ключові аспекти, які виправдовують вибір Agile:

1. Адаптивність до змін: Проекти часто зазнають змін у вимогах, обсягах та цілях. Agile методологія, з її короткими ітераціями та регулярним переглядом пріоритетів, дозволяє швидко адаптуватися до змін та внести необхідні корективи в плани та прогнози, забезпечуючи актуальність та відповідність реальним потребам.

2. Фокус на користувацьку цінність: Agile підкреслює важливість доставки користувацької цінності на ранніх етапах та продовженні її забезпечення протягом усього проекту. Це означає, що прогнозування часу виконання орієнтоване на досягнення корисних результатів, а не лише на слідування жорстким термінам.

3. Покращена комунікація та співпраця: Agile вимагає тісної співпраці між усіма членами команди та зацікавленими сторонами. Це сприяє кращому розумінню вимог, пріоритетів та потенційних ризиків, що підвищує точність прогнозів часу виконання та зменшує ймовірність непередбачених затримок.

4. Прозорість та зворотний зв'язок: Регулярні зустрічі (наприклад, щоденні стендапи, ретроспективи, демонстрації) і відкритий обмін інформацією забезпечують прозорість ходу проекту та дозволяють швидко виявляти та вирішувати проблеми, що також сприяє більш точному прогнозуванню часу.

5. Ітеративний підхід та раннє тестування: Agile дозволяє рано та часто тестувати функціональність, що може виявити потенційні проблеми та неефективності на ранніх стадіях, забезпечуючи вищу якість кінцевого продукту і більш точне прогнозування.

6. Гнучке планування та прийняття рішень: В Agile планування виконується постійно, а рішення приймаються на основі актуальної інформації. Це забезпечує можливість оперативно коригувати плани та прогнози, засновуючись на реальних даних та результатів команди.

7. Неперервне вдосконалення: Агільні методології підтримують культуру неперервного вдосконалення, де кожна ітерація та проект стає можливістю для навчання та покращення процесів, включно з прогнозуванням часу виконання.

Вибір Agile для систем прогнозування часу виконання проєктів надає командам гнучкість, необхідну для адаптації до змін, підтримки високої продуктивності та забезпечення якості та відповідності результатів очікуванням зацікавлених сторін. Завдяки цим перевагам Agile стає оптимальним вибором для динамічного та непередбачуваного світу сучасної розробки програмного забезпечення.

Вибір архітектури для системи прогнозування часу виконання проєктів може бути ефективно здійснений з використанням платформи .NET від Microsoft, що інтегрує легкість у розробці з високою продуктивністю виконання коду. Така платформа пропонує багатий набір інструментів і можливостей для розробки, включаючи технологію .NET Remoting, яка дозволяє створювати розподілені додатки з можливістю взаємодії між різними компонентами системи через мережу.

Переваги платформи .NET для прогнозування часу виконання проєктів:

1. Єдина розробницька платформа: .NET надає єдине середовище для розробки клієнтських та серверних додатків, що забезпечує консистентність та зменшує криву навчання для розробників.

2. Безпечна взаємодія компонентів: .NET Remoting дозволяє безпечно і ефективно взаємодіяти компонентам системи, незалежно від їх розташування, що є ключовим для розподілених систем прогнозування.

3. Висока продуктивність та оптимізація коду: Завдяки оптимізованому виконанню коду та ефективному управлінню ресурсами, .NET забезпечує високу продуктивність необхідну для обробки великих обсягів даних і складних алгоритмів прогнозування.

4. API для розширення можливостей: Інтегровані API в Adapton Server та інші компоненти .NET дозволяють розробникам легко розширювати функціональність і інтегрувати систему прогнозування з іншими додатками та сервісами.

5. Підтримка мов програмування: .NET підтримує багато мов програмування, дозволяючи розробникам вибрати найбільш підходящу мову для конкретного компонента системи або типу завдання.

6. Легкість розгортання та обслуговування: Системи, розроблені на платформі .NET, легко розгортати та підтримувати, що знижує загальні витрати на власність та спрощує управління версіями та оновленнями.

Узагальнюючи, використання платформи .NET і технології .NET Remoting для розробки системи прогнозування часу виконання проєктів надає високий рівень гнучкості, продуктивності та безпеки. Це забезпечує відмінну базу для створення надійної, масштабованої та ефективної системи, яка може адаптуватися до різних потреб та вимог проектів.

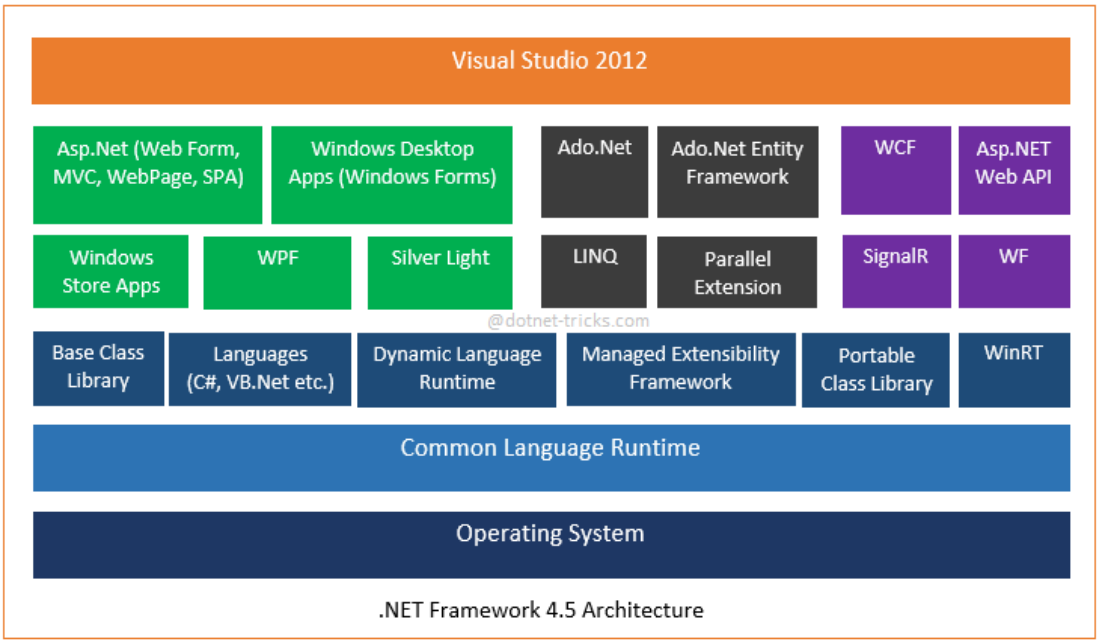


Рис. 2.1. Архітектура.*NET Framework*

## 2.5. Висновки до розділу

У другому розділі кваліфікаційної роботи проведено всебічний розгляд та систематизацію методів групової динаміки, які істотно впливають на процес відбору персоналу та прогнозування часу виконання проектів. Аналітична робота була зосереджена на двох основних напрямках: методології збору даних та стратегіях глибинного аналізу. Виявлено, що загальноприйняті техніки збору інформації, такі як анкетування та інтерв'ювання, спостереження та аналіз документації, можуть бути адаптовані для використання у специфіці ІТ-галузі.

Встановлено сім основних факторів, що впливають на динаміку роботи глобальних команд. Це дозволило розробити комплексну модель, яка ілюструє зв'язки між цими факторами та їх вплив на продуктивність команди, використовуючи математичний апарат методу найменших квадратів для кількісного оцінювання цих взаємозв'язків.

Важливо підкреслити, що "людський капітал" залишається основним активом в будь-якій організації, і його розвиток та залучення в проектну діяльність є визначальними для успішності виконання проектів. Процес підбору персоналу та управління компетенціями вимагає глибокого аналізу та використання методично обґрунтованих компетенцій, які включають не тільки професійні знання та навички, а й особистісні якості та мотивацію кандидатів.

Центри оцінки персоналу виявилися ефективним інструментом, що дозволяє всебічно оцінити потенціал працівників, хоча їх використання обмежене через високі витрати та необхідність залучення висококваліфікованих фахівців для проведення оцінювання.

Дослідження також звернуло увагу на унікальні виклики, з якими стикаються розподілені команди, такі як зменшення особистісної взаємодії та зміни у стандартних етапах розвитку команди. Відмінності між розподіленою та традиційною командою вимагають адаптованих підходів до групової динаміки, що включають спеціально розроблені інструменти для ефективної взаємодії та управління проектами в умовах глобальної розробки ПЗ.

# РОЗДІЛ 3
# РЕАЛІЗАЦІЯ ПРОГРАМНОГО МОДУЛЯ ПРОГНОЗУВАННЯ ЧАСУ ВИКОНАННЯ ПРОЄКТІВ

## 3.1. Дослідження методу соціально-технічної послідовності в прогнозуванні часу виконання проєктів

Метод соціально-технічної послідовності є одним із сучасних підходів до управління проектами та прогнозування часу їх виконання. Цей метод поєднує в собі соціальні та технічні аспекти проекту для покращення його результативності. Основні принципи цього методу включають:

- Інтеграція соціальних та технічних факторів: Однією з ключових ідей методу є зіставлення соціальних аспектів, таких як комунікація між членами команди, з технічними параметрами проекту. Це дозволяє краще розуміти, як взаємодія між людьми впливає на технічну частину проекту і навпаки.
- Аналіз послідовності подій: Метод передбачає аналіз послідовності подій у проекті, включаючи робочі процеси, завдання та взаємозв'язки між ними. Цей аналіз допомагає визначити, які етапи проекту вимагають більшої уваги та оптимізації.
- Управління ризиками: Метод соціально-технічної послідовності враховує ризики, пов'язані з якістю комунікації в команді та технічними ризиками проекту. Це дозволяє вчасно виявляти та усувати проблеми, що можуть впливати на результати проекту.

### 3.1.2. Застосування методу соціально-технічної послідовності в програмному модулі

У реалізації програмного модуля для прогнозування часу виконання проєктів в ІТ-компанії метод соціально-технічної послідовності використовується для покращення управління проектами та точнішого прогнозування часових рамок. Це досягається за допомогою наступних кроків:

- Визначення соціальних факторів: Перший крок полягає в аналізі соціальних аспектів команди, таких як рівень комунікації, співпраця та роль кожного члена. Для цього можуть використовуватися методи соціометрії та аналізу ролей.
- Аналіз технічних параметрів: Паралельно з аналізом соціальних факторів проводиться оцінка технічних параметрів проекту, таких як обсяг робіт, складність завдань та інші технічні фактори.
- Інтеграція даних: На цьому етапі дані з аналізу соціальних та технічних факторів інтегруються для створення загального зображення проекту. Це дозволяє виявити можливі взаємодії між ними та визначити, як вони впливають на тривалість проекту.
- Прогнозування часу виконання: На підставі інтегрованих даних розробляються моделі прогнозування часу виконання проекту, які враховують як соціальні, так і технічні аспекти. Ці моделі дозволяють більш точно прогнозувати часові рамки та ризики проекту.
- Управління проектом: Один із головних вигод методу полягає в можливості управляти проектом на основі отриманих даних. При виникненні проблем у комунікації чи технічних складнощах можна вчасно реагувати та коригувати плани.

3.1.3. Переваги та обмеження методу соціально-технічної послідовності

Використання методу соціально-технічної послідовності в програмному модулі для прогнозування часу виконання проєктів має свої переваги та обмеження:

Переваги:

- Краща адаптація до реальних умов: Метод дозволяє краще враховувати соціальні та технічні змінні, які можуть виникати під час проекту та впливати на його тривалість.
- Покращена комунікація: Завдяки аналізу соціальних факторів, можна виявити та вирішити проблеми у комунікації, що покращує співпрацю в команді.

