

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Таврійський національний університет
імені В.І. Вернадського
Навчально-науковий інститут муніципального управління
та міського господарства
Кафедра інженерних систем та технологій

На правах рукопису

Кваліфікаційна робота на здобуття ступеня вищої освіти «бакалавр»

ОПТИМІЗАЦІЯ ПРОЦЕСІВ УПРАВЛІННЯ ПРОГРАМНИМИ
ПРОЄКТАМИ НА ОСНОВІ ВПРОВАДЖЕННЯ МЕТОДІВ
АВТОМАТИЗАЦІЇ

Здобувача вищої освіти
Зубковського Андрія Олексійовича
Освітня програма
«Автоматизація та комп'ютерно-
інтегровані технології»
(Спеціальність
174 «Автоматизація, комп'ютерно-
інтегровані технології та
робототехніка»)

(підпис)

Науковий керівник:
ст.викл. Мінаєва Ю.Ю.

(підпис)

Національна шкала _____
Кількість балів _____
Оцінка: ECTS _____

Таврійський національний університет імені В. І. Вернадського
Навчально-науковий інститут муніципального управління
та міського господарства
Кафедра інженерних систем та технологій
Освітньо-кваліфікаційний рівень бакалавр
Спеціальність 174 «Автоматизація, комп'ютерно-інтегровані технології та
робототехніка»

«ЗАТВЕРДЖУЮ»
Завідувач кафедри
_____ Наталія ОМЕЦІНСЬКА
«___» _____ 2025 року

ЗАВДАННЯ НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ

Зубковського Андрія Олексійовича

1. Тема роботи: Оптимізація процесів управління програмними проєктами на основі впровадження методів автоматизації

керівник роботи: ст.викл. Мінаєва Ю.Ю.

затверджені Наказом ТНУ імені В.І Вернадського:
від «7» листопада 2025 № 138/1-ОД

2. Строк подання студентом роботи: 25.11.2025

3. Вихідні дані до роботи: масиви статистичних даних про результати тестування програмних продуктів; Загальна кількість програмних модулів, виділених у структурі системи; Сумарна кількість змінних у програмному коді; Кількісні показники зв'язності та зачеплення модулів для вибору стратегії тестування.

4. Зміст розрахунково-пояснювальної записки: Проаналізувати сучасні тенденції автоматизації розробки програмного забезпечення; Обґрунтувати доцільність використання системи MS Project для оптимізації процесів управління проєктами; Запропонувати комплекс автоматизованих рішень для бізнес-процесів малого підприємства; Пропрацювати комплексні заходи з охорони праці, розрахувати параметри освітлення і кондиціонування та визначити алгоритми дій персоналу в надзвичайних ситуаціях.

5. Перелік графічного матеріалу: графічна робота виконана у вигляді мультимедійної презентації

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Розділ 1	старший викладач Юлія МІНАЄВА		
Розділ 2	старший викладач Юлія МІНАЄВА		
Розділ 3	старший викладач Юлія МІНАЄВА		
Розділ 4	старший викладач Юлія МІНАЄВА		

7. Дата видачі завдання _____ 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Термін виконання етапів роботи	Заключний документ етапу
1	Вибір теми бакалаврської роботи, призначення керівника	до 31.10.2025	Затвердження переліку тем бакалаврських робіт
2	Пошук і відбір літератури по темі роботи, складання плану бакалаврської роботи	до 14.12.2025	Список літературних джерел, план роботи
3	Визначення об'єкта, предмета, мети, завдань та методів дослідження, написання вступу до теми бакалаврського дослідження	до 25.01.2026	Текст вступу
4	Написання тексту бакалаврської роботи відповідно до її структури:	26.01.2026-17.05.2026	Текст розділів
	4.1 I розділ	26.01.2026-6.03.2026	Текст розділу
	4.2 II розділ	9.03.2026-27.03.2026	Текст розділу
	4.3 III розділ	30.03.2026-17.04.2026	Текст розділу
	4.4 IV розділ	20.04.2026-8.05.2026	
5	Підготовка графічних матеріалів чи іншого унаочнення	11.05.2026-15.05.2026	Роздатковий матеріал, презентація
6	Оформлення кінцевого списку використаних джерел та додатків	11.05.2026-15.05.2026	Список літературних джерел
7	Оформлення та попередній захист бакалаврської роботи	18.05.2026-22.05.2026	Бакалаврська робота, рішення кафедри про допуск до захисту
8	Внесення коректив та кінцеве редагування бакалаврської роботи	25.05.2026-29.05.2026	Бакалаврська робота
9	Реєстрація бакалаврської роботи на кафедрі	1.06.2026-5.06.2026	Бакалаврська робота внесена до журналу реєстрації випускових робіт
10	Захист бакалаврської роботи	10.06.2026-28.06.2026	Рішення Екзаменаційної комісії про захист

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ.....	07
ВСТУП	08
РОЗДІЛ 1 АВТОМАТИЗАЦІЯ ПРОЦЕСІВ СТВОРЕННЯ ТА УПРАВЛІННЯ ПРОГРАМНИМИ ПРОЄКТАМИ	10
1.1 Методи і засоби автоматизації	10
1.2 Автоматизація процесу розробки	12
1.3 Розвиток мов програмування	17
1.4 Сервісна модель процесу розробки програмного забезпечення	19
Висновки до розділу 1	22
РОЗДІЛ 2 АВТОМАТИЗОВАНІ СИСТЕМИ ТА МЕТОДИ УПРАВЛІННЯ ПРОЄКТАМИ НА ПІДПРИЄМСТВАХ	24
2.1 Загальні відомості про управління проєктами	24
2.2 Що розуміється під проєктом	26
2.3 Коротка історія питання	28
2.4 Постановка задачі управління проєктами	33
2.5 Сучасні стандарти управління проєктами	37
2.6 Інформаційні системи	39
2.7 Використання систем тривимірного моделювання	42
2.8 Програмне забезпечення MS Project	44
Висновки до розділу 2	47
РОЗДІЛ 3 ПРАКТИЧНА МЕТОДОЛОГІЯ ВИКОРИСТАННЯ MS PROJECT	49
Висновки до розділу 3	58
РОЗДІЛ 4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ	61
4.1 Розрахунок штучного освітлення у приміщенні	61
4.2 Первинні засоби пожежегасіння	64
4.3 Кондиціонування повітря в лабораторії	67
4.4 Безпека в надзвичайних ситуаціях	70
Висновки по розділу 4	71
ЗАГАЛЬНИЙ ВИСНОВОК	73
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	76

АНОТАЦІЯ

Зубковський А. О. Оптимізація процесів управління програмними проєктами на основі впровадження методів автоматизації. – Рукопис.

Кваліфікаційна бакалаврська робота за спеціальністю 174 «Автоматизація, комп'ютерно-інтегровані технології та робототехніка». – Таврійський національний університет імені В.І. Вернадського, Київ, 2026 рік.

У бакалаврській кваліфікаційній роботі досліджено теоретичні засади технічної діагностики та автоматизованого оцінювання надійності програмних систем. Розроблено програмний засіб попереднього опрацювання даних про відмови програмних систем, інтегрований із системою JIRA. Запропоновано комплекс рішень для автоматизації діяльності малого підприємства на прикладі фірми Sigma, що включає розробку спеціалізованої програми для SPACE-аналізу, бази даних «Менеджер» у середовищі MS Access та інтерактивного веб-сервісу діагностики гаджетів на базі хмарних технологій.

Практичне значення полягає у можливості застосування методу нелінійного регресійного аналізу для підвищення точності прогнозування кількості помилок (на 3–5%) та скорочення тривалості тестування (в середньому на 30-40%), а також у впровадженні розроблених програмних інструментів для оптимізації стратегічних, комерційних і сервісних процесів підприємства без додаткових значних капіталовкладень.

Об'єкт дослідження – процес оцінювання, прогнозування та забезпечення показників надійності програмних систем.

Предмет дослідження – математичні моделі, методи, алгоритми та інструментальні програмні засоби для тестування, технічної діагностики та автоматизації бізнес-процесів.

Мета роботи – розробка та впровадження автоматизованих методів і програмних засобів для оцінювання надійності програмних систем, а також комплексної автоматизації діяльності структурних підрозділів малого підприємства.

Методи дослідження – системний аналіз, нелінійний регресійний аналіз, математичне та структурне моделювання, SPACE-аналіз, методи проектування реляційних баз даних.

Ключові слова: надійність програмних систем, автоматизація тестування, технічна діагностика, математичне моделювання, бази даних, автоматизація малого бізнесу, веб-сервіс, SPACE-аналіз.

ABSTRACT

Zubkovskiy A. O. Optimization of software project management processes based on the implementation of automation methods. – Manuscript.

Bachelor's qualification work in specialty 174 "Automation, Computer-Integrated Technologies, and Robotics". – Taurida National V.I. Vernadsky University, Kyiv, 2026.

The bachelor's qualification work investigates the theoretical foundations of technical diagnostics and automated reliability assessment of software systems. A software tool for pre-processing data on software system failures, integrated with the JIRA system, has been developed. A set of solutions for automating the activities of a small enterprise is proposed on the example of the "Sigma" company, which includes the development of a specialized program for SPACE analysis, a "Manager" database in the MS Access environment, and an interactive web service for diagnosing gadgets based on cloud technologies.

The practical significance lies in the possibility of applying the non-linear regression analysis method to increase the accuracy of predicting the number of errors (by 3–5%) and reduce the duration of testing (by an average of 30-40%), as well as in the implementation of the developed software tools to optimize the strategic, commercial, and service processes of the enterprise without significant additional capital investment.

Object of study – the process of assessment, forecasting, and ensuring reliability indicators of software systems.

Subject of study – mathematical models, methods, algorithms, and instrumental software tools for testing, technical diagnostics, and automation of business processes.

Purpose of the work – development and implementation of automated methods and software tools for assessing the reliability of software systems, as well as comprehensive automation of the activities of structural units of a small enterprise.

Research methods – system analysis, non-linear regression analysis, mathematical and structural modeling, SPACE analysis, methods of designing relational databases.

Keywords: software system reliability, test automation, technical diagnostics, mathematical modeling, databases, small business automation, web service, SPACE analysis.

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

ІТ- інформаційні технології

ЕОМ (ПЕОМ) - персональна електронно-обчислювальна машина

ПЗ - програмне забезпечення

ПМ - мова програмування

ООП - мова об'єктно-орієнтованого програмування

ПД - програмна документація

СУБД - система управління базами даних

САПР - система автоматизованого проектування

ВСТУП

У представленій бакалаврській кваліфікаційній роботі здійснюється глибоке дослідження сучасних тенденцій, спрямованих на мінімізацію та усунення рутинних процесів на всіх етапах життєвого циклу створення програмного забезпечення (ПЗ). Зокрема, детально розглядаються процеси автоматизації під час проектування архітектури, безпосереднього написання програмного коду, проведення комплексного тестування, розгортання готових рішень та формування супровідної технічної документації. Крім того, у дослідженні ґрунтовно проаналізовано практичне застосування цих інноваційних трендів шляхом впровадження спеціалізованого інструментарію, який дозволяє ефективно автоматизувати як окремі етапи, так і цілі комплекси завдань у сфері програмної інженерії.

Особлива увага в тексті приділяється наочним прикладам еволюції та трансформації традиційних парадигм створення програмних продуктів. Цей перехід від класичних підходів до новітніх методологій розглядається крізь призму актуальних запитів суспільства та стрімкого розширення технологічних можливостей сучасного ринку інформаційних технологій (ІТ). Додатково в роботі висвітлюються перспективні напрямки та ймовірні сценарії якісної еволюції ролей як кінцевих користувачів цифрових продуктів, так і безпосередньо фахівців-професіоналів, що працюють у високотехнологічній ІТ-галузі.

В якості головного об'єкта даного дослідження бакалаврського рівня виступають сучасні автоматизовані інформаційні системи, призначені для супроводу та підтримки процесів розробки програмного забезпечення. Виходячи з цього, основна мета кваліфікаційної роботи полягає у всебічному аналізі наявних рішень та безпосередньому проектуванні ефективних шаблонів управління проектами. Ці шаблони розробляються спеціально для інтеграції у найбільш інтуїтивно зрозуміле та загальнодоступне інформаційне середовище, призначене для моніторингу та контролю за процесом виконання завдань зі створення нових програмних продуктів.

Для досягнення поставленої мети було сформовано низку ключових дослідницьких завдань, які послідовно вирішуються в роботі. Перш за все, передбачається проведення детального критичного аналізу передових технологій конструювання програмного забезпечення, а також вивчення існуючих цифрових середовищ, що застосовуються для управління проектами. Наступним важливим кроком є обґрунтований вибір оптимальної системи, що дозволяє автоматизувати організаційні процеси в команді розробників. На завершальному етапі дослідження виконується практична реалізація завдання зі створення комплексної структури спеціалізованих шаблонів управління проектами, які будуть застосовуватися безпосередньо під час розробки ПЗ.