• Можливість прогнозування ризиків: Метод дозволяє ідентифікувати можливі ризики, пов'язані як з технічними, так і з соціальними аспектами проекту.

Обмеження:

• Складність аналізу: Аналіз соціальних факторів може бути складним та вимагати значних ресурсів.

• Залежність від якості даних: Точність прогнозування великою мірою залежить від якості та актуальності вхідних даних.

• Необхідність спеціалізованого програмного забезпечення: Для впровадження цього методу може бути необхідне спеціалізоване програмне забезпечення.

Метод соціально-технічної послідовності є важливим інструментом в реалізації програмного модуля для прогнозування часу виконання проєктів в IT-компанії. Цей метод дозволяє інтегрувати соціальні та технічні аспекти проекту для поліпшення управління та прогнозування результатів проекту. Він може бути особливо корисним у глобальних розробницьких командах, де ефективна комунікація та розуміння взаємовідносин між членами команди важливі для успішного виконання проектів.

## 3.2. Кількісні методи дослідження в глобальній розробці проєктів

Кількісні методи дослідження є важливою складовою процесу прогнозування та управління проектами, оскільки вони дозволяють отримувати об'єктивні дані та робити інформовані рішення.

Кількісні методи дослідження є невід'ємною частиною управління проектами та прогнозуванням часу виконання проєктів. Вони дозволяють отримати об'єктивні дані про різні аспекти проекту, такі як обсяг робіт, ризики, витрати ресурсів тощо. Роль кількісних методів включає:

• Об'єктивність і точність: Кількісні методи надають можливість вимірювати та аналізувати параметри проекту чисельно, що дозволяє отримати об'єктивну картину стану справ.

- Прогнозування та планування: На основі кількісних даних можна розробляти прогнози та плани, визначати терміни виконання завдань, розподіляти ресурси та бюджет.
- Оцінка ризиків: Аналіз кількісних даних допомагає виявити потенційні ризики та їх вплив на проект, що дозволяє приймати заздалегідь заходи для їх уникнення чи зменшення.
- Підтримка рішень: Об'єктивні дані, отримані за допомогою кількісних методів, стають основою для прийняття обґрунтованих рішень в процесі управління проектом.

3.2.2. Застосування кількісних методів в програмному модулі

У реалізації програмного модуля для прогнозування часу виконання проєктів в ІТ-компанії кількісні методи дослідження використовуються для отримання об'єктивних даних про різні аспекти проектів. Нижче розглянемо деякі з них та їх застосування:

- Оцінка обсягу робіт: Для планування та прогнозування часу виконання проєкту важливо визначити обсяг робіт. Кількісні методи, такі як методи оцінки функціональності, дозволяють визначити кількість робочих одиниць, які потрібно виконати.
- Аналіз ресурсів: Для ефективного розподілу ресурсів (людських, матеріальних, фінансових) необхідно мати кількісні дані про їх доступність та витрати. Кількісні методи допомагають визначити потрібний обсяг ресурсів.
- Моделювання ризиків: За допомогою кількісних методів можна моделювати різні сценарії ризиків та їх вплив на проект, що дозволяє приймати заздалегідь заходи для зменшення ризиків.
- Оцінка вартості проекту: Кількісні методи дозволяють оцінити вартість проекту та скласти бюджет, враховуючи різні види витрат.

3.2.3. Популярні кількісні методи в глобальній розробці

У глобальній розробці проєктів існує безліч кількісних методів дослідження, і вибір методу залежить від конкретних завдань та особливостей проекту. Ось декілька популярних кількісних методів:

- Метод оцінки функціональності (Function Point Analysis): Цей метод використовується для оцінки обсягу функціональності програмного продукту на основі кількості функцій, що він виконує. Він дозволяє визначити обсяг робіт та ресурси, необхідні для розробки.
- Метод оцінки трудовитрат (COCOMO): Цей метод використовується для оцінки трудовитрат, необхідних для розробки програмного продукту. Він враховує різні аспекти проекту, такі як обсяг робіт, складність та інші чинники.
- Метод вартісного аналізу (Cost-Benefit Analysis): Цей метод допомагає визначити вартість проекту та його користь. Він використовується для оцінки ефективності проекту та прийняття рішення про його реалізацію.
- Метод аналізу ризиків (Risk Analysis): Цей метод використовується для ідентифікації та оцінки потенційних ризиків проекту. Він дозволяє приймати заходи для зменшення ризиків та підвищення ймовірності успіху проекту.

Було вивчено, як відстань між розробниками впливає на їх мережу взаємовідносин (рис. 3.1).

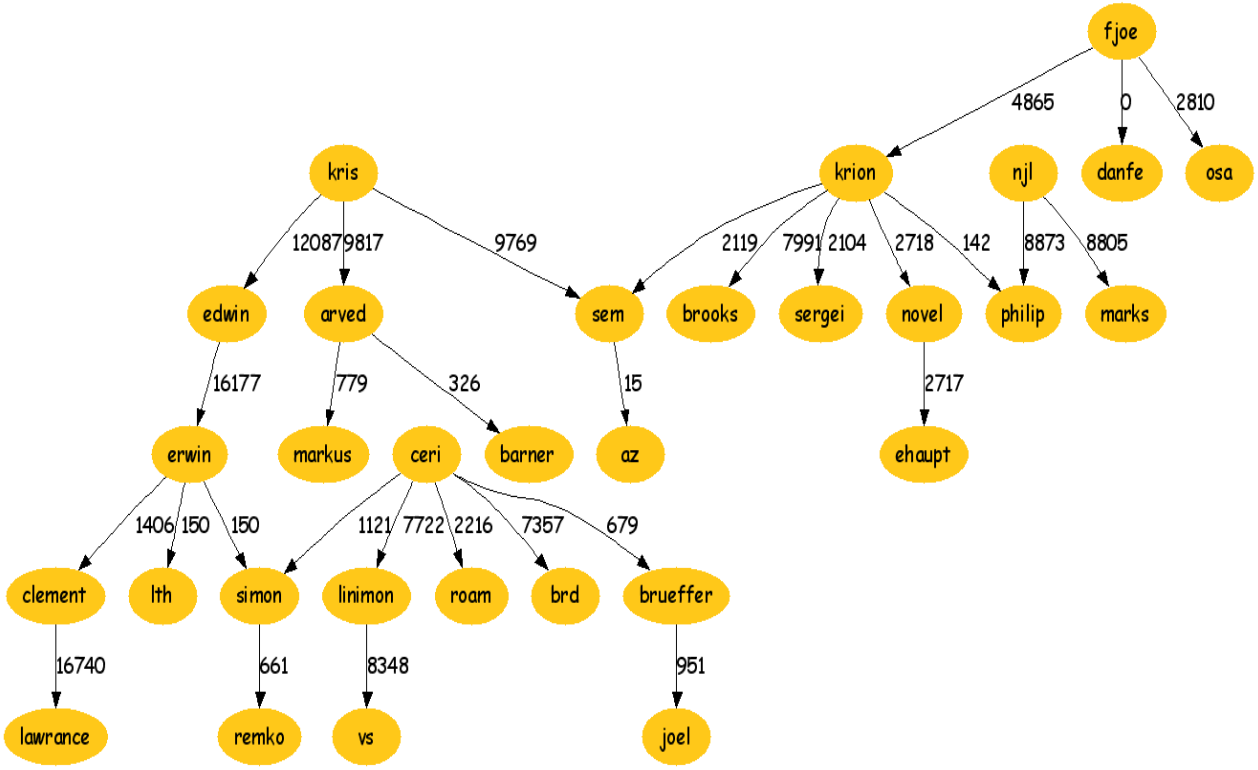


Рис. 3.1. Мережевий графік взаємозв’язків між розподіленими розробниками

## 3.3. Якісні методи дослідження розробки програмного забезпечення для прогнозування часу виконання проєктів

Якісні методи дослідження відіграють важливу роль у зборі та аналізі інформації, розумінні потреб та вимог користувачів, а також в управлінні ризиками та якістю проекту.

### 3.3.1. Роль якісних методів дослідження

Якісні методи дослідження в програмній розробці мають важливе значення, оскільки вони дозволяють збирати та аналізувати якісну інформацію, яка не завжди може бути виміряна числовими показниками. Роль якісних методів включає:

- Збір вимог користувачів: Якісні методи, такі як спостереження за користувачами, інтерв'ю та фокус-групи, допомагають зрозуміти потреби та вимоги користувачів до програмного продукту.
- Аналіз процесів розробки: Якісні методи дозволяють досліджувати процеси розробки програмного забезпечення, виявляти можливі проблеми та шляхи їх вирішення.
- Управління ризиками: Аналіз ризиків та їх впливу на проект вимагає якісного підходу. Якісні методи дозволяють виявити потенційні ризики та розробити стратегії їх управління.
- Забезпечення якості: Виявлення і аналіз дефектів та помилок в програмному продукті, врахування відгуків користувачів і тестерів, допомагає підвищити якість програмного забезпечення.

### 3.3.2. Застосування якісних методів в програмному модулі

У реалізації програмного модуля для прогнозування часу виконання проєктів в ІТ-компанії якісні методи дослідження використовуються для отримання якісної інформації про різні аспекти проектів. Нижче розглянемо деякі з них та їх застосування:

- Спостереження за користувачами: Цей метод дозволяє спостерігати за користувачами програмного продукту, аналізувати їхні дії та виявляти проблеми в інтерфейсі та функціоналі.
- Інтерв'ю зі зацікавленими сторонами: Інтерв'ю з клієнтами, менеджерами проекту, розробниками та іншими зацікавленими сторонами допомагають зрозуміти їх очікування та потреби.
- Фокус-групи: Фокус-групи можуть використовуватися для обговорення питань, пов'язаних з проектом, та збору думок і вражень учасників.
- Аналіз відгуків користувачів: Відгуки користувачів про попередні версії програмного продукту можуть допомогти виявити проблеми та покращити його якість.
- Аналіз документації та артефактів проекту: Документація проекту, така як специфікації, плани проекту, може містити важливу інформацію про його стан та вимоги.

3.3.3. Популярні якісні методи в глобальній розробці

У глобальній розробці програмного забезпечення існує безліч якісних методів дослідження. Ось декілька популярних якісних методів:

- Метод "Дизайн-мислення" (Design Thinking): Цей метод спрямований на розробку інноваційних та користувачоорієнтованих рішень. Він включає етапи спільної роботи з користувачами для розробки та тестування ідей.
- Метод "Сценарії використання" (Use Case Scenarios): Цей метод дозволяє описати різні сценарії використання програмного продукту з точки зору користувачів, що допомагає зрозуміти їхні потреби.
- Метод "SWOT-аналіз": SWOT-аналіз використовується для оцінки сильних та слабких сторін проекту, а також можливостей та загроз, що впливають на нього.
- Метод "Дизайн інтерфейсу" (User Interface Design): Цей метод спрямований на розробку інтерфейсу програмного продукту, що забезпечує зручність та задоволення користувачів.

Таблиця 3.1

Фактори, що впливають на глобальний розвиток [5]

| Фактори | Ступінь впливу | | | |
|---|---|---|---|---|
| | Дуже високо | Високий | Середній | Низький |
| Компетенції розробників | 9% | 87% | 3% | 1% |
| Структура команд | | 2% | 2% | 96% |
| Зв'язок | 95% | 3% | 2% | |
| Відстані | | 1% | 2% | 97% |
| Культурні відмінності | | 4% | 1% | 95% |
| Час | 3% | 1% | | 96% |
| Самомотивація | 11% | 57% | 32% | |

Для дослідження цих гіпотез та для ілюстрації взаємозв'язку між змінними використовувались методи графічного моделювання (рис. 3.2).