РОЗДІЛ 1 АВТОМАТИЗАЦІЯ ПРОЦЕСІВ СТВОРЕННЯ ТА УПРАВЛІННЯ ПРОГРАМНИМИ ПРОЄКТАМИ

1.1 Методи і засоби автоматизації

Розглядаючи функціональні етапи процесу створення програмного забезпечення, варто зазначити, що вони знаходять своє втілення через різноманітні архітектурні моделі та спеціалізовані методології. У сучасній інженерній практиці традиційно виокремлюють класичну послідовність кроків, необхідних для повноцінної реалізації проєктів такого типу. Цей комплексний процес розпочинається з глибокого аналізу предметної області та детальної формалізації існуючих бізнес-процесів. Далі настає черга збору, аналізу та формування чітких вимог до майбутнього програмного продукту, після чого фахівці переходять до проєктування архітектури системи та стратегічного планування її подальшої реалізації. Наступними критично важливими стадіями є безпосереднє написання програмного коду, а також проведення ретельного тестування і налагодження створених модулів. Невід'ємною складовою успішного проєкту є підготовка супровідної програмної документації, яка передуює етапам офіційної сертифікації розробленого продукту (як самого коду, так і документації до нього). Завершальними, але не менш значущими фазами життєвого циклу є безпосереднє впровадження готового рішення в експлуатацію та його подальший технічний супровід.

Історично склалося так, що послідовне проходження зазначених вище фаз сформувало базис для так званої каскадної, або водоспадної, моделі конструювання програмного забезпечення, концепцію якої вперше детально охарактеризував Вінстон Ройс у 1970 році. Проте з часом ця парадигма зіткнулася зі значною критикою через свою недостатню гнучкість та надмірну тривалість виробничого циклу, оскільки вона унеможливлювала старт наступного етапу до повного завершення попереднього. Усвідомлюючи ці недоліки, сам автор згодом запропонував вдосконалену модифікацію цього підходу. Оновлена концепція трансформувалася в модель ітеративної розробки, головною перевагою якої стало розпаралелювання робочих процесів, що

супроводжувалося постійним моніторингом проміжних результатів та можливістю оперативного внесення коректив у вже пройдені етапи.

Логічним продовженням еволюції ітеративного підходу стала спіральна модель, розроблена Баррі Боемом у 1986 році. Ця архітектура передбачає безперервне здійснення аналізу потенційних ризиків, паралельне проектування та поетапне створення прототипів протягом усього життєвого циклу проекту, гнучко реагуючи на початкові та нові вимоги, що постійно виникають у процесі роботи. Схожа філософія безперервного вдосконалення та поглибленої деталізації проектних рішень також стала фундаментом для V-подібної моделі, яка набула широкого поширення в дев'яностих роках минулого століття.

Якщо розглядати практичну площину застосування принципів спіральної та V-подібної моделей, зокрема в контексті безпосереднього управління діяльністю команд, то вони знайшли своє яскраве відображення у сучасних гнучких методологіях. До цієї категорії належать такі популярні та ефективні підходи родини Agile, як розробка через тестування (Test-Driven Development), екстремальне програмування (Extreme Programming), а також фреймворки SCRUM, Lean, DSDM та низка інших споріднених практик.

Стрімке зростання популярності гнучких методологій насамперед продиктоване жорсткими умовами сучасного ринку, який вимагає від компаній максимального скорочення термінів випуску готової продукції при одночасному підвищенні її надійності та якості. Саме тому в реальному секторі економіки сьогодні домінують моделі поетапного створення програмних продуктів із застосуванням інструментів безперервного прототипування. Цікавим наслідком такої трансформації є поступове розмиття чітких меж між професійними ролями в IT-командах, де фахівці все частіше виконують крос-функціональні завдання, поєднуючи аналіз вимог, написання коду, проведення тестування та інші обов'язки.

Однак, окрім суто методологічних підходів та питань управління організаційними процесами, критично важливим критерієм успіху є наявність та раціональне використання ергономічного робочого інструментарію. У сфері програмної інженерії до таких засобів належать сучасні мови програмування,

інтегровані середовища розробки, а також різноманітні допоміжні методики та спеціалізовані інструменти автоматизації рутинних операцій. Варто підкреслити, що еволюція цього технічного забезпечення стимулюється не лише поточними бізнес-потребами компаній, але й очікуваннями глобальних якісних трансформацій у всій галузі інформаційних технологій. Зважаючи на це, в рамках даної роботи ставиться амбітне завдання не лише провести глибокий аналітичний огляд поточного стану автоматизації у сфері розробки програмного забезпечення, але й здійснити науково обґрунтоване прогнозування майбутніх векторів розвитку ІТ-індустрії, базуючись на виявлених закономірностях.

1.2 Автоматизація процесу розробки

На сучасному етапі розвитку програмної інженерії чітко простежується тенденція до невинного підвищення рівня абстракції як під час безпосереднього програмування, так і в процесі системного проектування. Сьогодні спостерігається поступове розмиття жорстких кордонів між класичними етапами життєвого циклу розробки. Як приклад, сучасні архітектурні конструкції, реалізовані у вигляді паттернів, вже не є суто теоретичними моделями — вони інкапсулюють у собі конкретні механізми реалізації. Це включає автоматичну генерацію програмного коду, створення моделей даних та інтерфейсів їх представлення, а також автоматизоване формування наборів для модульного тестування і генерацію відповідної програмної документації.

Якщо розглядати методичний зріз, то класичні об'єктно-орієнтовані шаблони проектування (такі як «Фасад», «Спостерігач», «Компонувальник» та багато інших) еволюціонували у комплексні архітектурні схеми реалізації додатків. До найвідоміших серед них належать патерни Model-View-Controller (MVC), Model-View-Presenter (MVP) та Model-View-ViewModel (MVVM). Кожна з цих архітектурних парадигм представляє собою складну систему, що складається з низки взаємодіючих компонентів, логічно розділених за функціональним призначенням. Наприклад, типовим є суворе розмежування

бізнес-логіки додатка, моделі зберігання даних та модулів, що відповідають за інтерфейсне відображення для користувача [6].

Саме через впровадження таких архітектурних рішень на практиці реалізуються фундаментальні SOLID-принципи об'єктно-орієнтованого дизайну, які свого часу були систематизовані Робертом Мартіном [7]. Ці принципи включають: Single Responsibility (принцип єдиної відповідальності, згідно з яким клас або компонент повинен вирішувати лише одне завдання); Open-Closed (програмні сутності мають бути відкритими для подальшого розширення свого функціоналу, але закритими для модифікації існуючого коду); Liskov Substitution (принцип підстановки Барбари Лісков, який стверджує, що функції, які використовують посилання на базові класи, повинні мати можливість коректно працювати з об'єктами будь-яких похідних від них класів без зміни своєї поведінки); Interface Segregation (принцип розділення інтерфейсів, що запобігає створенню "товстих" інтерфейсів); та Dependency Inversion (принцип інверсії залежностей, який декларує, що модулі верхніх рівнів не повинні залежати від модулів нижніх рівнів, натомість обидва типи модулів мають залежати від абстракцій). Зазначені базові принципи також повністю корелюють з ідеєю повторного використання незалежних програмних модулів, що є основою компонентно-орієнтованого підходу до розробки [8]. Таким чином, передові практики проектування та кодування проходять процес уніфікації, перетворюючись на своєрідні «будівельні блоки», які стандартизовано застосовуються при створенні нових програмних продуктів, подібно до того, як стандартна цегла використовується при зведенні нової будівлі.

На практичному, прикладному рівні описані вище архітектурні концепції отримують потужну підтримку з боку спеціалізованого інструментарію розробки програмного забезпечення. Сучасні засоби переважно реалізовані у форматі модульних програмних каркасів — так званих фреймворків. Ці комплексні рішення об'єднують велику кількість розрізнених компонентів для вирішення широкого спектру прикладних задач. Крім того, розширення базового функціоналу таких інтегрованих середовищ розробки легко

здійснюється шляхом підключення зовнішніх бібліотек та додаткових пакетів. Найбільш репрезентативними прикладами подібних потужних інструментальних комплексів на сьогодні є популярні фреймворки, призначені для створення веб-додатків, серед яких варто виділити Ruby on Rails, Pylons, Phalcon, Django, Zend Framework, Symfony, Yii та низку інших аналогічних рішень [9].

Застосування цих програмних каркасів дозволяє суттєво інтенсифікувати процес розробки завдяки механізмам автоматизованого генерування структури бази даних на основі описаних моделей, а також швидкому створенню компонентів інтерфейсу користувача та базового функціоналу. Ще однією вагомою перевагою сучасних мов програмування та середовищ розробки є глибока інтеграція інструментів для автоматизованого тестування ПЗ. Яскравими прикладами можуть слугувати вбудовані модулі doctest та unittest у стандартній бібліотеці мови Python, кросплатформний фреймворк TextTest, призначений для тестування графічних інтерфейсів користувача (GUI), система Selenium WebDriver для автоматизації тестування веб-інтерфейсів у браузерях, а також безліч інших спеціалізованих інструментів.

Як логічний наслідок автоматизації, на фінальному етапі генерується стандартизована програмна документація в уніфікованих форматах (таких як HTML, HTMLHelp, PDF, RTF тощо), яка може бути орієнтована як на вузькопрофільних технічних спеціалістів, так і на кінцевих користувачів готового продукту. Через наявність суттєвих синтаксичних відмінностей між різними мовами програмування та специфічні особливості прикладних середовищ розробки, сьогодні практично для кожного популярного фреймворку створено відразу кілька сумісних генераторів документації. Наприклад, потужна кросплатформна система документування вихідних кодів Doxygen забезпечує підтримку таких мов, як C++, C, Objective-C, Python, Java, IDL та PHP. Генератор документації та пошуковий рушій Sphinx, початково створений для мови Python, має глибоку інтеграцію з веб-фреймворками Django та Pylons. Комерційний продукт VSdocman, призначений для генерування документації з вихідних кодів на мовах VisualBasic.NET та C#, реалізований у

формі зручного доповнення безпосередньо до інтегрованого середовища розробки Microsoft Visual Studio. Не можна оминати увагою і спеціалізований інструментарій дескрипторів Javadoc, який еволюціонував у стандарт де-факто для документування класів, інтерфейсів, полів, конструкторів, методів та цілих пакетів у програмних продуктах, написаних на мовах програмування Java-подібного сімейства.

Значне скорочення загальних часових витрат на розробку досягається не лише завдяки автоматизації окремих розрізнених етапів, але й за рахунок впровадження єдиного (наскрізного) інструментарію розробки. Це позбавляє команди необхідності багаторазово та ітеративно виконувати супутні процеси при кожному внесенні змін до коду. Такий рівень оптимізації стає можливим завдяки наявності єдиного центрального джерела істини (моделі) та здатності системи автоматично коригувати і оновлювати всі пов'язані з нею елементи. Ця передова концепція наскрізної інтеграції даних, які генеруються та використовуються на абсолютно різних фазах життєвого циклу розробки ПЗ, є фундаментальною і для сучасних веб-систем управління знаннями та ІТ-проектами. Зазвичай, архітектура таких баз знань спирається на вікі-рушії і має форму веб-сайту, структуру та інформаційне наповнення якого авторизовані користувачі здатні самотійно і оперативно модифікувати за допомогою набору вбудованих інструментів редагування. Серед найвідоміших та найпоширеніших систем у цій специфічній категорії варто відзначити DokuWiki, XWiki та ThoughtFarmer.

З метою забезпечення комплексної автоматизації всіх аспектів створення ПЗ, корпоративні бази знань глибоко інтегруються з системами відстеження помилок (баг-трекерами), системами контролю версій вихідного коду та спеціалізованими інструментами управління ІТ-проектами. Класичними прикладами такої синергії є відкритий веб-додаток Redmine, а також платформи Kerio Samerage чи XWiki, які успішно поєднують у собі функціонал одразу кількох напрямків. Особливо популярною є потужна зв'язка комерційних продуктів від компанії Atlassian: системи управління знаннями Confluence та баг-трекера JIRA. Існують також інші подібні комплексні рішення, які

дозволяють додавати прямі посилання на конкретні ділянки програмного коду, розміщені на спеціалізованих сервісах веб-хостингу ІТ-проектів, таких як Bitbucket та GitHub [11].

Переходячи до розгляду етапу розгортання (deployment) новоствореного програмного забезпечення, який логічно слідує одразу за фазами розробки та тестування, слід зазначити, що найбільший інтерес в сучасних реаліях становить модель PaaS (Platform as a Service — Платформа як послуга) у сфері хмарних обчислень. У межах цієї моделі хмарний провайдер надає розробникам повністю готову програмну платформу, заздалегідь оснащену всіма необхідними компонентами для стабільного функціонування програмного забезпечення, що розгортається. Цей комплекс зазвичай включає операційну систему, систему управління базами даних (СУБД), налаштовані середовища виконання для різних мов програмування та інші супутні сервіси, формуючи ідеальне середовище для швидкого розміщення нових додатків. Для забезпечення ізоляції та масштабованості на цьому етапі активно застосовуються передові технології віртуалізації додатків, контейнеризації та складні системи оркестрації мікросервісів.

Наступна фаза — етап безпосереднього впровадження готового програмного забезпечення в операційну діяльність компанії-замовника — зазвичай є комплексним організаційно-методичним проектом. Він охоплює не лише технічні аспекти, але й навчання майбутніх користувачів, а в деяких випадках вимагає радикальної зміни звичного алгоритму виконання ними своїх повсякденних робочих завдань. Втім, навіть цей складний процес піддається частковій автоматизації шляхом використання типових процедур та методик впровадження, які є специфічними для кожного конкретного програмного продукту. Наприклад, провідні компанії-розробники корпоративного ПЗ, такі як SAP, 1С, Директум та багато інших, пропонують своїм клієнтам готові шаблони та найкращі світові практики (best practices) для максимально ефективного впровадження власних продуктів [13]. Серед стандартизованих рішень у цій сфері варто виділити методологію «Rapid Concept Development & Implementation» (RCDI). Вона характеризується високою сумісністю з

ключовими ідеологіями в галузі управління проектами та управління якістю (такими як PMI PMBOK та TQM), а також відповідає суворим вимогам міжнародних стандартів серії ISO 9001:2000 (в Україні адаптовано як ДСТУ ISO 9001:2001) та стандарту соціальної відповідальності SA 8000:1997 [14].

Схожі принципи організації застосовуються і до етапу післягарантійного та технічного супроводу програмного забезпечення, процеси якого також частково автоматизовані завдяки впровадженню спеціалізованих методичних та програмних засобів. Відповідно до концепції процесного підходу, бібліотека інфраструктури інформаційних технологій (ITIL) розглядає супровід ПЗ як комплексну сукупність тісно взаємопов'язаних сервісів. Ці сервіси охоплюють управління проблемами, інцидентами, конфігураціями, релізами, змінами тощо, а також безпосередню підтримку кінцевих користувачів (Help Desk / Service Desk). Згідно з рекомендаціями ITIL, система технічної підтримки користувачів має бути організована за принципом суворої централізації: всі звернення кінцевих користувачів збираються в єдиній точці контакту (Single Point of Contact), після чого диспетчеризуються та розподіляються за відповідними рівнями (лініями) підтримки, залежно від складності та специфіки виявленої проблеми [15].

1.3 Розвиток мов програмування

В межах безпосереднього етапу створення програмного забезпечення, а саме під час написання коду, критично важливим постає питання обґрунтованого вибору мови програмування (МП). Цей вибір має стратегічне значення, оскільки саме він визначає можливість ефективного застосування різноманітних засобів автоматизації робочого процесу, таких як архітектурні шаблони, готові фреймворки та спеціалізовані бібліотеки. Історично склалося так, що традиційна класифікація мов програмування базується переважно на їхній приналежності до певної фундаментальної обчислювальної моделі або специфічної парадигми. Проте сучасна інженерна практика диктує інші правила: визначальним критерієм під час вибору парадигми та відповідного інструменту для її реалізації стає безпосередній спектр завдань, що підлягають

вирішенню, а також специфіка предметної області майбутнього програмного продукту. Яскравим прикладом цього є розробка високонавантажених розподілених додатків. Для таких складних систем найбільш доцільним є використання мов, архітектура яких містить вбудовані засоби генерування легковагових паралельних процесів та забезпечення їхньої безпечної взаємодії шляхом обміну асинхронними повідомленнями, що цілком відповідає відомій моделі акторів [17]. На сьогоднішній день найяскравішими та найбільш затребуваними представниками цього напрямку справедливо вважаються такі мови програмування, як Erlang та Go.

Спираючись на результати глибокого аналізу сучасних тенденцій у сфері появи нових та еволюції вже існуючих мов програмування, можна констатувати впевнений рух індустрії в напрямку мультипарадигмальності. Цей підхід передбачає гнучке та органічне поєднання в межах одного інструменту сильних сторін як імперативного, так і декларативного стилів написання коду. Зокрема, розробники отримують комплексний інструментарій, що об'єднує можливості об'єктно-орієнтованого, класичного процедурного, структурного, а також потужного функціонального та логічного програмування. До яскравих прикладів таких відносно молодих, але вже надзвичайно популярних та визнаних світовою спільнотою розробників мов, можна віднести Scala (представлена у 2003 році), F# (2005 рік), раніше згадану Go (2009 рік), системну мову Rust (2010 рік), створену для екосистеми Apple мову Swift (2014 рік) та низку інших сучасних рішень.

Іншою, не менш яскраво вираженою тенденцією розвитку ринку мов програмування протягом останніх років, є стрімке зростання популярності декларативної парадигми. Сучасні реалії такі, що дедалі більша кількість розроблюваних інструментів намагається повноцінно підтримувати функціональні та логічні підходи до проектування логіки. Наочною ілюстрацією ефективності та зручності декларативного стилю є широко відомі мови запитів та розмітки даних, серед яких XSLT, SQL, каскадні таблиці стилів CSS, мова гіпертекстової розмітки HTML тощо. Головною рушійною силою такої масової популяризації декларативних мов є постійне та неминуче

підвищення загального рівня складності програмних систем, що змушує інженерів переходити до вже згаданої концепції програмування «за принципом конструктора Lego». Фундаментальна перевага декларативних мов полягає в тому, що для вирішення конкретного завдання програмісту більше не потрібно алгоритмічно, крок за кроком, розписувати послідовність дій; замість цього достатньо лише чітко формалізувати та описати бажаний кінцевий результат через систему певних правил та обмежень.

Більше того, сама семантика декларативних мов програмування закладає базовий принцип незалежності смислового навантаження конкретного використовуваного оператора від його фізичного розташування у загальній структурі програмного коду. Саме ця унікальна властивість робить програми, написані в декларативному стилі, значно простішими для сприйняття, аналізу та подальшої підтримки порівняно з їхніми імперативними аналогами. У підсумку варто зазначити, що стрімка популяризація та впровадження окремих передових тенденцій у розвитку мов програмування знаходить свою потужну та безперервну підтримку через використання відповідних методів та засобів автоматизації процесу розробки ПЗ, про які детально йшлося вище: сучасних гнучких методологій, оптимізованих архітектурних моделей, перевірених часом найкращих практик (best practices), а також стандартизованих шаблонів та фреймворків.

1.4 Сервісна модель процесу розробки програмного забезпечення

Спираючись на попередньо проаналізовані вектори технічної та організаційної еволюції у сфері програмної інженерії, є всі підстави прогнозувати ще інтенсивніше масштабування тенденцій до комплексної автоматизації всіх без винятку виробничих процесів у цій високотехнологічній галузі в осяжній перспективі. Зважаючи на тотальне проникнення інформаційних технологій в абсолютно всі аспекти життєдіяльності сучасного суспільства, фундаментальна теза про те, що базові навички програмування є життєво необхідною компетенцією для сучасної людини, набуває безпрецедентної актуальності. Варто нагадати, що цей концепт був уперше

публічно артикульований ще на початку вісімдесятих років минулого століття видатним науковцем А. П. Єршовим під час проведення Третьої всесвітньої конференції під егідою ІФП та ЮНЕСКО, яка була присвячена питанням впровадження електронно-обчислювальних машин в освітні процеси [18]. Сьогодні ми є свідками того, як безліч повсякденних явищ, специфічна термінологія та інноваційні бізнес-стратегії — такі як віртуальні комунікації через соціальні платформи, сервісно-орієнтовані моделі надання різноманітних послуг та гнучкі методи управління проектами — органічно інтегрувалися в реальний сектор економіки та повсякденний побут саме з передової ІТ-індустрії. Разом з тим, необхідно чітко усвідомлювати, що аматорський підхід до програмування та професійне володіння цим складним інженерним ремеслом (яке багато хто справедливо прирівнює до мистецтва [19]) мають кардинальні відмінності. Головною демаркаційною лінією тут виступає наявність глибоких, систематизованих знань щодо теоретичного базису та фундаментальних принципів функціонування обчислювальних систем. Незважаючи на це, глобальна діджиталізація та повсюдне впровадження спеціалізованого ІТ-інструментарію в усі без винятку сфери економічної діяльності (починаючи від платформ електронної комерції і закінчуючи складними обліково-керуючими інформаційними системами) закономірно стимулюють широку популяризацію базових цифрових навичок та їх активне засвоєння фахівцями суміжних і навіть гуманітарних професій.

Як наслідок цих трансформацій, на сучасному етапі розвитку технологій потенційний замовник або кінцевий споживач цифрового продукту володіє достатнім інструментарієм, щоб не просто вербально висловити свої очікування, але й самостійно здійснити їх сувору формалізацію. Для цього активно застосовуються загальноприйняті стандарти уніфікованого графічного моделювання, зокрема діаграми сімейств IDEF, UML, BPMN, EPC та інші споріднені нотації. Якщо екстраполювати цю тенденцію на найближче майбутнє, особливо в контексті тотального домінування хмарної концепції SaaS (програмне забезпечення як послуга), цілком ймовірним виглядає сценарій масової появи інтелектуальних платформ, здатних автоматично генерувати

повноцінний програмний код виключно на основі формалізованих вимог особи, яка одночасно виступає і замовником, і майбутнім користувачем системи. Варто підкреслити, що зародки таких сервісно-орієнтованих рішень для автоматизованої генерації типових веб-проектів успішно функціонують вже сьогодні: яскравими прикладами є популярні онлайн-конструктори на кшталт wix.com, weebly.com та подібні до них системи компонування контенту. Паралельно з цим, у руслі глобального тренду на мобільність, активно еволюціонують аналогічні екосистеми, які пропонують розробникам та користувачам набори готових графічних шаблонів та базових програмних модулів для швидкого створення нативних додатків під мобільні пристрої [21].

Беручи до уваги безперервне збільшення ринкової частки веб-орієнтованих та мобільних програмних рішень у загальній структурі сучасних інформаційних систем, а також стрімку інтеграцію технологій штучного інтелекту (зокрема, алгоритмів машинного навчання та методів глибинного інтелектуального аналізу великих масивів даних) в інструментарій розробників, логічним є висновок про неминучу і тотальну експансію сервісної моделі конструювання програмного забезпечення. Така зміна парадигми неминуче спровокує ще глибшу та більш деталізовану сегментацію професійних профілів всередині самої ІТ-індустрії. Ця диференціація відбуватиметься не лише за традиційним рольовим принципом поділу праці (бізнес-аналітики, системні архітектори, розробники коду, QA-інженери тощо) чи за специфікою застосовуваного технологічного стеку (різноманітні мови програмування, апаратні платформи, фреймворки та патерни). На перший план вийде стратифікація фахівців саме за глибиною експертизи та рівнем фундаментального розуміння процесів. Для кращого усвідомлення цього феномену доцільно провести паралель із глобальною автомобілізацією суспільства: в сучасних реаліях переважна більшість дорослого населення володіє базовими навичками керування транспортним засобом, проте лише незначний відсоток водіїв дійсно розуміє тонкощі його внутрішньої будови. І лише вузьке коло висококваліфікованих експертів (професійних автомеханіків, автоелектриків та інженерів-діагностів) володіє компетенціями для точного виявлення та усунення складних апаратних

чи програмних збоїв у конкретних моделях техніки. Судячи з усього, це є абсолютно закономірним та еволюційно виправданим шляхом розвитку будь-яких складних технічних систем та технологій, і інформаційна сфера не є винятком. Як підсумок, поточний тренд на максимальне звуження професійної спеціалізації починає активно перетинатися з ортогональною концепцією формування універсального IT-експерта — фахівця найвищого рівня, здатного компетентно орієнтуватися практично у всіх суміжних дисциплінах та розуміти архітектуру як єдине ціле, від рівня апаратної автоматизації до високорівневого управління.

Висновки до розділу 1

У межах першого розділу було проведено ґрунтовне дослідження еволюційного шляху методологій розробки програмного забезпечення, починаючи від класичних каскадних та ітеративних моделей і закінчуючи сучасними гнучкими підходами сімейства Agile. Встановлено, що масовий перехід до гнучких методик безпосередньо продиктований сучасними умовами ринку, які вимагають від розробників суттєвого скорочення термінів випуску готового продукту при одночасному забезпеченні його високої якості та надійності.

Важливим вектором розвитку галузі визначено стале підвищення рівня абстракції в процесах програмування та системного проектування, що призводить до поступового стирання жорстких меж між традиційними етапами життєвого циклу розробки. На практичному рівні сучасні архітектурні парадигми, такі як MVC, MVP та MVVM, разом із фундаментальними SOLID-принципами об'єктно-орієнтованого дизайну, знаходять своє втілення через використання потужних модульних програмних каркасів — фреймворків. Застосування таких передових архітектурних конструкцій дозволяє ефективно автоматизувати не лише написання програмного коду, але й створення моделей даних, користувацьких інтерфейсів, наборів для тестування та супровідної документації.