Вузли на рис. 3.2 представляють змінні, а посилання – часткову кореляцію між ними. Чорний колір вказує на позитивну часткову кореляцію, тоді як сірий – на негативну часткову кореляцію. Товщина з'єднання вказує на його важливість.

Під час аналізу відповідей у дослідженні з'ясовано, що деталі, що стосуються конкретних особистісних аспектів та обґрунтованості проектних рішень, зазвичай відповідають на питання з певною послідовністю, яка не вимагає подальшого глибокого аналізу. Однак, було ідентифіковано сім ключових чинників, які мають значний вплив на ефективність роботи команди в умовах міжнародної розробки ПЗ. Ці чинники охоплюють широкий спектр елементів, від міжкультурної комунікації до управління проектами та технологічних процесів.

На графічній моделі, представленій на рисунку 3.3, відтворено систему взаємодії цих чинників. Модель ілюструє, як кожен з факторів співвідноситься з іншими та як вони колективно впливають на загальну продуктивність. Ступінь впливу кожного чинника (P) було вираховано за допомогою статистичного методу найменших квадратів, що дозволило квантифікувати взаємозв'язки між ними та оцінити їх вплив на процес розробки. Ця модель стала інструментом для

визначення ключових точок втручання для покращення взаємодії в команді та оптимізації процесу розробки на глобальному рівні.

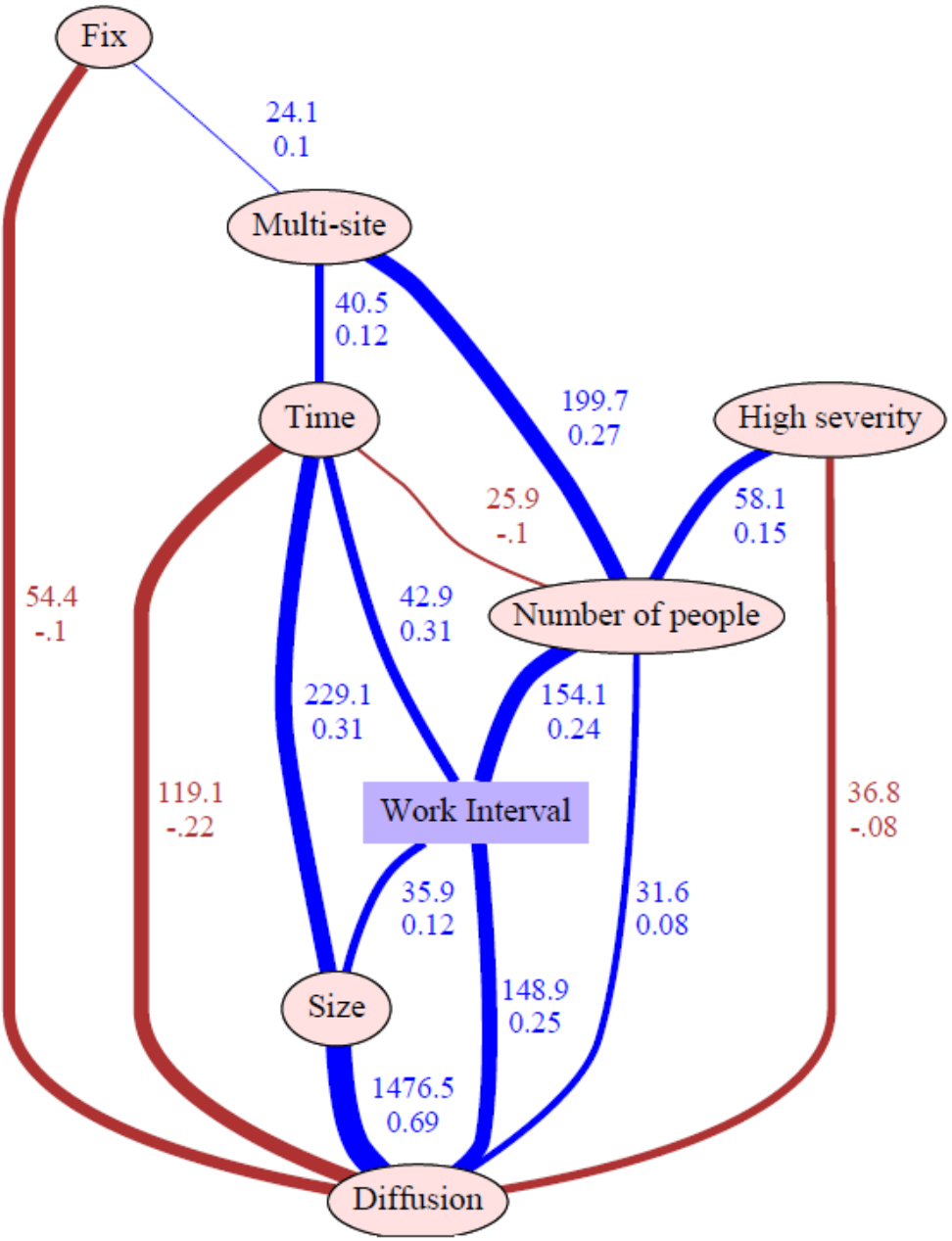


Рис. 3.2. Графічна модель робочого інтервалу [25]

Кількісні дані, пов’язані з обґрунтованістю конструкції та визначенням імовірно пов’язаних особистих питань, на які респонденти відповідали послідовно, і їх можна ігнорувати. Визначено сім різних факторів, які так чи інакше впливають на команду в глобальній розробці програмного забезпечення.

На рисунку 3.3 показана модель, що відображає взаємозв'язок між сімома різними факторами та ступенем впливу (P), розрахованим методом найменших квадратів.

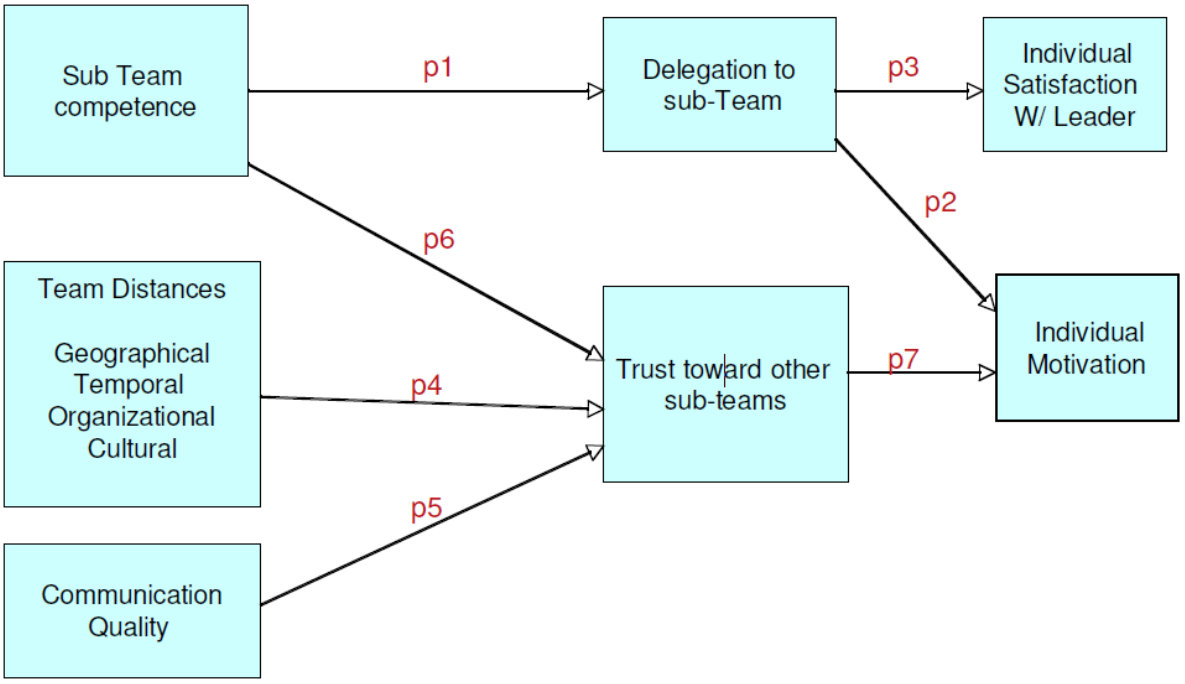


Рис. 3.3. Модель зв’язків між факторами проєктів [26]

3.4. Описання системи керування контентом проєктів

Система керування контентом проєктів є важливим компонентом будь-якої розробки програмного забезпечення, оскільки вона допомагає забезпечити ефективне керування проєктами, зберігання та обмін необхідною інформацією. Опис системи керування контентом проєктів повинен включати докладний огляд її функціональності та можливостей.

Основні функції системи керування контентом проєктів:

1. Ведення декількох проєктів: Система має забезпечувати можливість одночасного керування декількома проєктами. Це дозволяє командам ефективно організовувати свою роботу та ресурси.

2. Гнучка система доступу, заснована на ролях: Для забезпечення безпеки та конфіденційності інформації, система повинна мати систему керування доступом, яка базується на ролях користувачів. Це означає, що кожний користувач має певну роль і повноваження в системі, і вони обмежені та контрольовані.

3. Система відслідковування помилок: Для підтримки якості та ефективності розробки, система повинна включати інструменти для відслідковування та управління помилками та проблемами в проєкті. Це допомагає забезпечити своєчасну виправлення помилок та підвищити продуктивність команди.

4. Ведення новин проєкту, документів і керування файлами: Система повинна дозволяти ведення новин та оголошень проєкту, а також забезпечувати можливість зберігання та обміну документами та файлами, пов'язаними з проєктом.

5. Оповіщення про зміни: Для забезпечення ефективної комунікації та інформування користувачів проєкту система має включати механізми оповіщення про зміни, оновлення та події в системі.

6. Можливість самостійної реєстрації нових користувачів: Для зручності користувачів система повинна дозволяти новим користувачам самостійно реєструватися та отримувати доступ до необхідної інформації та функціоналу.

7. Багатомовний інтерфейс (у тому числі російська): Система повинна підтримувати багатомовний інтерфейс для забезпечення зручності користувачів з різних країн та мовних груп. Наявність російської мови є важливою для користувачів з російськомовних регіонів.

8. Система нагадування: Для підтримки строків та завдань система повинна включати функцію нагадування, яка допомагає користувачам вчасно виконувати завдання та контролювати процеси в проєкті.

Описана система керування контентом проєктів є важливою складовою для успішної реалізації програмного проєкту. Вона забезпечує зручний та ефективний спосіб керування інформацією та ресурсами, що допомагає команді досягти поставлених цілей та завдань.

Програмний засіб був розроблений на прикладі організації, яка має таку структуру, яку зображено на рисунку 3.4.