Суттєва оптимізація часових ресурсів під час розробки досягається завдяки впровадженню концепції єдиного наскрізного інструментарію, що виключає необхідність багаторазового ітеративного виконання супутніх процесів при внесенні змін у проект. Для забезпечення комплексної автоматизації діяльності команд корпоративні бази знань сьогодні глибоко інтегруються із системами контролю версій, інструментами управління IT-проектами та баг-трекерами, серед яких провідне місце посідає система JIRA. На етапі розгортання готових рішень найбільшу актуальність має хмарна модель PaaS, у межах якої провайдери надають повністю оснащені програмні платформи для швидкого та стабільного розміщення додатків.

Дослідження також виявило чітку орієнтацію індустрії мов програмування на мультипарадигмальність та стрімку популяризацію декларативного підходу. Використання декларативних мов дозволяє значно спростити процес розробки, оскільки вони дають змогу не описувати покроковий алгоритм дій, а лише формалізувати бажаний результат через систему правил. У перспективі, з урахуванням розвитку SaaS-концепції, прогнозується поява інтелектуальних сервісів для автоматичного генерування програмного забезпечення на основі лише формалізованих вимог користувача. Така зміна сервісної моделі призведе до ще глибшої сегментації фахівців у IT-сфері, де паралельно існуватимуть як вузькопрофільні фахівці, так і універсальні професіонали, здатні охоплювати архітектуру системи як цілісний організм.

РОЗДІЛ 2 АВТОМАТИЗОВАНІ СИСТЕМИ ТА МЕТОДИ УПРАВЛІННЯ ПРОЄКТАМИ НА ПІДПРИЄМСТВАХ

2.1 Загальні відомості про управління проєктами

У сучасних умовах стрімкого розвитку високих технологій та глобальної цифровізації економіки, управління проєктами трансформувалося з допоміжної адміністративної функції у фундаментальну наукову та інженерну дисципліну. Це особливо актуально для сфери автоматизації та комп'ютерно-інтегрованих систем, де реалізація будь-якого технічного рішення — від розробки програмного модуля до впровадження складної системи управління інтелектуальною будівлею — вимагає суворої структуризації, чіткого планування та координації ресурсів. Під управлінням проєктами розуміють застосування професійних знань, навичок, інструментів і методів до проєктної діяльності для досягнення поставлених цілей у межах визначених обмежень.

Фундаментальним поняттям даної галузі є сам «проєкт», який характеризується унікальністю результату, чітко визначеними часовими межами (наявністю дати початку та завершення) та цілеспрямованістю. На відміну від операційної діяльності, яка є повторюваною та циклічною, проєкт завжди спрямований на створення чогось нового або суттєву модернізацію існуючого об'єкта чи процесу. У контексті автоматизації програмних проєктів це може бути розробка унікального алгоритму діагностики, створення реляційної бази даних або інтеграція хмарного сервісу в інформаційну інфраструктуру підприємства.

Ефективне управління проєктом традиційно розглядається крізь призму так званого «проєктного трикутника», який охоплює три критичні обмеження: зміст (обсяг робіт), час (терміни виконання) та вартість (бюджет). Якість результату є центральною категорією, що залежить від балансу між цими чинниками. Будь-яка зміна в одному з кутів трикутника неминуче впливає на інші: наприклад, скорочення термінів розробки програмного забезпечення без залучення додаткових фінансових чи людських ресурсів може призвести до зниження показників надійності системи та збільшення кількості прихованих

дефектів. Саме тому для спеціалістів з автоматизації критично важливо використовувати математичні моделі та методи прогнозування для об'єктивної оцінки стану проєкту на кожному етапі його життєвого циклу.

Процес управління проєктом поділяється на кілька логічних фаз, що утворюють його життєвий цикл:

- Ініціація: визначення концепції, мети та основних завдань дослідження, а також офіційне затвердження теми та призначення відповідальних осіб.
- Планування: розробка календарного плану, деталізація змісту розрахунково-пояснювальної записки, визначення необхідних вхідних даних (масивів результатів тестування, показників зв'язності модулів тощо) та вибір методології розробки.
- Виконання: безпосередня реалізація проєктних завдань, написання тексту розділів, проектування структурних схем та створення програмного коду.
- Моніторинг і контроль: постійне відстеження прогресу, порівняння фактичних показників із плановими, проведення технічної діагностики та оцінка надійності розроблених компонентів.
- Завершення: оформлення кінцевої документації, підготовка презентаційних матеріалів, захист результатів перед екзаменаційною комісією та архівування здобутого досвіду.

Особливе місце в сучасній теорії управління проєктами посідає автоматизація управлінських процесів. Використання таких інструментальних засобів, як MS Project або Atlassian JIRA, дозволяє трансформувати суб'єктивне управління в точний інженерний процес. Автоматизація забезпечує наскрізну інтеграцію даних, дозволяє в реальному часі відстежувати завантаженість персоналу, автоматично генерувати звіти та оперативно реагувати на виникнення непередбачуваних ситуацій чи аварій. Для малого бізнесу та вузькоспеціалізованих ІТ-підрозділів це відкриває можливості для оптимізації стратегічних і комерційних процесів без значних капіталовкладень, забезпечуючи високу конкурентоспроможність розроблених рішень. Таким чином, володіння методами управління проєктами стає обов'язковою

складовою професійної кваліфікації інженера з автоматизації та комп'ютерно-інтегрованих технологій.

2.2 Що розуміється під проектом

Щоб глибоко осягнути сутність управління, необхідно перш за все чітко визначити, які саме фундаментальні ознаки виокремлюють проект з-поміж безлічі інших видів професійної діяльності. Функціонування будь-якої сучасної організації чи підприємства загалом можна звести до двох базових категорій активності: це стандартні операційні процеси (рутинна діяльність) та, власне, проекти. Безперечно, ці два вектори функціонування не існують ізольовано; вони здатні тісно взаємодіяти, перетинатися в робочому просторі та навіть частково дублювати один одного, оскільки володіють низкою спільних характеристик. До таких спільних рис можна віднести те, що обидва види діяльності реалізуються людськими ресурсами, функціонують в умовах жорстких обмежень (фінансових, часових, матеріальних), а також підлягають обов'язковому плануванню, безпосередньому виконанню та подальшому контролю і перевірці результатів.

Незважаючи на згадані подібності, між операційними процесами та проектами існують колосальні, принципові відмінності. Ймовірно, найголовнішим критерієм розмежування виступає той факт, що процеси є безперервними, циклічними та здатні постійно повторюватися за єдиним алгоритмом, у той час як проекти завжди являють собою щось абсолютно унікальне та суворо обмежене в часі. Це означає, що для будь-якого проекту критично важливою є наявність чітко зафіксованої точки старту та заздалегідь визначеної ознаки його остаточного завершення. Виходячи з цього, концепцію проекту можна сформулювати наступним чином: це комплексний, тимчасовий захід, який цілеспрямовано здійснюється для створення унікального, неповторного продукту, послуги чи результату, що мають суттєві, якісні відмінності від інших існуючих на ринку аналогів. Масштаби проектної діяльності можуть варіюватися в надзвичайно широких межах, охоплюючи всі без винятку ієрархічні рівні організації. До їх виконання може залучатися як один вузькопрофільний спеціаліст, так і багатотисячні міждисциплінарні

команди, а загальна тривалість життєвого циклу таких ініціатив іноді розтягується на кілька років.

Різноманітність проектної діяльності можна проілюструвати багатьма прикладами з різних сфер суспільного та економічного життя. Зокрема, до типових проектів належать процеси розроблення абсолютно нових споживчих продуктів або сервісних послуг, реалізація комплексних заходів, орієнтованих на глибоку трансформацію організаційної структури, внутрішньої політики чи корпоративного стилю управління компанією. Не менш яскравими прикладами є конструювання та впровадження інноваційних інформаційних систем, кардинальна модифікація існуючого програмного забезпечення, інженерне проектування новітніх промислових виробів, проведення масштабних політичних чи маркетингових кампаній, а також реінжиніринг та оптимізація бізнес-процесів підприємства. Важливо усвідомлювати, що незважаючи на те, що окремі проекти здатні тривати роками, їх категорично не можна класифікувати як перманентну, постійну діяльність. За своєю природою вони завжди залишаються тимчасовими і рано чи пізно досягають свого логічного фіналу. Водночас, варто чітко розмежовувати поняття «проект» і звичайну «тимчасову роботу». Фундаментальна різниця полягає в тому, що життєвий цикл проекту безповоротно завершується саме в момент стовідсоткового досягнення його глобальної мети, тоді як звичайні тимчасові роботи можуть бути призупинені, а згодом — знову відновлені без втрати сутності. Тимчасова природа проекту додатково підкреслюється тим фактом, що спеціальна робоча команда часто формується виключно під конкретне завдання і підлягає розпуску одразу після фіналізації всіх робіт.

Ще однією невід'ємною характеристикою проектної діяльності є те, що в результаті її виконання на світ з'являються ексклюзивні продукти чи послуги, які не мали стовідсоткових аналогів у минулому. Навіть якщо створений об'єкт належить до добре відомої, стандартизованої категорії товарів, він неминуче володітиме власними, специфічними ознаками та характеристиками, що виділятимуть його на тлі інших подібних об'єктів. Показовим прикладом є зведення офісних будівель: незважаючи на те, що зовні архітектурні споруди

можуть виглядати абсолютно ідентичними, кожна з них матиме унікальне внутрішнє інженерне обладнання, специфічне просторове планування, іншу географічну локацію, різних власників та цільове призначення. Наявність певних постійних, повторюваних елементів у виробничому циклі жодним чином не нівелює загальної унікальності тих інтелектуальних та фізичних зусиль, які були докладені для реалізації конкретного задуму. Підсумовуючи, можна стверджувати, що проекти та сукупність методів їх втілення, яка формує дисципліну управління проектами, кардинально відрізняються від повсякденної операційної діяльності підприємства. Ця різниця базується на двох фундаментальних стовпах — суворій обмеженості в часі (тимчасовості) та абсолютній неповторності (унікальності) кожного окремо взятого починання. І хоча різні проекти можуть мати спільні методологічні риси чи використовувати схожі технологічні елементи, кожен із них у своїй сукупності залишається глибоко індивідуальним та неповторним.

2.3 Коротка історія питання

Історичне коріння методологій управління проектами сягає сивої давнини. Вражаючі результати реалізації грандіозних проектів минулого людство має змогу спостерігати й сьогодні — це монументальні єгипетські піраміди, розгалужені стародавні іригаційні канали та велична Велика китайська стіна. Про інші, не менш масштабні починання, ми дізнаємося переважно з детальних історичних хронік та описів сучасників. До таких подій належать безпрецедентні за масштабом військові кампанії, організовані Чингісханом та Олександром Македонським, а також епохальні трансокеанські дослідницькі експедиції під керівництвом Христофора Колумба і Фернана Магеллана. У сучасному науковому світі існують ґрунтовні академічні праці, які глибоко аналізують специфічні методи проектного управління, що їх застосовували стародавні єгипетські зодчі під час зведення пірамід, або ж скандинавські вікінги в процесі ретельної підготовки та здійснення своїх складних військово-морських операцій.

Проте, теоретичний фундамент сучасного проектного менеджменту базується переважно на математичних методиках сіткового планування, які були закладені в Сполучених Штатах Америки наприкінці 1950-х років. Визначальним у цьому контексті став 1956 рік, коли Морган Уокер, фахівець хімічного гіганта «DuPont», розпочав пошук шляхів оптимізації роботи корпоративної електронно-обчислювальної машини Univac. Для досягнення цієї мети він об'єднав свої зусилля з Джеймсом Келлі, представником групи планування капітального будівництва компанії «Remington Rand». Цей дослідницький тандем здійснив революційну спробу задіяти обчислювальні потужності ЕОМ для автоматизованого складання складних планів-графіків, необхідних для реалізації масштабної програми модернізації виробничих потужностей заводів «DuPont». Результатом їхньої плідної співпраці стало створення напрочуд раціонального та математично прозорого методу формалізованого опису проекту за допомогою комп'ютерної техніки. Спочатку ця інноваційна розробка отримала назву методу Уокера-Келлі, проте згодом закріпилася в науці та практиці під загальновідомим терміном «Метод критичного шляху» (МКШ), або в англomовній літературі — Critical Path Method (CPM).