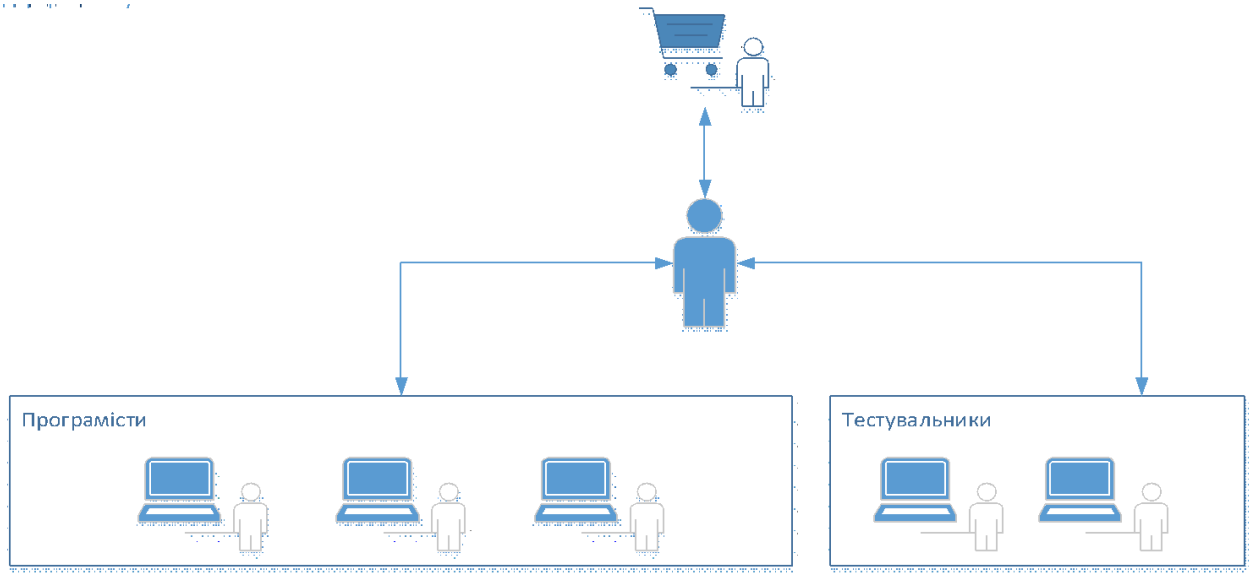


Рис. 3.4. Ієрархія проєктної групи

На рисунку зображено ієрархічну структуру IT-департаменту з роллю керівника проєктів, який координує роботу між програмістами та тестувальниками. Керівник проєктів відіграє ключову роль, керуючи всіма аспектами проєкту, включаючи розподіл завдань і нагляд за процесами розробки та тестування.

Розширений опис може включати наступні кроки та функціональні можливості:

1. Реєстрація нових користувачів та управління ролями:

• Система дозволяє замовникам реєструватися самостійно, після чого вони можуть переглядати загальну інформацію про проєкти.

• Менеджери проєктів мають можливість створювати нові проєкти, призначати програмістів та тестувальників до завдань.

2. Ведення проєктів:

• Для кожного проєкту ведеться облік завдань, помилок, новин, документації та файлів.

• Встановлення гнучкої системи доступу, де кожен користувач має доступ до інформації згідно зі своєю роллю.

3. Контроль процесу розробки:

• Програмісти отримують задачі від менеджерів проєктів і виконують їх, маючи доступ лише до відповідної інформації.

• Тестувальники, також отримуючи завдання від менеджерів проєктів, перевіряють програмне забезпечення на наявність помилок і звітують про результати.

4. Оповіщення та система нагадувань:

• Користувачі отримують оповіщення про зміни в проєктах, нові завдання, помилки чи важливі новини.

• Система нагадувань допомагає тримати роботу в термінах і попереджати про майбутні дедлайни.

5. Багатомовний інтерфейс:

• Інтерфейс програми підтримує кілька мов, у тому числі російську, що робить її доступною для міжнародних команд.

6. Звітність та аналіз:

• Менеджери проєктів можуть генерувати звіти про стан проєктів, виконання завдань та продуктивність команди.

• Аналітичні інструменти можуть допомагати визначати "вузькі місця" в процесах та вносити корективи для підвищення ефективності.

Під час розробки були додатково використані такі компоненти System.Windows.Controls.Ribbon – це компонент Microsoft.ASPNET Framework 4.5.1, який відповідає за відображення Ribbon панелі на формі:

• System.Windows.Controls.Ribbon: Це клас, який знаходиться в просторі імен WPF, який дозволяє розробникам додавати інтерфейс користувача Ribbon до своїх додатків. Ribbon - це тип панелі інструментів, який представляє команди, згруповані за категоріями. Ribbon UI є популярним в Microsoft Office продуктах, де він допомагає організувати команди таким чином, що їх легше знайти та використовувати.

• .NET Framework 4.5.1: Це версія .NET Framework, яка включає ряд поліпшень у продуктивності, підтримку новітніх стандартів і більш зручні можливості для розробників. Ця версія framework містить багато класів та бібліотек, які дозволяють створювати як настільні, так і веб-додатки.

- Ribbon панель на формі: Ribbon UI в WPF дозволяє розробникам реалізувати власний Ribbon інтерфейс у своїх застосунках. Це робить інтерфейс користувача більш інтуїтивним та легшим для використання, особливо в складних додатках з багатьма функціями.

Тож, з огляду на виправлення, можна сказати, що під час розробки додатка на базі .NET Framework 4.5.1 було використано System.Windows.Controls.Ribbon для створення Ribbon інтерфейсу, що поліпшує навігацію та доступність функцій у настільному додатку.

Діаграма класів клієнтської частини:

- Клас App може бути головним вхідним пунктом додатку, який керує запуском додатка.
- Клас MainWindow може представляти основне вікно користувача, і може включати компоненти, такі як RibbonWindow, що є інтерфейсом Ribbon.
- Create_task, CreateProject, Open_task, OpenProject зазвичай є вікнами або діалогами, що дозволяють користувачу створювати нові завдання чи проекти, а також відкривати існуючі.
- Project_History, project_information, і Project_Stan могли б бути спеціалізованими класами для управління історією проектів, інформацією та статусами.
- Settings може бути класом, який керує конфігураціями та налаштуваннями додатку.
- Task та user_information могли б представляти класи для управління інформацією про завдання та користувачів відповідно.

Діаграма класів серверної частини:

- App аналогічно може бути вхідним пунктом для серверного додатку.
- Backup та Backup2 можуть бути класами, що відповідають за резервне копіювання даних.
- ChangePassword та CreatePassword можуть управляти аутентифікацією та безпекою.

- MainWindow може бути серверним аналогом клієнтського MainWindow, можливо, для адміністрування.
- Network може бути відповідальним за мережеві операції.
- Project і Users можуть управляти інформацією про проекти та користувачів відповідно.
- Settings може використовуватися для конфігурації серверних налаштувань.
- Stop_Server та Start_Server можуть бути командами для управління станом сервера.

Зверніть увагу, що ці припущення базуються на загальних назвах класів, які часто використовуються у програмному забезпеченні, і можуть не відображати точну структуру або взаємодію вашого конкретного додатку. Для точного опису діаграми класів потрібно більше контексту про додаток та його функціональність.

Схема алгоритму роботи з програмою керівника *IT*-департаменту, яку наведено у Додаток А, вимагає формування наступної архітектури ПЗ (рис. 3.5).

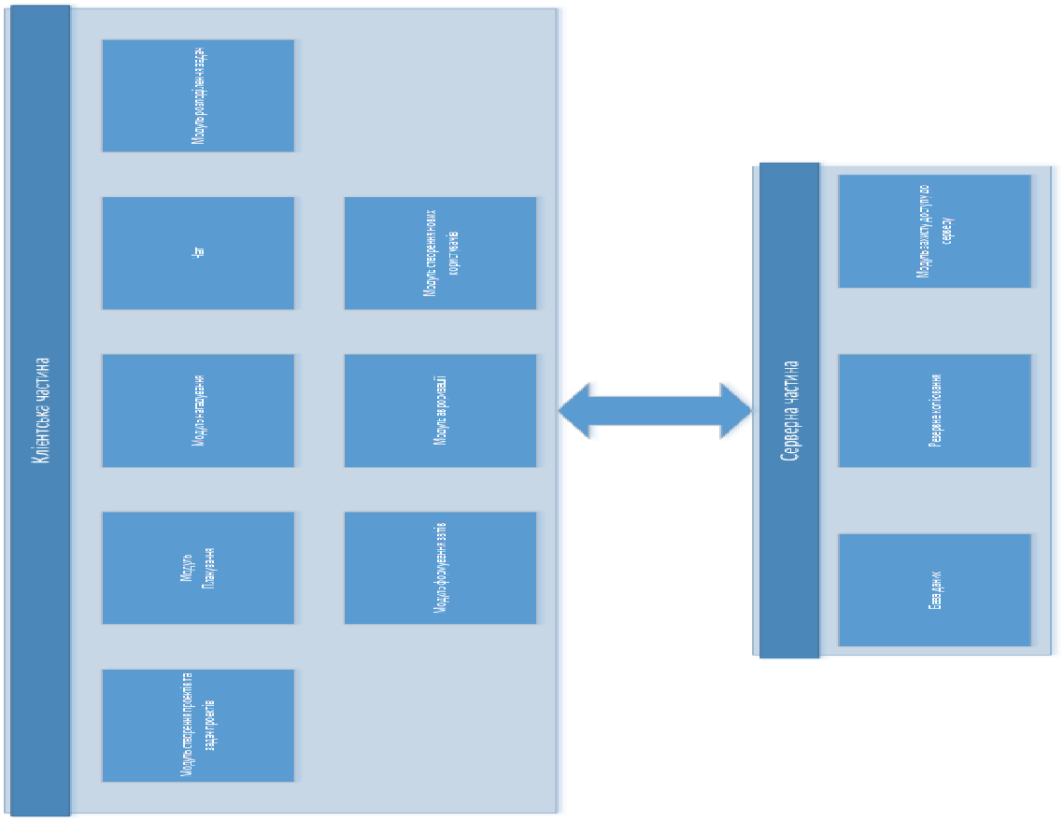


Рис. 3.5. Архітектура програмного забезпечення

Зображення показує архітектуру програмного забезпечення, яке складається з клієнтської та серверної частин. Я не можу бачити зміст зображення, але я можу надати вам уявлення про те, як зазвичай структуруються подібні архітектури програмного забезпечення на основі стандартних практик.

Клієнтська Частина: Клієнтська частина архітектури зазвичай містить наступні модулі:

- Модуль створення проєктів/завдань: Дозволяє користувачам створювати нові проекти або завдання, визначати їх параметри та налаштування.
- Модуль Планування: Використовується для планування ресурсів, часу та інших аспектів проектів.
- Модуль Аналітики: Надає інструменти для аналізу даних проекту, включно з відстеженням прогресу та ефективності.
- Чат: Забезпечує можливість спілкування між членами команди в реальному часі.
- Модуль розподілу навантаження: Для розподілу завдань та відповідальностей між співробітниками.
- Модуль Документації: Управління документами проекту, включно з версіями та затвердженнями.
- Модуль Авторизації: Контроль доступу до різних частин системи на основі ролей та прав користувачів.
- Модуль Сповіщень користувачів: Відправлення повідомлень користувачам про важливі події або зміни в проектах.

Серверна Частина: Серверна частина забезпечує функціональність, необхідну для підтримки клієнтських операцій, і зазвичай включає:

- База даних: Зберігає всі дані проекту, користувачів, завдань та іншу важливу інформацію.
- Сервер додатків: Виконує бізнес-логіку програми, обробляє запити від клієнтської частини та забезпечує інтеграцію з базою даних.
- Модуль доступу до серверу: Керує з'єднаннями між клієнтською та серверною частинами, включно з аутентифікацією та шифруванням.

Центральна стрілка, яка з'єднує клієнтську та серверну частини, символізує мережеву взаємодію між ними, де клієнтські запити обробляються сервером, який надає відповідні дані та сервіси.

Опис схеми-алгоритму:

1. Початок.
2. Введення ідентифікатора проекту (id_proj).
3. Організація пошуку за введеним ідентифікатором.
4. Перевірка на наявність проекту в базі даних.
5. Якщо проект не знайдено:

- Вивід повідомлення про відсутність проекту.
- Повернення до кроку введення ідентифікатора.

6. Якщо проект знайдено:

- Надання списку корегуючих робіт.

7. Перевірка на наявність графіків:

- Якщо графіків немає, процедура виведення форми розподілу робіт.
- Якщо графіки є, перевірка форми затверджених графіків з проектами.

8. Визначення загальної кореляції робіт з історичними даними.
9. Фільтрування за певним рівнем кореляції (наприклад, > 0.7).
10. Оновлення структури прогнозування часом.
11. Кінець.

Ця схема може бути частиною більшого бізнес-процесу або програмного додатку для управління IT-проектами, де використовуються історичні дані для планування та корегування робіт.

Під час розробки були додатково використані такі компоненти *System.Windows.Controls.Ribbon* – це компонент *Microsoft.ASPNET Framework* 4.5.1, який відповідає за відображення *Ribbon* панелі на формі.

Діаграма класів клієнтської частини (рис. 3.6).

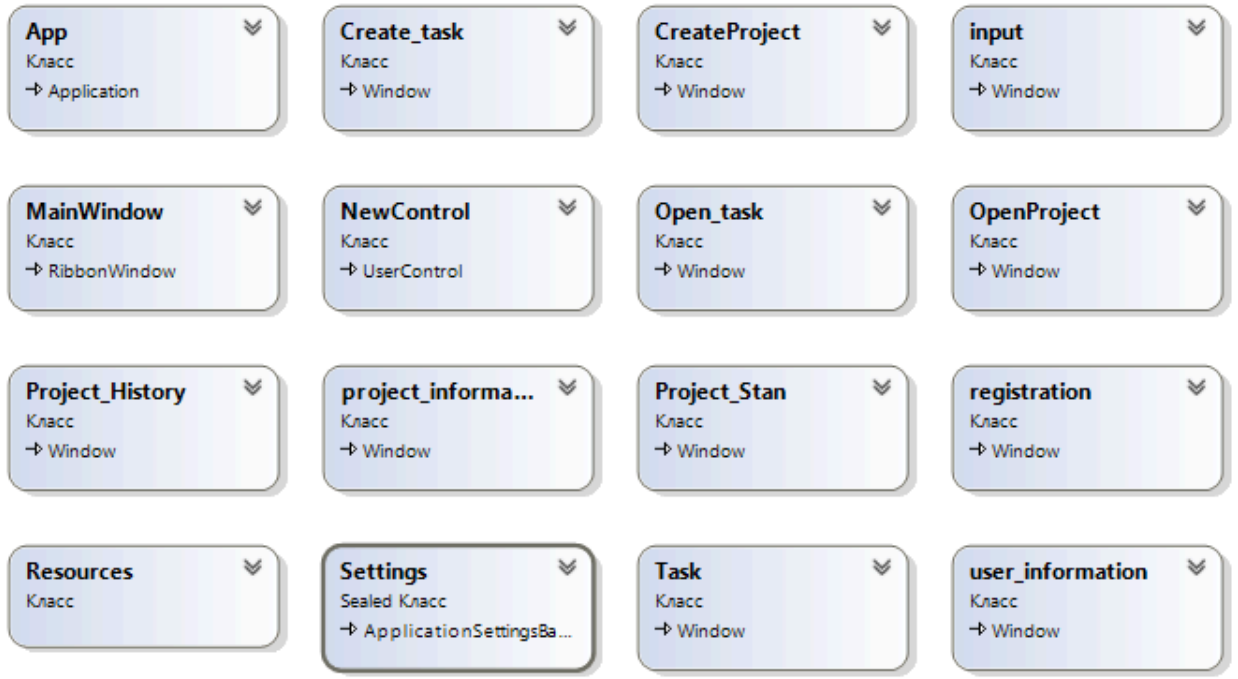


Рис. 3.6. Діаграма класів клієнтської частини

Діаграма класів серверної частини (рис. 3.7).

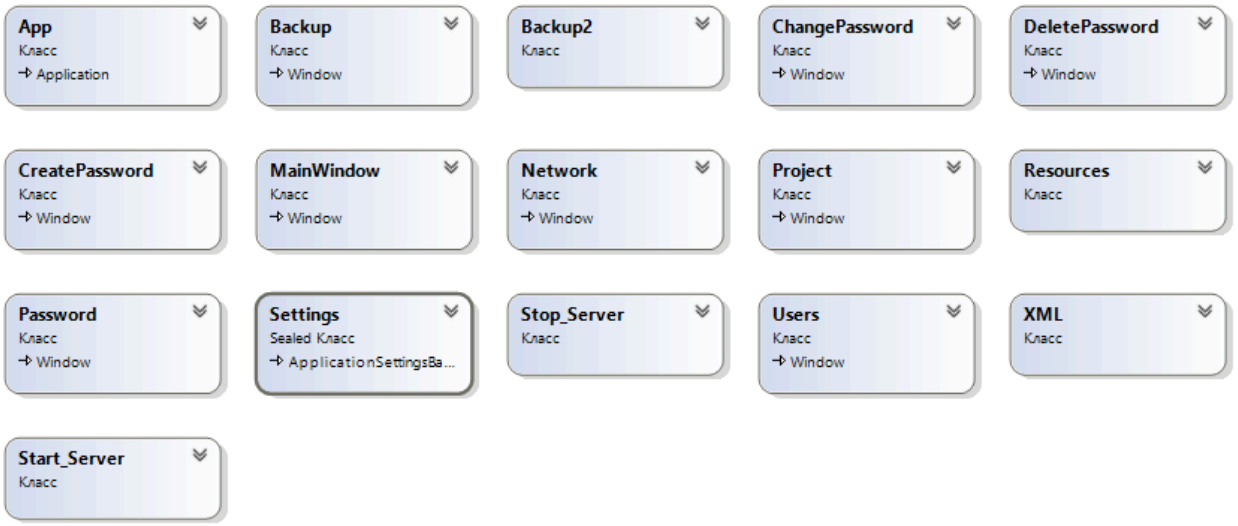


Рис. 3.7. Діаграма класів серверної частини

База даних, розроблена з використанням системи управління базами даних (СУБД) MSSQL, має ряд значущих переваг, що роблять її популярним вибором для бізнес-додатків, включно з системами прогнозування часу виконання проєктів. Ось деякі з ключових переваг MSSQL:

1. Продуктивність та масштабованість: MSSQL відомий своєю високою продуктивністю та здатністю обробляти великі обсяги даних. СУБД оптимізована для роботи з різними розмірами баз даних, від невеликих до величезних, забезпечуючи швидку обробку запитів та транзакцій. Масштабованість системи дозволяє легко адаптуватися до зростаючих вимог проекту.

2. Надійність та доступність: MSSQL має вбудовані засоби для забезпечення високої доступності та надійності. Механізми реплікації даних, резервного копіювання та відновлення після збоїв допомагають мінімізувати час простою та забезпечують безперервну роботу систем. Функції моніторингу та оповіщення допомагають оперативно реагувати на проблеми та забезпечувати стабільність роботи.

3. Безпека: MSSQL надає розгорнуті можливості управління доступом та аудитом, дозволяючи детально контролювати, хто має доступ до даних та які операції з ними виконуються. Це включає підтримку шифрування, політик безпеки, а також інтеграцію з Windows Authentication для забезпечення багаторівневої безпеки.

4. Інтегрованість із іншими продуктами Microsoft: Оскільки MSSQL є частиною екосистеми продуктів Microsoft, вона забезпечує відмінну інтеграцію з іншими продуктами та сервісами, такими як Microsoft Azure, Visual Studio, і .NET Framework. Це спрощує розробку, розгортання та управління базами даних.

5. Інструменти для розробки та управління: MSSQL поставляється з потужними інструментами для розробки, адміністрування та оптимізації баз даних, такими як SQL Server Management Studio та SQL Server Profiler. Ці інструменти роблять процес створення, налагодження та управління базами даних зручним та ефективним.

6. Підтримка транзакцій і ACID властивостей: MSSQL забезпечує надійну підтримку транзакцій та дотримання ACID властивостей (атомарність, консистентність, ізоляція, та стійкість), що критично важливо для забезпечення цілісності даних та здійснення складних операцій.

7. Розширена аналітика та звітність: З використанням SQL Server Analysis Services та SQL Server Reporting Services, MSSQL надає розширені можливості для аналітики даних та генерації звітів, що дозволяє отримувати глибокі інсайти та підтримувати прийняття обґрунтованих рішень.

Обравши MSSQL як платформу для бази даних системи прогнозування часу виконання проєктів, організації отримують потужний, надійний та гнучкий інструмент, який сприяє підвищенню продуктивності, забезпеченню безпеки даних і підтримці ефективного виконання проектів.

## 3.5. Інтерфейс клієнтської частини програмного продукту

Приклад використання клієнтської частини в ролі керівника *IT*-департаменту. На рис. 3.8 зображено головне вікно програми.

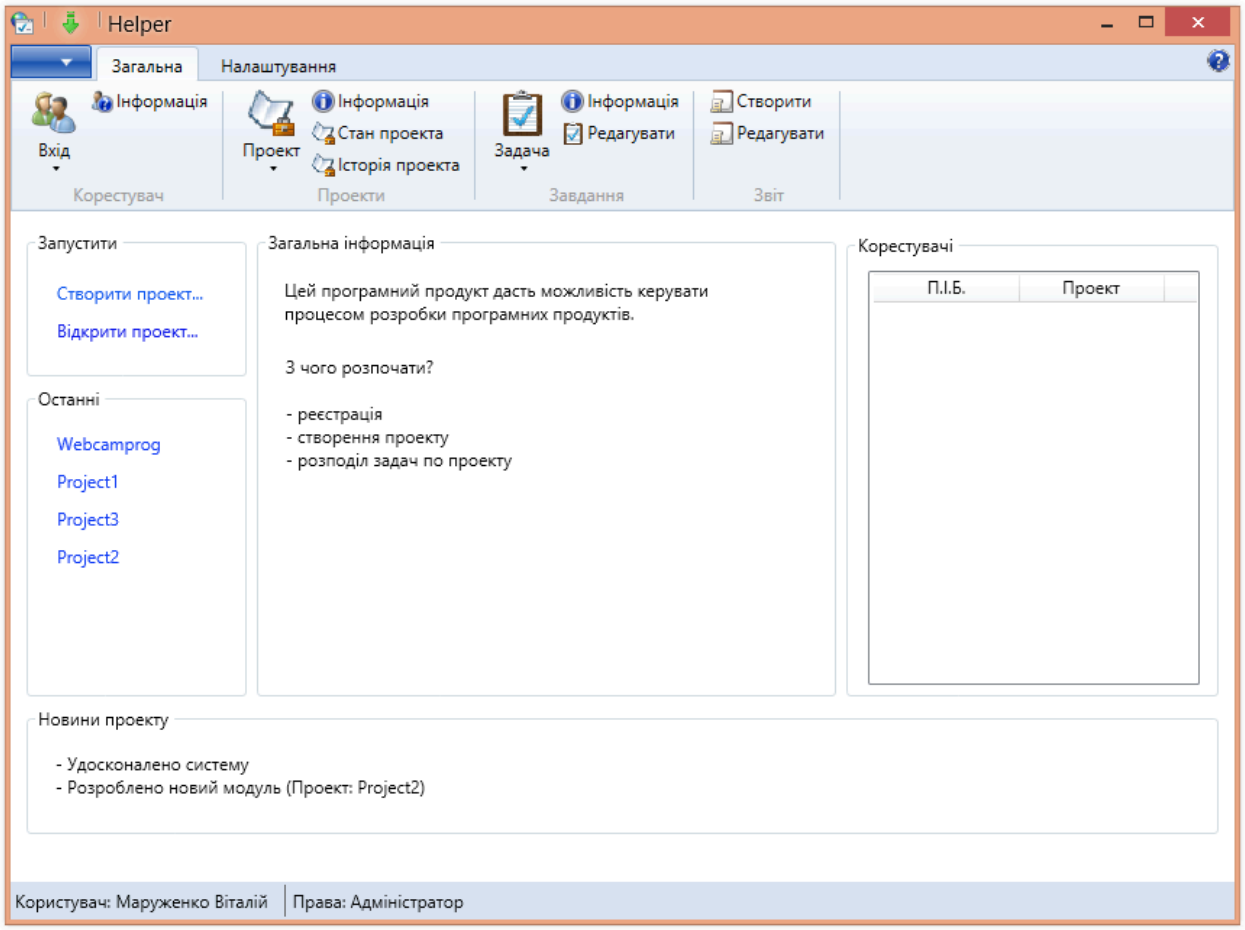


Рис. 3.8. Головне вікно програми (клієнтська частина)