Майже одночасно з цим, але абсолютно незалежно, у надрах військово-морських сил США народжувався інший потужний інструмент — Метод оцінки та аналізу програм (Program Evaluation and Review Technique, або PERT). Ця методологія була спільно розроблена інженерами корпорації Lockheed Air Craft та фахівцями провідної консалтингової фірми Booz, Allen & Hamilton. Головною метою її створення стала гостра необхідність ефективної реалізації надзвичайно складного оборонного проекту з розробки балістичної ракетної системи «Polaris». Цей проект об'єднував зусилля близько 3800 ключових підрядних організацій та складався з колосального масиву — понад 60 тисяч окремих операцій. Впровадження методу PERT надало керівництву програми безпрецедентний рівень контролю: вони отримали можливість точно знати, які конкретні дії необхідно виконувати в кожний момент часу, хто персонально несе відповідальність за їх реалізацію, а також математично обчислювати

ймовірність своєчасного завершення кожної окремої стадії. Ефективність такого інноваційного підходу до управління виявилася настільки приголомшливою, що грандіозний військовий проект вдалося повністю завершити на два роки раніше від початково встановленого терміну. Зважаючи на такий тріумфальний успіх, методологія PERT була дуже швидко стандартизована і почала масово застосовуватися для планування та координації масштабних проектів у всіх без винятку видах збройних сил США, особливо під час взаємодії з численними підрядниками при розробці новітніх зразків озброєння.

Успішний досвід військових не залишився непоміченим у цивільному секторі: великі промислові корпорації практично відразу почали адаптувати та впроваджувати подібні управлінські методики для оптимізації процесів розробки нових видів комерційної продукції та глобальної модернізації власних виробництв. Особливо широкого поширення методологія сіткового планування набула в галузі капітального будівництва. Класичним і надзвичайно показовим прикладом є управління проектом спорудження потужної гідроелектростанції на річці Черчїлл, розташованій на півострові Лабрадор (Ньюфаундленд). Загальний бюджет цієї грандіозної будови склав 950 мільйонів доларів. Масштабні будівельні роботи тривали майже десятиліття — з 1967 по 1976 рік. Проектна структура охоплювала понад сотню окремих великих будівельних контрактів, причому вартість деяких із них сягала 76 мільйонів доларів. Вражає той факт, що вже у 1974 році фактичний хід виконання робіт за проектом випереджав затверджений розклад на цілих 18 місяців, при цьому жорстко вкладаючись у рамки попередньо спланованого кошторису витрат. Замовником цього інфраструктурного об'єкта виступала корпорація Churchill Falls Labrador Corp., яка завбачливо залучила для безпосередньої розробки проекту та оперативного управління всім процесом будівництва спеціалізовану фірму Acres Canadian Betchel.

Слід зазначити, що такий колосальний виграш у часі став можливим виключно завдяки застосуванню точних математичних методів управління складними багаторівневими комплексами робіт. У свою чергу, реалізація цих

методів на практиці стала здійсненою завдяки стрімкому розвитку обчислювальної техніки. Проте, важливо розуміти, що електронно-обчислювальні машини перших поколінь відзначалися надзвичайно високою вартістю та складністю обслуговування, тому доступ до них мали лише найбагатші державні інституції та найбільші корпорації. Отже, в історичній ретроспективі, перші формалізовані проекти являли собою переважно масштабні державні програми, що характеризувалися грандіозними обсягами робіт, величезною кількістю залучених виконавців та багатомільйонними капіталовкладеннями.

На початковому етапі розвитку ІТ-індустрії великі корпорації були змушені самотійно розробляти специфічне програмне забезпечення для підтримки та супроводу власних унікальних проектів. Однак дуже скоро на молодому ринку програмного забезпечення почали з'являтися перші комерційні системи управління проектами. Ці піонерські рішення проектувалися виключно під потужні мейнфрейми та розгалужені мережі міні-ЕОМ. Ключовою перевагою та основною характеристикою систем цього класу була їхня надзвичайна обчислювальна потужність, яка дозволяла максимально детально описувати складні багаторівневі проекти із застосуванням просунутих методів сіткового планування. Цільовою аудиторією таких систем були виключно високопрофесійні менеджери вищої ланки, які керували процесом розробки найбільших проектів і вільно орієнтувалися як у складних алгоритмах сіткового планування, так і в специфічній професійній термінології. Найчастіше безпосередня розробка архітектури проекту та надання експертних консультацій щодо управління ним делегувалися спеціалізованим консалтинговим компаніям.

Справжня революція та етап найбільш бурхливого, масового розвитку програмних інструментів для управління проектами розпочалися синхронно з появою на ринку доступних персональних комп'ютерів. Саме тоді комп'ютер перетворився на звичайний, повсякденний робочий інструмент для надзвичайно широкого кола керівників середньої та нижчої ланки. Таке безпрецедентне розширення потенційної аудиторії користувачів породило гостру потребу у

створенні систем управління проектами абсолютно нового типу. Відтепер чи не найважливішим критерієм якості такої системи стала її простота та інтуїтивна зрозумілість. Управлінське програмне забезпечення нового покоління розроблялося як універсальний засіб контролю, логіка якого мала бути очевидною для будь-якого пересічного менеджера без спеціальної математичної чи технічної підготовки, забезпечуючи можливість легкого, швидкого та ефективного включення в робочий процес. Яскравим представником цього нового класу систем стала програма Time Line. Розробники наступних версій таких систем, намагаючись будь-що зберегти їхню зовнішню простоту та дружність до користувача, водночас невпинно розширювали їхні базові функціональні можливості та обчислювальну потужність. При цьому цінова політика залишалася гнучкою, що робило такі програмні комплекси економічно доступними для компаній практично будь-якого масштабу та рівня доходів.

На сьогоднішній день у США та інших розвинених країнах світу вже остаточно сформувалися глибокі, стійкі традиції масового використання спеціалізованих систем управління проектами в найрізноманітніших сферах економічної та соціальної життєдіяльності. Цікаво зазначити, що переважну більшість серед усіх планованих та реалізованих проектів становлять ініціативи невеликого та середнього розміру. Так, масштабні дослідження, проведені авторитетним комп'ютерним тижневиком InfoWorld, яскраво продемонстрували, що близько половини (50%) усіх користувачів подібного ПЗ у США потребують систем, які здатні ефективно підтримувати та адмініструвати плани обсягом від 500 до 1000 окремих робіт (завдань). І лише 28 відсотків користувачів мають справу з надзвичайно складними розкладами, які містять понад 1000 робіт. Якщо проаналізувати ситуацію з управлінням ресурсами, то статистика свідчить, що 38 відсоткам менеджерів доводиться в межах одного проекту контролювати та розподіляти від 50 до 100 різних видів ресурсів, і тільки 28 відсотків стикаються з необхідністю управління більш ніж сотнею різноманітних ресурсних категорій. У ході цих же досліджень було виведено усереднені параметри типових розкладів для проектів різних

масштабів. Так, для категорії «малих» проектів середнім показником є наявність 81 окремої роботи та залучення 14 видів ресурсів; для «середніх» проектів ці цифри зростають до 417 робіт і 47 видів ресурсів відповідно; а класичні «великі» проекти оперують у середньому 1198 роботами та потребують координації 165 видів ресурсів.

Наведені статистичні дані є надзвичайно цінними: вони можуть слугувати чудовою відправною точкою та орієнтиром для будь-якого менеджера чи власника бізнесу, який зараз обмірковує доцільність і потенційну корисність переходу своєї організації на сучасну проектну форму управління діяльністю. Як переконливо свідчить світова практика, впровадження спеціалізованої системи керування проектами може виявитися надзвичайно ефективним, економічно виправданим та корисним інструментом навіть для реалізації зовсім невеликих, локальних проектів. Цілком природно, що паралельно з постійним розширенням кола користувачів систем проектного менеджменту відбувається безперервний процес удосконалення, розширення та адаптації методів і прийомів їхнього практичного застосування. Провідні західні комп'ютерні та бізнес-видання регулярно публікують ґрунтовні аналітичні статті, присвячені еволюції систем для управління проектами, де містяться не лише практичні поради для користувачів такого ПЗ, але й глибокий аналіз ефективності використання методик сіткового планування для вирішення нестандартних завдань у найрізноманітніших сферах управлінської діяльності.

2.4 Постановка задачі управління проектами

Традиційне, вузькопрофільне тлумачення терміна «проект» зазвичай зводиться до сукупності конструкторської та технічної документації, яка детально описує певний виріб або специфічне інженерне рішення. Проте в сучасних умовах розвитку науки про управління це поняття набуло значно ширшого, комплексного змісту. Сьогодні під проектом розуміють цілеспрямовану сукупність взаємопов'язаних заходів та дій, успішна реалізація яких має забезпечити досягнення чітко визначеної системи цілей в умовах суворих часових рамок та лімітованих матеріальних ресурсів. Найбільш

характерною та фундаментальною рисою будь-якого проекту є його унікальність: він являє собою комплекс робіт (або окремих операцій), які носять неповторний характер і, як правило, не відтворюються циклічно. З точки зору сучасної теорії наукового управління, процеси планування та безпосереднього керівництва проектною діяльністю доцільно розглядати крізь призму трьох базових рівнів. Першим є стратегічний рівень управління компанією як самостійним економічним суб'єктом. Цей рівень орієнтований на довгострокову перспективу (горизонт планування охоплює кілька років) і зосереджується на формуванні глобальних цілей та векторів розвитку організації, в межах яких ініціюється конкретний проект. Другий рівень — це оперативне управління, головним завданням якого є забезпечення стабільного, безперебійного функціонування фірми та формування надійного потенціалу для її подальшого зростання; часовий горизонт тут зазвичай коливається від одного місяця до одного року. Третій рівень являє собою поточне (або тактичне) управління, що базується на безперервній реєстрації, систематизації та глибокому аналізу інформації щодо будь-яких відхилень реального процесу від початково затвердженого плану. Його ключова мета полягає в оперативному розробленні та впровадженні управлінських рішень, спрямованих на повне усунення або максимальну мінімізацію таких небажаних відхилень.

У подальшому викладі матеріалу основний акцент буде зроблено виключно на тих методах та інструментах, що застосовуються для вирішення завдань проектного управління саме на оперативному та поточному рівнях. У спеціалізованій літературі цей комплекс завдань часто класифікують як задачі оперативно-календарного планування та управління. Варто наголосити, що проблематика розробки, адаптації та дослідження відповідних математичних методів є предметом вивчення низки економічних та економіко-математичних наукових дисциплін, серед яких ключове місце посідає теорія дослідження операцій. Незважаючи на теоретичну значущість, будь-який обчислювальний алгоритм, побудований на основі певної економіко-математичної моделі, набуває реальної практичної цінності виключно тоді, коли він фізично реалізований у вигляді функціонального програмного інструменту. Саме в цій

площині задачі управління проектами стають об'єктом пильної уваги з боку економічної інформатики та інженерії програмного забезпечення. Тісний взаємозв'язок та перетин «сфер інтересів» економіко-математичних методів оперативно-календарного планування з інструментарієм інформатики додатково зумовлений необхідністю проектування та організації надійних інформаційних баз даних, які є фундаментом для функціонування сучасного програмного забезпечення з управління проектами. Важливо розуміти, що будь-який управлінський процес завжди явно чи неявно спирається на певну цільову функцію, наявність якої дозволяє об'єктивно оцінювати кінцеві результати та загальну ефективність виконаних робіт. Згідно з класичними підходами в галузі оперативно-календарного планування, головна цільова функція формулюється як прагнення до максимальної мінімізації загальної тривалості реалізації проекту за умови жорсткого дотримання обмежень щодо доступних ресурсів.

Методологія сіткового планування та управління проектами структурно поділяється на три фундаментальні етапи: структурне планування, календарне планування та безпосереднє оперативне управління. На початковій стадії — етапі структурного планування — здійснюється глобальна декомпозиція проекту, тобто його логічне розбиття на окремі, менш масштабні та керовані операції. Під терміном «операція» (або завдання) в цьому контексті слід розуміти специфічну діяльність чи робочий процес, виконання якого неминуче вимагає витрат певного часу та залучення відповідних матеріальних, фінансових або людських ресурсів. Після визначення повного переліку робіт формується комплексна логічна схема їхньої послідовності та взаємозв'язків, яку прийнято називати мережевою моделлю проекту. Наступним кроком є етап календарного планування, під час якого розробляється деталізований календарний графік розкладу робіт. Цей документ чітко регламентує директивні моменти початку та остаточного завершення кожної запланованої операції. Побудований календарний графік, розглядаючись у нерозривній єдності з мережевою моделлю, надає управлінцю можливість ідентифікувати так звані критичні операції — завдання, будь-яка затримка чи зміна тривалості

яких автоматично призведе до зсуву фінального терміну завершення всього проекту в цілому. Заключним, третім етапом є стадія оперативного управління процесом фізичної реалізації проекту. На цьому етапі розроблена раніше мережева модель та затверджений календарний графік активно використовуються як базові інструменти для збору та формалізації звітної інформації щодо фактичного прогресу виконання робіт. Водночас, ці ж інструменти підлягають регулярному коригуванню та оновленню на основі результатів аналізу поточного стану справ, що дозволяє гнучко реагувати на виклики зовнішнього та внутрішнього середовища.