Специфікація програмних модулів визначає структуру та функціональність кожного компонента програми. Ось специфікація на основі наданої таблиці:

1. Головна форма:

• Опис модуля: Центральний інтерфейс програми, що містить навігаційне меню з кнопками, які представляють основні функції та можливості програми. Використовується для швидкого доступу та керування різними операціями в рамках системи. Вхідні дані для цього модуля включають вибір операції, яку користувач бажає здійснити.

2. Додати/видалити макет:

• Опис модуля: Функціональний компонент, який дозволяє користувачам реєструвати нові або видаляти існуючі макети виробів. Вхідні дані включають назву макета, його версію, та FTP-адресу для вихідних файлів. Цей модуль забезпечує легке управління макетами, що є важливим для збереження організованої інформації про продукти.

3. Додати/видалити проектанта:

• Опис модуля: Модуль для реєстрації або видалення інформації про проектантів. Вхідні дані включають ім'я, по-батькові, прізвище, та email-адресу проектанта. Цей компонент важливий для управління інформацією про учасників проекту та їх внески.

4. Додати/видалити виріб:

• Опис модуля: Дозволяє користувачам реєструвати нові вироби або видаляти існуючі, а також управляти версіями виробів. Вхідні дані включають назву виробу, версію, дату створення, інформацію про макет, проектанта, ліцензію, та FTP-адресу вихідних файлів. Це дозволяє зберігати актуальну та організовану інформацію про всі вироби.

5. Пошук виробу за датою:

• Опис модуля: Функція пошуку, яка виводить список виробів, створених між вказаними датами. Вхідні дані включають початкову і кінцеву дати,

а вихідні - деталі виробів, такі як назва, версія, дата, FTP-адреса, проектант та макет.

6. Пошук виробу за автором:

• Опис модуля: Дозволяє вивести список всіх виробів, створених конкретним проектантом. Вхідні дані - ім'я та прізвище автора, вихідні - деталізована інформація про вироби, включаючи назву ПЗ, версію, дату, ліцензію, FTP-адресу, імена проектантів та їх email-адреси.

7. Пошук виробу за назвою:

• Опис модуля: Забезпечує вивід всіх виробів з заданою назвою. Вхідні дані - назва виробу, вихідні - інформація про виріб, включаючи назву, версію, дату, FTP-адресу, проектанта, макет і FTP-адресу макета.

8. Пошук проектанта:

• Опис модуля: Модуль для виведення інформації про всі вироби, створені заданим проектантом. Вхідні дані - прізвище проектанта, вихідні дані включають ім'я, по-батькові, прізвище, та email-адресу проектанта.

Кожен з цих модулів відіграє свою специфічну роль у загальній архітектурі системи прогнозування часу виконання проєктів, забезпечуючи необхідний функціонал для збору, управління, пошуку та аналізу даних. Вони разом формують комплексне рішення, яке може враховувати різноманітні аспекти проектної діяльності і надавати користувачам зручні та ефективні інструменти для планування та управління.

Зображення демонструє головне вікно програми, яке використовується керівником IT-департаменту для ефективного управління проектами, задачами та командами. Ось детальний опис використання цього інтерфейсу:

Інтерфейс з Ribbon Панеллю Головне вікно програми включає інноваційний Ribbon інтерфейс, що значно спрощує навігацію. Ribbon організовує інструменти та функції в зручні вкладки і групи, що дозволяє швидко переключатися між різними категоріями задач без потреби прокручування численних меню.

Категорії Ribbon Панелі

- Проєкти: Секція, де керівник може створювати нові проєкти, переглядати існуючі, вносити зміни та відстежувати історію змін.
- Задачі: Група, що уможливлює детальне управління задачами проєкту, від їх створення до відстеження їх виконання та реалізації.
- Звіти: Розділ для формування звітів, який дозволяє керівнику отримувати аналітичну інформацію та оцінювати продуктивність проєктів та команд.

Статичні Поля

- Новини проєкту: Це поле динамічно оновлюється та показує останні події та зміни по всіх проєктах, надаючи керівнику актуальну інформацію для прийняття оперативних рішень.
- Останні: Тут відображаються назви шести останніх проєктів, над якими проводилась робота, забезпечуючи швидкий доступ до найновіших діяльностей.

Використання Програми Керівник може ефективно керувати ресурсами та пріоритетами, використовуючи цей інтерфейс. Наприклад, вони можуть відразу побачити стан всіх проєктів, перевірити нові завдання, що були додані в проєкти, і переглянути останні новини. Це дозволяє зосередитись на управлінні високого рівня, мінімізуючи потребу в глибокому зануренні в деталі кожного проекту.

Переваги для Керівника ІТ-департаменту Головна перевага цієї системи полягає в тому, що вона забезпечує централізоване та інтуїтивно зрозуміле середовище для управління всіма аспектами роботи ІТ-департаменту. Керівник отримує повний контроль над процесами, що відбуваються в департаменті, і може відповідно реагувати на зміни в реальному часі.

На рис. 3.9  користувач водить логін та пароль для доступу в систему. Якщо користувач вперше використовує програму, йому потрібно зареєструватися.

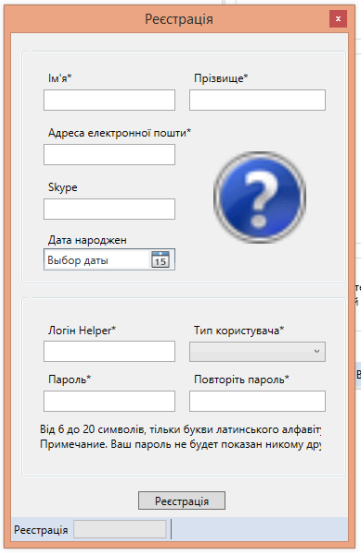


Рис. 3.9. Вікно авторизації

Завантажене вами зображення, ймовірно, відображає діалогове вікно реєстрації для програми, яке використовується новими користувачами для створення облікового запису. Ось детальний опис процесу реєстрації на основі наведеної інформації:

Діалогове Вікно Реєстрації У цьому вікні користувачам пропонується ввести наступну інформацію для створення нового облікового запису:

- Ім'я: Користувач повинен ввести своє реальне ім'я, яке буде використовуватися для ідентифікації в системі.
- Прізвище: Подібно до імені, це поле для вводу фамілії користувача.
- Адреса електронної пошти: Необхідно вказати дійсну електронну адресу, яка може використовуватися для комунікації та відновлення паролю.
- Дата народження: Запитується для додаткової перевірки особи користувача або для забезпечення відповідності віковим обмеженням.
- Логін Helper: Це унікальний ідентифікатор користувача в системі.
- Пароль: Поле для введення секретного пароля, який буде використовуватися для доступу до облікового запису.
- Підтвердження пароля: Повторне введення пароля для забезпечення його правильності.

На додаток, в нижній частині вікна знаходиться зауваження про необхідність використання не менше 20 символів для паролю та наявність хоча б однієї літери у верхньому регістрі.

Процес Входу та Реєстрації На основному інтерфейсі програми використовується Ribbon панель, яка дозволяє легко переходити до необхідних функцій. Для входу в систему існуючим користувачам необхідно ввести свої облікові дані. Якщо користувач є новим, він може перейти до реєстрації, вибравши опцію «Створити нового» в контекстному меню під вкладкою «Вхід» на Ribbon панелі, після чого відкриється вікно реєстрації.

Цей інтерфейс покликаний зробити процес входу та реєстрації інтуїтивно зрозумілим та безпечним, забезпечуючи гнучкий, але контрольований доступ до різних рівнів системи в залежності від ролі користувача.

Під час створення проєкту потрібно вибрати користувачів які будуть приймати участь в даному проєкті (рис. 3.10).

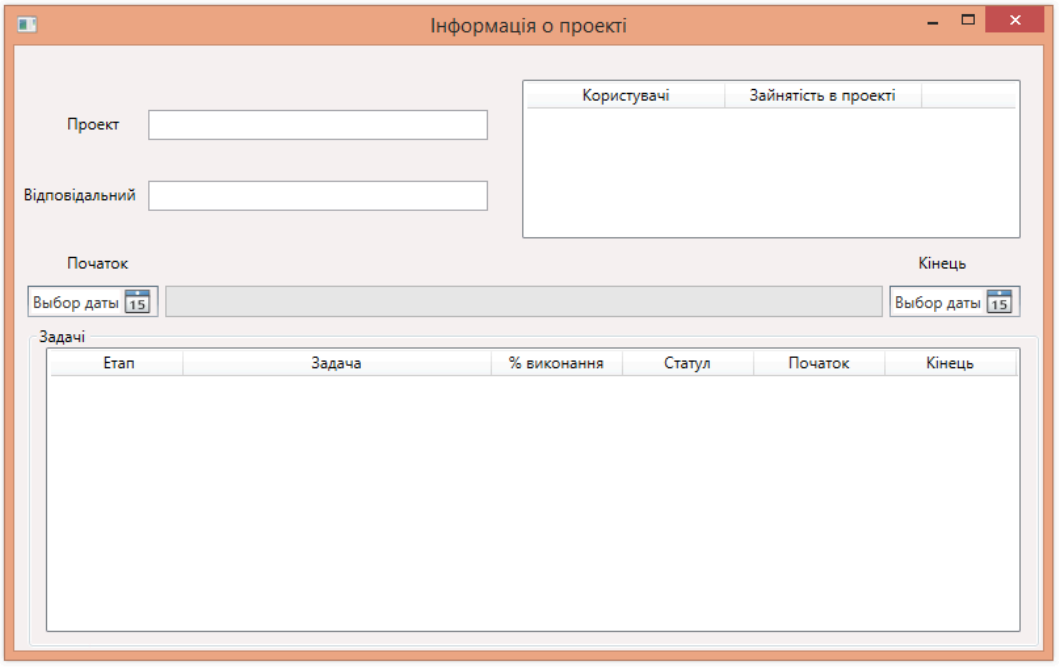


Рис. 3.10. Вікно формування інформації о проєкті

За допомогою часових рамок є можливість зафіксувати час на виконання поставленої задачі, та вказати крайній термін виконання. В поле задача заноситься

опис задачі яку потрібно виконати. В поле коментар заноситься додаткова інформація яка допоможе користувачеві зменшити час виконання даної задачі. Повідомлення може містити деталі задачі та будь-які спеціальні інструкції або рекомендації, що містяться в полі коментар. Система також автоматично додасть задачу до календаря користувача або до списку активних завдань, щоб він міг легко відстежувати свої обов'язки та управляти своїм часом. У разі потреби користувач може запитати додаткову інформацію або уточнення щодо задачі, використовуючи вбудовані засоби спілкування, такі як чат або електронні листи.

Коли задача виконується, користувач маркує її як завершену в системі, що дає можливість керівництву або відповідальним особам переглянути виконану роботу та підтвердити її закінчення. Це також може запустити процес перевірки або тестування, залежно від природи проекту. Система може надавати звіти про статус виконання задач, що дозволяє керівництву відстежувати прогрес проекту в реальному часі та приймати обґрунтовані рішення щодо розподілу ресурсів та пріоритетів.

На рис. 3.11 зображено приклад оновлення клієнтської версії, якщо версії не співпадають, виконується оновлення клієнтської частини програми.

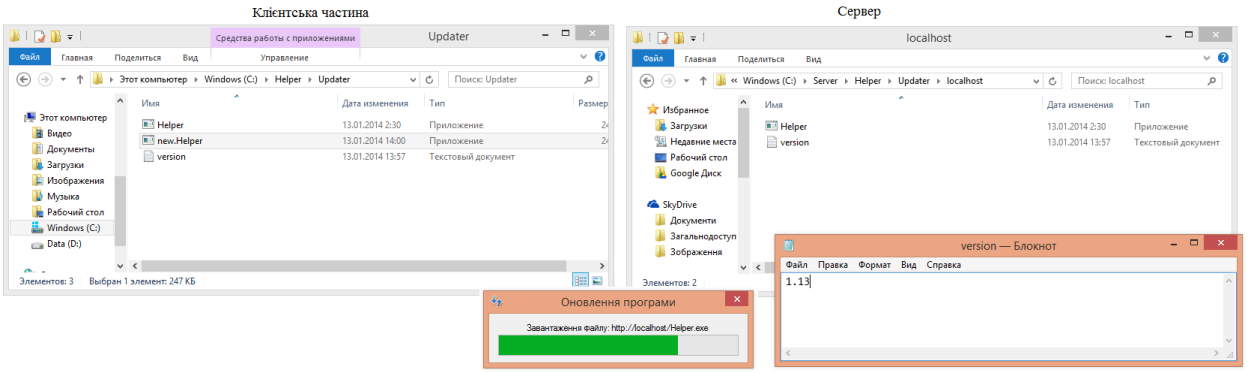


Рис. 3.11. Приклад оновлення клієнтської частини

На зображенні, яке ви надали, ймовірно, показано інтерфейси клієнтської та серверної частини системи управління проектами. Оскільки я не можу побачити зображення, я викладу концепцію, що може бути представлена:

Оновлення Клієнтської Версії Коли користувач відкриває клієнтську версію програми, система автоматично перевіряє наявність оновлень. Якщо поточна версія клієнта не відповідає останній версії на сервері, запускається процес оновлення, що може включати завантаження та інсталяцію останніх патчів, функціональних доповнень чи виправлень безпеки. Це забезпечує, що всі користувачі працюють з останньою та найбезпечнішою версією програми.

Розрахунок Часу Виконання Проекту Принципи розрахунку часу виконання проекту можуть включати аналіз історичних даних про подібні проекти, оцінку робочих годин, що потрібні для кожного етапу проекту, та врахування ресурсів, доступних для команди. Система може використовувати алгоритми для прогнозування, такі як метод критичного шляху або PERT (Program Evaluation and Review Technique), для визначення оптимального графіка виконання завдань і виявлення можливих ризиків затримки.

Додатково, система може надавати інструменти для відстеження прогресу проекту в реальному часі, дозволяючи керівництву та учасникам проекту своєчасно реагувати на відхилення від плану та вносити корективи для збереження графіка проекту в межах встановлених термінів.

## 3.6. Висновки до розділу

Глобальна розробка програмного забезпечення становить складне поле діяльності, де одновимірний аналіз є недостатнім для повного розуміння всіх аспектів. Замість цього, необхідно звертатися до багатовимірних досліджень, які включають різноманітність наукових підходів, контекстів, тематик та методологій. Такий широкий спектр дозволяє адаптуватися до змінних умов та використовувати наукові ресурси з максимальною ефективністю. Через використання комбінації кількісних та якісних методів дослідження, проект здатний охопити проблему з різних точок зору.

В розділі було описано розробку інструменту управління проектами, який базується на аналізі даних з попередніх проектів для прогнозування термінів

виконання майбутніх проектів. Це включає розробку трьох ключових компонентів: архітектури, користувацького інтерфейсу та бази даних, які були спроєктовані, реалізовані та інтегровані згідно з розробленими схемами та прототипами.

Робота також демонструє значущість використання сучасних інструментів та методологій розробки, що забезпечують можливість одночасної роботи декількох клієнтів. Наголошено, що впроваджена система управління процесом розробки позитивно впливає на якість кінцевого продукту, а моніторинг та контроль процесу розробки програмного забезпечення є однією з найбільш актуальних задач в індустрії програмного забезпечення на сьогоднішній день.

Головною особливістю ефективного інструменту прогнозування часу виконання проєктів є його вдале вбудовування в загальну систему управління проєктами, що включає такі аспекти:

1. Інтеграція з іншими системами: Інструмент прогнозування повинен легко інтегруватися з існуючими системами управління проектами, обліку часу, ресурсним плануванням та іншими інструментами. Це забезпечує плавний обмін даними та уникнення необхідності введення даних вручну, знижуючи ризик помилок та підвищуючи ефективність процесів.

2. Гнучкість та конфігурованість: Інструмент повинен бути достатньо гнучким, щоб адаптуватися до різних методологій управління проектами та специфічних вимог конкретного проєкту. Користувачі повинні мати можливість налаштовувати параметри прогнозування, критерії оцінки та інші аспекти для досягнення найбільш точних та відповідних результатів.

3. Автоматизація та підтримка рішень: Сучасні системи прогнозування часу виконання використовують автоматизацію для збору даних, обробки інформації та надання рекомендацій. Це включає використання алгоритмів машинного навчання та статистичного аналізу для виявлення шаблонів та розробки точних прогнозів.

4. Інтерактивність та зворотний зв'язок: Інструмент повинен надавати інтерактивні можливості для користувачів, дозволяючи їм вносити зміни, експериментувати з різними сценаріями та аналізувати вплив різних факторів на

час виконання проєктів. Система також має забезпечувати зворотний зв'язок та візуалізацію даних для легшого розуміння результатів прогнозу.

5. Точність та надійність: Висока точність та надійність прогнозів є критично важливими. Це вимагає використання перевірених та валідованих методів, регулярного оновлення баз даних та вдосконалення алгоритмів на основі зібраних даних та досвіду виконання проектів.

6. Юзабіліті та доступність: Інструмент має бути легким у використанні для всіх учасників проекту, включаючи менеджерів, планувальників, розробників та аналітиків. Інтуїтивно зрозумілий інтерфейс, доступність інструкцій та підтримка допомагають користувачам ефективно використовувати систему для своїх потреб.

Вдале вбудовування інструменту прогнозування часу виконання проєктів в загальну систему управління проєктами та адаптація його до конкретних потреб організації можуть значно підвищити точність планування, оптимізувати розподіл ресурсів та покращити загальну ефективність виконання проєктів.

# ВИСНОВКИ

В кваліфікаційній роботі "Програмний модуль прогнозування часу виконання проєктів в компанії" було вирішено задачу розробки системи прогнозування часу розробки програмного забезпечення. У результаті були проаналізовані наступні рішення ключових проблем:

1. Оптимізація процесу збору даних: Створення автоматизованих інструментів для збору та агрегації історичних даних про проекти, що дозволило зменшити час на підготовку інформації для аналізу.

2. Розробка алгоритмів прогнозування: Впровадження статистичних та машинного навчання методів для точного прогнозування тривалості проектів на основі різних параметрів, таких як обсяг роботи, складність задач і ресурси команди.

3. Інтеграція з існуючими системами управління проектами: Розроблено інтерфейси для злиття нової системи прогнозування з існуючими інструментами управління проектами, що забезпечило безшовну інтеграцію і використання.

4. Валідація та тестування моделі: Програмний модуль був ретельно протестований на реальних проектах компанії, що дозволило підтвердити точність прогнозів і здійснити необхідні коригування моделі.

5. Розробка користувацького інтерфейсу: Створено інтуїтивно зрозумілий і зручний інтерфейс, який дозволяє менеджерам проектів та розробникам легко використовувати систему прогнозування в щоденній роботі.

6. Управління змінами та навчання користувачів: Проведено серію тренінгів та семінарів для забезпечення плавного переходу персоналу на нову систему і забезпечення ефективного використання інструменту.

Ці рішення в сукупності дозволили значно покращити точність планування та контролю за часом виконання проектів в компанії, знизити ризики невиконання проектів у встановлені терміни, та підвищити загальну продуктивність розробки.

Основні рішення, що були реалізовані для вирішення ключових проблем, охоплюють широкий спектр аспектів, від технічних до організаційних. Нижче детально описані ці рішення:

1. Створення централізованої бази даних: Було розроблено централізовану базу даних, що зберігає всю інформацію, необхідну для прогнозування часу виконання проєктів. Це включає дані про попередні проекти, їх тривалість, ресурси, ефективність команд та інші критичні метрики. Використання такої системи дозволяє збільшити точність прогнозів та поліпшити планування ресурсів.

2. Розробка модуля аналітики: Для обробки зібраних даних і визначення трендів було створено спеціалізований модуль аналітики. Він використовує статистичні методи та алгоритми машинного навчання для оцінки ймовірного часу виконання майбутніх проєктів на основі історичних даних.

3. Інтеграція з існуючими системами управління проектами: Система була інтегрована з існуючими інструментами управління проектами та ресурсами. Це дозволило автоматизувати процес збору даних та забезпечити безперервне оновлення інформації в системі прогнозування.

4. Розробка користувацького інтерфейсу: Було створено зручний та інтуїтивно зрозумілий користувацький інтерфейс, що дозволяє керівникам проєктів, планувальникам та аналітикам легко використовувати систему для отримання прогнозів та роботи з даними.

5. Реалізація функціоналу для коригування та оновлення даних: Система була обладнана засобами для коригування та оновлення даних, що дозволяє оперативно вносити зміни в інформацію про проекти, реагувати на зміни в умовах виконання робіт та підтримувати актуальність прогнозів.

6. Впровадження стратегій безпеки: Були впроваджені комплексні стратегії безпеки для захисту даних та забезпечення конфіденційності інформації. Це включає шифрування даних, управління доступом та аудит системи.

7. Поліпшення процесів вдосконалення та масштабування: Система була розроблена з можливістю легкого вдосконалення та масштабування, щоб відповідати зростаючим потребам організації та змінам у методології проєктного управління.

Реалізація цих рішень забезпечила створення надійної, ефективної та адаптивної системи для прогнозування часу виконання проєктів, що дозволяє зменшити ризики, оптимізувати ресурси та підвищити загальну продуктивність проектної діяльності в компанії.