Для наочного відображення логічної послідовності виконання проектних операцій найчастіше використовують математичний апарат теорії графів. У практиці сіткового планування найбільшої популярності набули два основні різновиди таких моделей: вершинні та стрілочні графи. У концепції стрілочних графів окремі операції графічно позначаються у вигляді спрямованих стрілок (дуг), напрямом яких символізує природний плин часу (зазвичай візуалізація будується зліва направо). При цьому вузлові моменти, що фіксують фактичний початок або завершення певної операції, іменуються подіями і традиційно зображуються у вигляді точок або кружечків. Альтернативним підходом є використання вершинних графів, де самі проектні операції представляються у вигляді геометричних вузлів (вершин), а зв'язки та залежності між ними позначаються з'єднувальними стрілками. Головною перевагою вершинних графів є відсутність потреби у штучному введенні так званих фіктивних операцій для збереження логіки розкладу, хоча, з іншого боку, такий формат іноді дещо ускладнює загальне візуальне сприйняття процесу переходу від одного етапу до іншого. Розглядаючи існуючі методи презентації розкладу проектів, неможливо оминати увагою такий класичний та загальновизнаний інструмент, як стрічковий (лінійний) графік, більш відомий як діаграма Гантта. Структура графіка Гантта передбачає відкладання термінів початку та закінчення кожної операції вздовж єдиної горизонтальної часової шкали. Відповідно, довжина графічних відрізків (ліній), що символізують окремі завдання, є прямо пропорційною їхній плановій тривалості. Незважаючи на

свою простоту та наочність, цей метод має суттєвий недолік: у класичному вигляді він не дозволяє чітко візуалізувати складні логічні залежності та взаємозв'язки між різними операціями. Проте, діаграма Гантта досі залишається неперевершеним інструментом для швидкого та ефективного аналізу розподілу часових ресурсів між структурними частинами проекту.

2.5 Сучасні стандарти управління проектами

Протягом останніх п'ятдесяти років методологія управління проектами не залишалася статичною; вона перебувала у стані безперервного еволюційного розвитку та системного вдосконалення. За цей тривалий період на світовій арені сформувалася ціла низка авторитетних національних та міжнародних інституцій, головною метою яких стала розробка, систематизація та підтримка актуальних галузевих стандартів. Результатом їхньої плідної діяльності стало створення фундаментального зводу знань з управління проектами, який отримав загальновідому аббревіатуру PMBOK (від англійського Project Management Body of Knowledge). Водночас варто зазначити, що внутрішня структура та безпосереднє змістовне наповнення стандартів PMBOK можуть мати певні розбіжності залежно від країни. Така варіативність зумовлена тим, що численні національні асоціації проектного менеджменту часто формують власні, специфічні погляди на те, який саме обсяг методологічних інструментів та практик має бути включений до цього базового документа. Зважаючи на це, для уникнення будь-яких різночитань та термінологічної плутанини, виникає гостра необхідність у чіткій фіксації єдиного понятійного апарату.

У подальшому викладі матеріалу загальна структура управління проектами та вся супутня англomовна термінологія наводяться у суворій відповідності до класичного керівництва Guide to the Project Management Body of Knowledge, розробленого авторитетним Американським інститутом управління проектами (Project Management Institute — PMI). Адаптовані переклади термінів базуються на фундаментальних працях, зокрема на довідкових матеріалах (за авторством В. М. Ліберзона, директора представництва PMI), що свого часу масово поширювалися у форматі електронних довідок разом із демонстраційними

версіями відомої системи управління проектами Spider Project. Вибір саме цих джерел в якості базових орієнтирів продиктований, перш за все, їхньою історичною відкритістю, безкоштовною доступністю для вільного завантаження з офіційних веб-ресурсів та значним впливом на формування профільної термінології.

Відповідно до визнаних світових стандартів, комплексна функціональна структура управління проектами логічно поділяється на дев'ять ключових взаємопов'язаних розділів, або галузей знань. Ця архітектура повноцінно охоплює: управління координацією (Project Integration Management), яке забезпечує системну цілісність усіх елементів; управління цілями та змістом (Project Scope Management), що суворо визначає межі необхідних робіт; управління часовими ресурсами (Project Time Management) для неухильного дотримання розроблених календарних графіків; управління вартістю (Project Cost Management), що відповідає за бюджетування та фінансовий контроль; управління якістю (Project Quality Management), спрямоване на максимальне задоволення цільових вимог замовника; управління людськими ресурсами (Project Human Resource Management), що стосується грамотного формування та мотивації робочої команди; управління комунікаціями (Project Communication Management) для забезпечення своєчасного та ефективного інформаційного обміну; управління проектними ризиками (Project Risk Management), що дозволяє мінімізувати непередбачувані загрози; а також управління постачанням (Project Procurement Management), яке регламентує всі процеси зовнішніх закупівель.

Усі перелічені вище функціональні галузі не існують ізольовано; вони надзвичайно тісно переплетені між собою та утворюють єдину синергетичну систему. Для більш глибокого розуміння природи цих складних взаємозв'язків необхідно розглянути безпосередньо сам процес управління проектом у його динаміці. Згідно з провідною методологією, у межах життєвого циклу абсолютно будь-якого проекту (або окремої його фази) обов'язково та послідовно реалізуються п'ять фундаментальних груп процесів. Цей цикл незмінно розпочинається з процесів ініціації (Initiating Processes), які полягають

в офіційному ухваленні та затвердженні стратегічного рішення щодо старту проекту в цілому або переходу до його чергової фази. Далі наступає черга процесів планування (Planning Processes), під час яких відбувається детальне визначення оптимальних робочих схем, алгоритмів та маршрутів для досягнення задекларованих цілей. Наступним кроком є процеси виконання (Executing Processes), сутність яких зводиться до практичної реалізації запланованих кроків шляхом ефективної координації людських та всіх інших наявних ресурсів. Паралельно з виконанням активуються процеси контролю та управління (Controlling Processes), які передбачають безперервне спостереження, об'єктивне вимірювання проміжних результатів та оперативне внесення необхідних коригувальних дій у разі виявлення відхилень від базового плану. Завершується цей складний цикл процесами закриття (Closing Processes), головне завдання яких — це формальна фіналізація, належне документальне оформлення та офіційне завершення діяльності за проектом або його конкретною фазою.

2.6 Інформаційні системи

Базові функції управління проектами мають глибоке спільне коріння з низкою інших класичних управлінських дисциплін. Саме тому на початкових етапах передпроектного аналізу, а також під час стратегічного планування глобальних цілей проекту, фахівці зазвичай застосовують той самий набір аналітичних інструментів, що й у сфері фінансового менеджменту. Проте суттєві методологічні та інструментальні розбіжності починають яскраво проявлятися вже на стадії глибокої декомпозиції поставлених цілей. Починаючи від цього переломного моменту і аж до остаточного завершення всіх проектних робіт, домінуючу позицію займають спеціалізовані системи календарно-мережевого планування (надалі — КМП). Ці програмні комплекси забезпечують вичерпний базовий набір функцій (наведений у таблиці нижче), який є життєво необхідним для грамотного проектування ієрархічної структури робіт як окремого проекту, так і цілої групи пов'язаних проектів. Крім того, вони відповідають за точне ресурсне планування, налагодження безперебійного

обміну актуальною інформацією між усіма учасниками робочого процесу, а також за здійснення суворого контролю за ходом виконання запланованих завдань. Сучасний ринок програмного забезпечення пропонує цей сегмент продуктів як у вигляді вузькоспеціалізованих КМП-систем, так і у форматі інтегрованих КМП-модулів, що входять до складу комплексних автоматизованих систем управління підприємством (АСУП).

Якщо оцінювати наявні функціональні можливості, то всі існуючі КМП-системи можна умовно класифікувати на дві великі категорії: потужні комплекси для щоденного використання професійними проектними менеджерами та дещо спрощені інструменти для керівників, яким доводиться займатися плануванням проектів лише періодично. Втім, варто зауважити, що подібний поділ є доволі умовним і розмитим. Навіть сучасні системи так званого «непрофесійного» рівня (ціновий діапазон яких зазвичай коливається від 300 до 1000 доларів США) володіють достатньою потужністю для побудови складних багаторівневих розкладів, що налічують десятки тисяч окремих робіт. Вони здатні ефективно моделювати групи взаємопов'язаних проектів, розподіляти практично необмежену кількість залучених ресурсів і, загалом, надають доступ до повного спектру функцій, критично необхідних для успішної реалізації середньостатистичного проекту. З іншого боку, професійні КМП-системи класу high-end (вартість яких стартує від 4 тисяч доларів США і вище) стають безальтернативним вибором, коли мова йде про реалізацію надзвичайно масштабних і комплексних проектів. Вони забезпечують максимально гнучке ресурсне планування, дозволяють проводити глибокий багатофакторний аналіз потенційних ризиків тощо. Варто також додати, що окрім систем календарно-мережевого планування, в екосистемі управління проектами активно застосовуються різноманітні вузькоспеціалізовані програмні рішення, зокрема інформаційні системи для адміністрування контрактів, поглибленого математичного аналізу ризиків та засоби для тривимірного комп'ютерного моделювання.

Таблиця 2.1

Спектр функціональних можливостей систем КМП

Блок інструментів	Детальний опис функціоналу
Засоби для опису структури робіт	Здійснення детального опису логічної архітектури робіт проекту в різноманітних розрізах (WBS-структура, побудова мережеских діаграм, багаторівневе кодування за окремими етапами, структурними підрозділами або відповідальними виконавцями). Планування з використанням методу критичного шляху. Точне визначення всіх часових параметрів проекту. Математичне моделювання проектного розкладу з обов'язковим урахуванням різноманітних часових обмежень.
Засоби для ресурсного планування	Формування детального опису структури наявних ресурсів та графіків їхньої доступності (ведення календарів ресурсів). Розподіл та призначення відповідних ресурсів на конкретні роботи. Надання функцій для моделювання поведінки проекту в умовах дії різноманітних обмежень щодо використання ресурсного пулу.
Засоби для проведення вартісного аналізу	Оцінка фінансових витрат, бюджетування та безперервний контроль вартості виконання проектних робіт на кожному з етапів.
Засоби для аналізу ризиків	Ідентифікація та визначення ступеня ризиків під час оцінювання загальної тривалості як окремих етапів робіт, так і всього проекту в цілому. Математичний розрахунок імовірності успішного завершення проекту у чітко встановлені директивні терміни.
Засоби для обміну інформацією	Забезпечення оперативної публікації проектної інформації на внутрішніх (intranet) або зовнішніх (Internet) серверах компанії. Оновлення та синхронізація баз даних проекту за допомогою механізмів віддаленого доступу або інтеграції з електронною поштою. Забезпечення можливості безперешкодного експорту/імпорту та обміну інформацією з будь-якими іншими суміжними програмними додатками.
Засоби для контролю за ходом виконання проекту	Надійна фіксація базових планових показників проекту. Регулярне введення поточної актуальної інформації щодо реального стану виконання запланованих робіт, рівня завантаження виділених ресурсів, фактичних фінансових витрат тощо. Проведення глибокого порівняльного аналізу між зафіксованими плановими та досягнутими фактичними показниками. Прогнозування та моделювання подальшого ходу виконання майбутніх завдань.
Засоби для наочного подання інформації	Автоматизована генерація та створення всіх необхідних типів звітів. Забезпечення наочного, інтуїтивно зрозумілого подання складної інформації про проект у вигляді різноманітних графічних елементів (зокрема:

	календарний графік виконання робіт у вигляді діаграми Гантта, деталізована мережева діаграма зв'язків проекту, візуальні гістограми завантаження ресурсів та ін.).
--	--

2.7 Використання систем тривимірного моделювання

Значним кроком в еволюції методологій управління проектами стала глибока інтеграція спеціалізованих інструментів просторової візуалізації. Яскравим прикладом цього є ґрунтовна доповідь представника відомої корпорації Chiyoda, яка пролунала під час проведення Міжнародного симпозіуму з управління проектами COBHET'99. Основною темою цього виступу стало практичне застосування передових систем тривимірного моделювання (3-DMS) безпосередньо у контексті проектного менеджменту. Варто зазначити, що базові можливості синергії функціоналу 3-DMS та класичних систем календарно-мережевого планування (КМП) вже активно і досить успішно експлуатуються багатьма технологічними компаніями. Наприклад, відомий розробник програмного забезпечення Primavera пропонує користувачам безшовну інтеграцію власної КМП-системи із сучасними системами автоматизованого проектування (САПР), що забезпечує значно вищий рівень наочності під час презентації та інформаційного супроводу проекту. Проте фахівці з Японії пішли значно далі у розвитку цього перспективного напрямку.