Розроблені методи прогнозування часу виконання проєктів включають собою різноманітні підходи та техніки, які базуються на аналізі історичних даних, статистичному моделюванні, експертних оцінках та сучасних алгоритмах машинного навчання. Ось детальний опис деяких з них:

1. Історичний аналіз: Метод базується на аналізі даних про попередні проекти, включаючи час виконання, затрачені ресурси, продуктивність команди та інші фактори. Аналізуючи схожі проекти, система може прогнозувати час виконання нових проектів на основі виявлених закономірностей та тенденцій.

2. Метод експертних оцінок (Delphi, Planning Poker): Ці методи залучають експертів для оцінки тривалості проектів на основі їх досвіду та інтуїції. Експерти працюють незалежно один від одного або у вигляді групи для досягнення консенсусу щодо найбільш вірогідного часу виконання.

3. Статистичні методи (Regression, Time Series Analysis): Використовуються для аналізу та прогнозування часу виконання на основі математичних моделей та статистичних даних. Регресійний аналіз дозволяє виявити залежності між часом виконання та іншими параметрами проекту, а часові ряди допомагають виявити тренди та сезонні коливання.

4. Методи машинного навчання: Застосування алгоритмів машинного навчання, таких як нейронні мережі, випадкові ліси, градієнтний бустинг, для прогнозування часу виконання проектів. Ці методи можуть ефективно обробляти великі обсяги даних та виявляти складні нелінійні залежності.

5. Метод PERT (Program Evaluation and Review Technique): ПЕРТ - це метод оцінки часу виконання проектів, що враховує різні можливі сценарії та надає три оцінки часу: оптимістичну, найімовірнішу та песимістичну. З цих оцінок формується узагальнений прогноз.

6. Critical Path Method (CPM): Метод критичного шляху визначає послідовність завдань, що впливають на загальний час виконання проекту. Аналізуючи ці завдання, можна прогнозувати загальний час виконання проекту та ідентифікувати критичні ділянки, які потребують особливої уваги.

7. Методи Monte Carlo Simulation: Методи моделювання Монте-Карло використовуються для оцінки ризиків та невизначеності в проектах шляхом моделювання тисячі можливих сценаріїв і визначення ймовірності різних результатів, включаючи час виконання.

Кожен з цих методів має свої особливості, переваги та сфери застосування. Часто в реальних проектах використовується комбінація кількох методів для досягнення максимальної точності та надійності прогнозів. Розробка та впровадження системи прогнозування часу виконання проєктів вимагає глибокого розуміння всіх цих методів, а також специфіки виконуваних проєктів та доступних даних.

В результаті виконання роботи було створено програмне забезпечення, яке автоматизує певні процеси управління проєктами та створює комунікаційні канали між учасниками проєкту, а також розроблено базу даних, яка служить сховищем для всієї інформації проєктів і користувачів. Даний програмний продукт може використовуватись як для управління великими так і для малими проєктами програмного забезпечення. Для реалізації проєкту було вибрано мову *C#* так як ця мова є дуже зручною при розробці програм. Головною особливістю є вдале вбудовування інструменту прогнозування часу виконання проєктів, що включає такі аспекти:

1. Інтеграція з іншими системами: Інструмент прогнозування легко інтегрується з існуючими системами управління проектами, обліку часу,

ресурсним плануванням та іншими інструментами. Це забезпечує плавний обмін даними та уникнення необхідності введення даних вручну, знижуючи ризик помилок та підвищуючи ефективність процесів.

2. Гнучкість та конфігурованість: Інструмент достатньо гнучкий, щоб адаптуватися до різних методологій управління проектами та специфічних вимог конкретного проєкту. Користувачі повинні мати можливість налаштовувати параметри прогнозування, критерії оцінки та інші аспекти для досягнення найбільш точних та відповідних результатів.

3. Автоматизація та підтримка рішень: використовує автоматизацію для збору даних, обробки інформації та надання рекомендацій. Це включає використання алгоритмів машинного навчання та статистичного аналізу для виявлення шаблонів та розробки точних прогнозів.

4. Інтерактивність та зворотний зв'язок: Інструмент надає інтерактивні можливості для користувачів, дозволяючи їм вносити зміни, експериментувати з різними сценаріями та аналізувати вплив різних факторів на час виконання проєктів. Система також має забезпечувати зворотний зв'язок та візуалізацію даних для легшого розуміння результатів прогнозу.

Вдале вбудовування інструменту прогнозування часу виконання проєктів в загальну систему управління проєктами та адаптація його до конкретних потреб організації можуть значно підвищити точність планування, оптимізувати розподіл ресурсів та покращити загальну ефективність виконання проєктів.

# СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ ВИКОРИСТАНИХ ДЖЕРЕЛ


1. Бойченко С.В., Іванченко О.В. Положення про дипломні роботи (проєкти) випускників Національного авіаційного університету. – К.: НАУ, 2017. – 63 с.

2. ДСТУ 3008-95 Документація. Звіти у сфері науки і техніки. Структура і правила оформлення.

3. *R. Sangwan, M. Bass, N. Mullick, D. Paulish, J. Kazmeier Global Software Developmenr Handbook, Auerbach Publications, Taylor & Francis Group* 2017.

4. *Rational informal collaboration tools for software developmentteams.* – Постійне посилання: *http://www.relationship-economy.com/*? *p*=1336.

5. *Library for global software developmentteams.* – Постійне посилання: *http://www.ibm.com/developerworks/ru/library/redge/jan*08/*fryer_gothe/index.html*

6. *H.Huang, E. Trauth, Cultural Diversity Challenge, Idea Group Inc*. 2017.

7. *S.Jarvenpaa, D.Leidner,Communication and Trust in Global Virtual Teams,* 2015.

8. *D.Forsyth Group Dynamics, Thompson, 2016.*

9. *Anderson, J., & McCormick, R. (2005). Ten pedagogic principles for e-learning. Insight – Thematic Dossiers – Ten Pedagogic Principles for E-learning. Retrieved from https://oeb.global/oeb-insights/wp-content/uploads/2011/09/10-Principles-for-Successful-E-learning.pdf.*

10. *Cui, X., Shi, H. A*-based Pathfinding in Modern Computer Games/ X. Cui, H. Shi // IJCSNS International Journal of Computer Science and Network Security-Vol.11 No.1, January 2011.*

11. *Il'kaev R.I., Seleznev V.E., Aleshin V.V., Klishin G.S. Numerical simulation of gas pipeline networks: theory, computational implementation, and industrial applications. Moscow: KomKniga, 2005. 720 p.*

12. *Lumsdaine, A. A., Glaser, R. Teaching Machines and Programmed Learning: A Source Book / Dept. of Audio-Visual Instruction, National Education Association, 1961. – 724 p.*

13. *Mika, M., Charla, C. Simple, Cheap Pathfinding / M. Mika, C. Charla // AI Game Programming Wisdom-2002.*

14. *O'Neill, J. C. Efficient Navigation Mesh Implementation / J. C. O'Neill // Journal of Game Development-Vol. 1 No. 1, 2004.-71-90 pp.*

15. *Operator Training Simulation Global Market Research Study. Market Analysis and Forecast through 2017. ARC Advisory Group. 2012*

16. *Ostapenko V.O. 3D visualization in learning systems / Artamonov Y.B., Ostapenko V.O. // Матеріали XIV міжнар. наук.-техн. конф. "Авіа-2019" (23-24 квітня 2019). – К.: НАУ, 2019. – електронний збірник. Постійне посилання: http://conference.nau.edu.ua/index.php/AVIA/AVIA2019/paper/view/6230/4724.*

17. *Yap, P. Grid-Based Path-Finding / P.Yap // AI '02 Proceedings of the 15th Conference of the Canadian Society for Computational Studies of Intelligence on Advances in Artificial Intelligence-2002. – 44-55 pp*

18. *Zhang, G. Precise algorithm to generate random sequential addition of hard hyperspheres at saturation / G. Zhang, S. Torquato // Physical review, E 88. 2013. pp.053312-1-9.*

19. *Alan F. Broady. Tools that can help bridge geographical and cultural gaps*. – Постійне посилання: *http*://*www.dev.com*/*d*/ *rational*/*library*.

20. *Developer works rational informal collaboration tools for global software development teams.* – Постійне посилання: *http*://*www.ibm.com*/*developerworks*/ *rational*/*library*/09/*informalcollaborationtoolsforglobalsoftwaredevelopmentteams*

21. *New developer works for global software development teams.* – Постійне посилання: *www.MasterNewMedia.org*

22. *Developer works rational informal collaboration tools for global software developmentteams.* – Постійне посилання: *http*://*www.nytimes.com*/2018/09/07/ *magazine*/07*awareness-t.htm*

23. *Turner, Michael S. V.* (2023-08-30). *Microsoft Solutions Framework Essentials*: *Building Successful Technology Solutions*. *Microsoft Press*.

24. *R. Sangwan, M. Bass, N. Mullick, D. J. Paulish and J. Kazmeier The Foundation of Global Development Management*. – 2015.

25. *Diomidis Spinellis Global Software Development in the FreeBSD Project*

26. *Brooks, Jr., F.P. The Mythical Man-Month*: *Essays on Software Engineering,* 20*th Anniversary Edition*. *Reading, MA*: *Addison-Wesley,* 1995*,* 322 *pages*.

27. Савченко Д.С. Програмний модуль прогнозування часу виконання проєктів в *IT*-компанії // Тези доповідей наук.-практ. конф. “Сучасні тенденції розвитку системного програмування” (23-24 листопада 2023 р.). – К.: НАУ*,* 2023. – С. 69.

Додаток А

Схема алгоритму роботи з програмою керівника проєкту

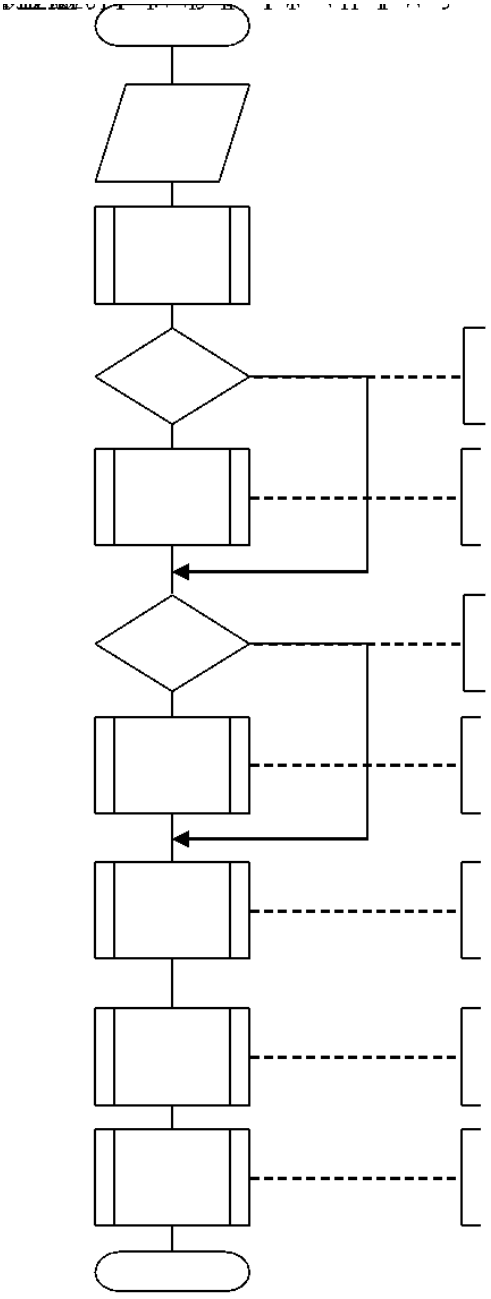

Рис. А.1. Схема алгоритму роботи з програмою керівника проєкту