Компанія Chiyoda, що має багаторічну спеціалізацію на виконанні складних інжинірингових робіт у сфері будівництва масштабних нафтопереробних заводів, впровадила по-справжньому інноваційний підхід. Корпоративна інформаційна система цієї компанії, спираючись на високодеталізовані тривимірні моделі фізичних об'єктів, дає змогу повністю відтворити майбутнє підприємство у віртуальному комп'ютерному середовищі ще до моменту фактичного старту будь-яких будівельно-монтажних робіт на локації. Унікальність даної технології полягає в тому, що всі без винятку спроектовані інженерні компоненти та архітектурні вузли жорстко прив'язані до конкретних етапів графіка робіт та виділених на них ресурсів. Такий підхід відкриває перед менеджерами безпрецедентну можливість динамічно

змінювати та оптимізувати загальний план проекту. Фахівці можуть коригувати послідовність виконання завдань та графіки завантаження ресурсів, безпосередньо маніпулюючи графічними елементами тривимірної моделі споруджуваного об'єкта на екрані монітора.

Завдяки такому комплексному віртуальному моделюванню, система дозволяє завчасно та з високою точністю знаходити відповіді на критично важливі просторові, ергономічні та логістичні питання. Наприклад, інженери можуть математично обчислити, якими саме темпами необхідно зводити один промисловий корпус, щоб його металоконструкції жодним чином не перешкоджали вільному розвороту стріли висотного баштового крана, який паралельно обслуговує будівництво сусіднього об'єкта. Крім того, програма допомагає раціонально оптимізувати просторове складування будівельних матеріалів та конструкцій таким чином, щоб повністю уникнути перетину транспортних потоків та мінімізувати затори на території будівельного майданчика. Більше того, надзвичайно високий рівень деталізації цифрового середовища дозволяє перевіряти навіть суто ергономічні параметри експлуатації: наприклад, система здатна проаналізувати, чи зможе середньостатистичний робітник без застосування додаткового обладнання вільно дотягнутися до запірного вентиля на певному специфічному сегменті трубопроводу.

Ще однією вагомою перевагою впровадження таких візуальних інновацій стало кардинальне вирішення логістичних та комунікаційних проблем, що є особливо актуальним, враховуючи, що головний офіс компанії Chiyoda розташований у Японії, тоді як її безпосередні масштабні будівельні роботи реалізуються на території Саудівської Аравії. Повна відмова від застарілого паперового документообігу на користь динамічних цифрових моделей, у поєднанні з використанням потужних супутникових каналів зв'язку, дозволила корпорації налагодити миттєвий та безперервний обмін критично важливими даними між штаб-квартирою та віддаленим будівельним майданчиком. Для порівняння варто згадати, що у минулому лише фізичне пересилання паперових креслень та специфікацій міжнародною поштою займало щонайменше п'ять діб.

Підсумовуючи розгляд цих інформаційних технологій, необхідно окремо наголосити на фундаментальному факті: сфера застосування передових методів управління проектами жодним чином не обмежується виключно підприємствами з яскраво вираженим проектним типом виробництва. У сучасних ринкових умовах абсолютно всі організації та установи щоденно реалізують найрізноманітніші проекти, чи то масштабна підготовка до участі у міжнародній галузевій виставці, плановий ремонт виробничого устаткування, чи комплексний процес розробки та виведення на ринок абсолютно нової продукції. Зрештою, варто завжди пам'ятати, що й сама організація як така існує та безперервно еволюціонує в рамках свого власного, глобального життєвого проекту.

2.8 Програмне забезпечення MS Project

Як типові та найвідоміші представники спеціалізованого програмного забезпечення для управління проектами традиційно розглядаються такі потужні системи, як TimeLine від компанії TimeLine Inc. та Microsoft Project від корпорації Microsoft. У подальшому детальний розгляд технологічних аспектів автоматизованого планування та супроводу проектної діяльності буде здійснюватися саме на прикладі інструментарію програмного пакета MS Project. Історичний шлях розвитку цього популярного продукту розпочався зі створення компанією Microsoft перших версій, які були адаптовані ще для операційної системи MS DOS. Згодом, реагуючи на стрімкий розвиток технологій, розробники презентували оновлені варіанти для графічної оболонки Windows 3.x (версії 3 та 4), а пізніше світ побачили такі знакові та вдосконалені релізи, як MS Project 98, MS Project 2000 та комплексне рішення Microsoft EPM. Важливо акцентувати увагу на тому факті, що хоча програма MS Project безпосередньо не поширюється у базовому складі стандартного пакета MS Office і є цілком самостійним комерційним продуктом, вона повноцінно належить до цієї єдиної програмної родини. Це означає, що її користувацький інтерфейс, логіка побудови меню та базові концепції функціонування максимально погоджені та наближені до ідеології добре знайомих офісних

додатків, таких як MS Word чи MS Excel. Додатковою і беззаперечною перевагою MS Project є глибока інтеграція з мережевими можливостями операційного середовища MS Windows, що дозволяє легко та ефективно організовувати спільну роботу цілої групи користувачів у межах єдиної корпоративної обчислювальної мережі.

Говорячи про глобальні корпоративні рішення, варто окремо виділити платформу Microsoft Office Enterprise Project Management (скорочено MS EPM). Цей масштабний комплекс спеціально орієнтований на великі організації та компанії, де одночасно реалізується значна кількість проектних ініціатив і виникає гостра потреба в їх системній координації. Платформа забезпечує вище керівництво цілісною картиною стану всіх портфелів проектів, гарантує централізоване управління наявними ресурсами та надає вичерпну аналітичну звітність для ухвалення стратегічних рішень. Функціонал MS EPM відкриває надзвичайно широкі можливості для детального планування, постійного відстеження поточного стану виконання робіт, швидкого доступу до необхідної проектної інформації та налагодження командної взаємодії. Це стає можливим завдяки вдалому об'єднанню кількох ключових програмних продуктів та технологій. Основною робочою станцією виступає настільна система Microsoft Office Project Професійний випуск 2003, яка розкриває свій потенціал у тісній зв'язці з серверною частиною. Технологія Microsoft Office Project Web Access забезпечує віддалений доступ користувачів до баз даних проектів і ресурсів через звичайний веб-інтерфейс, дозволяючи оперативно формувати різноманітні звіти. Своєю чергою, ядром корпоративної інфраструктури є Microsoft Office Project Server 2003, який відповідає за підтримку всіх управлінських функцій, вимагаючи для своєї повноцінної експлуатації обов'язкового розгортання потужного сервера баз даних на базі SQL Server 2000.

Розглядаючи детальніше функціональні можливості безпосередньо Project Professional 2003 у складі платформи EPM, слід відзначити її передовий інструментарій для комплексного планування та моніторингу. Він включає автоматизовані правила оновлення даних та інструменти для глибокого аналізу

проектних бюджетів. Користувачі отримують максимально зручний доступ до інформації з можливістю гнучкого налаштування виглядів, групування та сортування даних, а також використання наочних графічних індикаторів для оперативного виявлення проблемних зон та проведення класичного PERT-аналізу. Система пропонує розширені функції аналітики, зокрема застосування методології освоєного обсягу, налаштування специфічних календарів завдань, детальний розподіл матеріальних та трудових ресурсів, крос-проектний розрахунок критичного шляху та пріоритезацію завдань із використанням користувацьких формул і кодів. Потужний блок управління ресурсами дозволяє компанії оперувати єдиним спільним пулом корпоративних ресурсів, використовувати вбудований майстер їхньої оперативної заміни, вести детальні таблиці обліку робочого часу та навіть здійснювати інтелектуальний підбір співробітників у проектні команди на основі їхніх підтверджених професійних навичок. Невід'ємною складовою успішної роботи є також надійна система управління проектною документацією, що реалізується через глибоку інтеграцію з платформою документообігу Windows SharePoint Services (WSS) 2003. Вона гарантує строгий контроль версій документів, управління правами доступу, ведення реєстрів ризиків та адміністрування внутрішніх порталів.

Аналізуючи загальний баланс сильних та слабких сторін рішення Microsoft EPM, фахівці відзначають низку вагомих переваг. Насамперед, це абсолютна доступність інформації, що дозволяє керівництву консолідувати дані про всі ініціативи в масштабах цілої організації та значно оптимізувати процес прийняття рішень. Платформа суттєво підвищує ефективність управління виробничими процесами та розподілу ресурсів, що неминуче призводить до скорочення фінансових витрат, зменшення часу розробки та підвищення якості фінального результату. Потужні механізми комунікації створюють єдиний простір для спільної роботи всіх членів команд, а продумана гнучка архітектура дозволяє тонко адаптувати систему під специфічні бізнес-процеси підприємства, інтегрувати її з іншими інформаційними системами та легко масштабувати в міру розвитку бізнесу з дотриманням найвищих стандартів безпеки. Водночас, ця система має і певні об'єктивні обмеження, про які варто

пам'ятати при її впровадженні. Зокрема, до її слабких сторін традиційно відносять дещо обмежений інструментарій для комплексного управління проектними ризиками та відсутність низки специфічних можливостей для глибокого управління складними бюджетами. Крім того, на практиці користувачі можуть відчувати незручності через відсутність підтримки готових вертикальних галузевих рішень та стикатися з недостатністю вбудованих механізмів для реалізації концепції глобального планування за принципом «зверху-вниз».

Висновки до розділу 2

У другому розділі було проведено комплексне дослідження теоретичних засад та інструментальних засобів автоматизації процесів управління проектами на сучасних підприємствах. Встановлено, що проєкт являє собою унікальний і тимчасовий захід, цілеспрямовано спрямований на створення неповторного продукту чи послуги, який принципово відрізняється від безперервної операційної діяльності. Ефективне управління проєктною діяльністю базується на суворому контролі так званого «проєктного трикутника», що вимагає постійного балансування між змістом (обсягом робіт), часом виконання та виділеним бюджетом.

Доведено, що методологія управління проєктами пройшла тривалий шлях еволюції від базових математичних алгоритмів сіткового планування, таких як Метод критичного шляху (CPM) та Метод оцінки і аналізу програм (PERT), до створення глобальних міжнародних стандартів. На сьогодні фундаментальним зводом знань у цій галузі є стандарт PMBOK, який логічно структурує управління на дев'ять ключових галузей знань (управління цілями, часом, вартістю, якістю тощо) та п'ять базових груп процесів: ініціацію, планування, виконання, контроль і завершення.

Для практичної реалізації завдань оперативно-календарного планування критично необхідним є використання спеціалізованих інформаційних систем календарно-мережевого планування (КМП). Ці програмні комплекси дозволяють автоматизувати процеси декомпозиції робіт, оптимізувати розподіл

ресурсів, здійснювати вартісний аналіз та наочно візуалізувати розклади (наприклад, за допомогою діаграм Гантта). Перспективним інноваційним напрямком є інтеграція таких систем із технологіями тривимірного моделювання (3-DMS), що дозволяє віртуально відтворювати об'єкти ще до початку робіт і динамічно оптимізувати процеси безпосередньо на 3D-моделі.

Як потужний еталонний інструмент автоматизації було детально розглянуто програмне забезпечення Microsoft Project та корпоративну платформу MS Enterprise Project Management (EPM). Цей комплекс забезпечує керівництво підприємства консолідованою інформацією про всі портфелі проєктів, дозволяє централізовано керувати пулом корпоративних ресурсів та вести детальну аналітику. Незважаючи на деякі обмеження щодо інструментарію управління ризиками, впровадження таких комп'ютерно-інтегрованих систем дозволяє трансформувати суб'єктивне управління в точний інженерний процес, мінімізуючи витрати та значно підвищуючи загальну ефективність підприємства.

РОЗДІЛ 3 ПРАКТИЧНА МЕТОДОЛОГІЯ ВИКОРИСТАННЯ MS PROJECT

Запропонована методологічна концепція являє собою структурований комплекс дієвих прийомів та практичних рекомендацій, спрямованих на ефективне застосування програмного середовища MS Project для вирішення повсякденних і стратегічних завдань керівника проекту (КП). Варто зауважити, що даний підхід не позиціонується як абсолютно універсальний, адже його застосування має певні межі та обмеження. Для кращого розуміння специфіки роботи, доцільно актуалізувати базові вимоги до функціоналу проектного менеджера. Хоча для експертів з багаторічним досвідом ці аспекти є очевидними, фахівцям-початківцям або тим, хто лише планує обійняти керівну посаду, така систематизація буде вкрай корисною. Загальновідомо, що процес розробки програмного забезпечення завжди орієнтований на продукування унікального та інноваційного цифрового продукту. Відповідно, на кожній окремій фазі життєвого циклу розробки перед керівником постає специфічний спектр викликів та завдань.

Ще на етапі ініціації та попереднього планування, до фактичного старту робіт, від менеджера очікують чітких відповідей на два фундаментальні запитання: якою буде загальна тривалість реалізації задуманого і яких фінансових витрат вимагатиме цей процес. При цьому абстрактні формулювання на кшталт «приблизно через півроку» є абсолютно неприйнятними для замовника; необхідна чітка, обґрунтована оцінка максимальних термінів та бюджету, так звана оцінка «зверху».

Безпосередньо під час активної фази реалізації проекту головною місією КП стає забезпечення неухильного дотримання затверджених термінів, оскільки будь-які часові затримки мають прямий та негативний вплив на загальну вартість розробки. Зважаючи на те, що абсолютно кожен проект неминуче стикається з непередбачуваними обставинами та форс-мажорами, ризик зриву дедлайнів існує постійно. Теоретично можливе й несподіване скорочення термінів виконання, проте в реальній інженерній практиці такі

випадки трапляються вкрай рідко. Відтак, ключова компетенція керівника полягає у здатності миттєво та адекватно реагувати на будь-які відхилення, щоб максимально нівелювати їхні деструктивні наслідки. Найбільш дієвим механізмом досягнення цієї мети є педантичне та детальне планування, безперервний моніторинг потенційних загроз та регулярне, проактивне коригування робочих графіків. На завершальній стадії, коли продукт готовий, керівник проекту здійснює ретроспективний аналіз виконаної роботи. Основний акцент робиться на ретельному порівнянні фактичних показників із початковими плановими графіками, а також на виявленні та аналізі глибинних причин будь-яких зафіксованих відхилень чи розбіжностей.

Звертаючись безпосередньо до інструментарію MS Project, варто зазначити, що попри позірну візуальну та функціональну перевантаженість, концептуальна архітектура цієї програми є напрочуд прозорою. Базова логіка системи спирається на взаємодію всього трьох ключових сутностей: конкретних завдань, наявних ресурсів та робочих календарів, які об'єднані системою зв'язків. Технічно — це спеціалізована база даних, доповнена ергономічним інтерфейсом для маніпуляцій із зазначеними сутностями та оснащена базовим рівнем автоматизації, що дозволяє програмі самостійно перераховувати показники на основі введених користувачем змін. Кожна задача в системі характеризується чіткою тривалістю, обсягом робіт, закріпленим за нею виконавцем та широким спектром додаткових атрибутів. У разі специфічних потреб функціонал дозволяє створювати власні, кастомні властивості. Крім того, всі роботи в проекті можуть бути жорстко пов'язані між собою різноманітними логічними залежностями (наприклад, співвідношеннями типу попередник-наступник).

Що стосується ресурсів, то окрім стандартних описових параметрів, критично важливою є можливість налаштування їхньої індивідуальної доступності у часі за допомогою персональних календарів. Кожен такий ресурс згодом призначається на виконання певного завдання. Базуючись на цьому масиві інформації, алгоритми MS Project здатні генерувати різноманітні візуальні представлення, активно використовуючи механізми фільтрації,

сортування та багаторівневого групування даних. Найважливішою ж функцією системи є автоматизований розрахунок директивних дат старту та фінішу для кожної задачі, який здійснюється з урахуванням логічних зв'язків та графіків доступності призначеного персоналу.

Процес підготовки детального плану починається з глибокого аналізу технічного завдання, після чого необхідно дати обґрунтовані відповіді щодо загальних часових витрат, кількісного та якісного складу необхідної команди фахівців, а також очікуваних сумарних трудовитрат. Для цього в середовищі MS Project формується чорновий розклад, який передбачає послідовну фіксацію всіх робіт, необхідних для досягнення цілей. Процедура трансформації технічних вимог у перелік конкретних кроків є окремою складною методикою, проте безпосереднє створення плану в програмі відбувається через низку послідовних кроків. Вони охоплюють складання вичерпного реєстру завдань, встановлення чітких логічних залежностей між ними (визначення того, які результати є критичними для переходу до наступних кроків), закріплення відповідальних виконавців, вирівнювання ресурсного завантаження та фінальне балансування отриманого розкладу.

Під час роботи над проектом варто дотримуватися кількох концептуальних рекомендацій. Насамперед, настійно не рекомендується використовувати так звані сумарні завдання для створення ієрархічної декомпозиції робіт. Значно ефективнішим підходом є розміщення всіх без винятку завдань у єдиному плоскому лінійному списку. Попри можливий початковий дискомфорт, такий метод у майбутньому нівелює безліч структурних проблем, а для категоризації робіт краще застосовувати додаткові користувацькі поля. Також слід відмовитися від популярного, але неефективного на великих обсягах даних методу розстановки залежностей за допомогою перетягування мишею (Drag & Drop). Оптимальним рішенням є виведення окремого стовпця «Попередники» та явне, ручне введення ідентифікаційних номерів пов'язаних завдань. Тривалість виконання однієї задачі не повинна перевищувати двох робочих тижнів; якщо ж прогнозований термін більший — це прямий сигнал до необхідності подальшої декомпозиції завдання на менші підзадачі. Практика

показує ефективність спрощеної шкали оцінювання: примітивна дія займає два дні, робота середнього рівня складності — один тиждень, а складна — максимум два тижні, причому частка останніх має бути мінімальною. Такий підхід гарантує високу швидкість формування базового плану. Звісно, така оцінка може здаватися дещо наближеною, проте на невеликих проектах похибки тривалості окремих завдань зазвичай взаємно компенсуються, тоді як на мікропроектах виникає потреба у застосуванні більш точних метрик. Що стосується кадрового забезпечення, слід категорично уникати закріплення кількох виконавців за одним завданням. Виняток становить лише методологія парного програмування, коли двоє фахівців фізично працюють над одним кодом спільно; у всіх інших випадках задачу слід дробити. При первинному розподілі робіт головним критерієм має бути виключно професійна спеціалізація та рівень кваліфікації працівника, без огляду на рівномірність його завантаження на цьому етапі. Для логічного поділу загального обсягу робіт на глобальні фази можна обережно використовувати сумарні завдання, встановлюючи між ними послідовні зв'язки, що дозволяє отримати укрупнену структуру етапів.

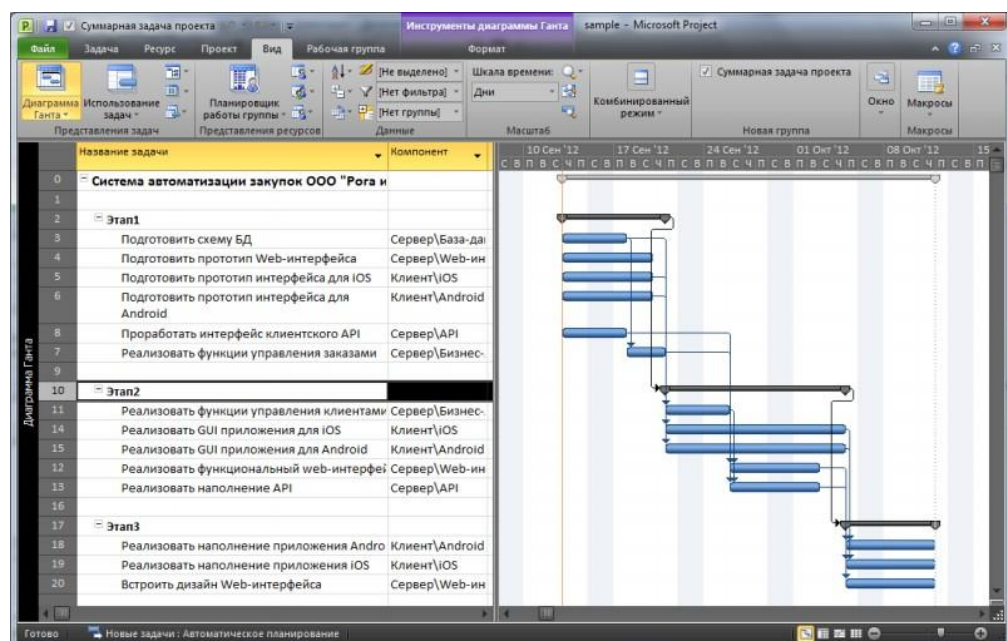


Рисунок 3.1 Відображення переліку завдань із загальним розподілом на основні проектні етапи

Центральним елементом усієї описаної методології є процес балансування розкладу. Головне завдання на цій стадії — трансформувати хаотичний первинний план у стрункий графік, де робоче навантаження буде максимально рівномірно розподілене між усіма членами команди протягом усього життєвого циклу проекту. Оскільки перший варіант плану зазвичай далекий від ідеалу, його необхідно ретельно впорядковувати в ручному режимі. Для цього застосовується інструмент групування завдань за конкретними виконавцями, а для кращої візуалізації послідовності — додаткове сортування за датою початку робіт.

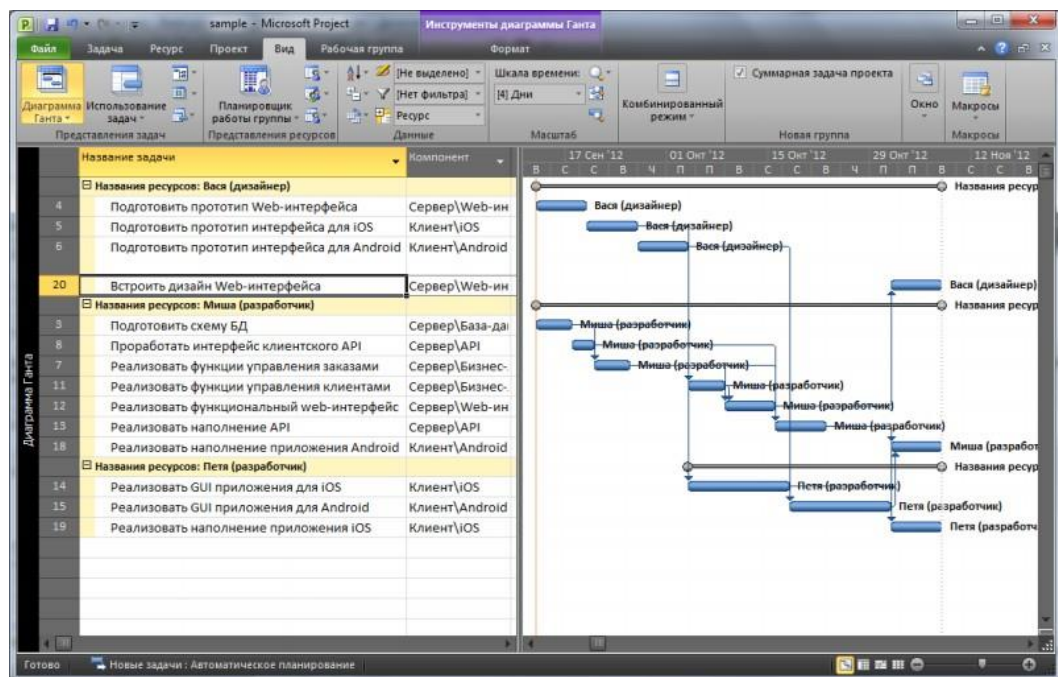


Рисунок 3.2 Застосування інструменту групування запланованих робіт за відповідальними виконавцями

Важливо розуміти, що класичні графіки ресурсного завантаження, які часто полюбляє керівництво для оцінки готового проекту, є абсолютно неефективними на етапі активного планування. Вони здатні лише констатувати факт перевантаження, але не надають інструментів для аналізу причин та пошуку шляхів вирішення проблеми. Сама ж процедура ручного балансування зводиться до мінімізації тривалості кожного окремого етапу через вирівнювання навантаження. Це досягається шляхом передачі частини завдань від перевантажених співробітників до тих, у кого в розкладі утворилися «вікна» простою. Крім того, активно застосовується перенесення певних некритичних

робіт на наступні етапи, якщо їх виконання гальмує закриття поточної фази. Зворотна дія також можлива: у разі наявності вільного часу в спеціаліста, йому можуть бути делеговані завдання з майбутніх періодів. Увесь цей ітеративний процес, хоч і вимагає постійного ручного перерахунку та вирівнювання після кожної зміни, насправді є цілком осяжним у часі. Наприклад, балансування річного розкладу для команди з восьми осіб, розбитого на чотири ключові фази, зазвичай потребує не більше години зосередженої роботи. Після завершення балансування проводиться фінальний аудит логічних зв'язків та перевірка коректності розподілу компетенцій.

Фінальним акордом на етапі планування є обов'язкове закладання резервів на управління ризиками. Навіть без застосування складних математичних моделей ризик-менеджменту, необхідно передбачити буферний час на випадок форс-мажорів, таких як лікарняні працівників чи виявлення неврахованого обсягу робіт. Практичний спосіб реалізації цього підходу полягає у додаванні в кінець кожного етапу фіктивного завдання з умовною назвою «інші додаткові роботи» для кожного залученого ресурсу. Завдяки низькому пріоритету, при вирівнюванні ці резервні блоки автоматично зміщуються у фінал етапу. Тривалість таких буферів визначається індивідуально, залежно від складності проекту, стану здоров'я команди та загального рівня невизначеності, і на практиці становить від чверті до третини від загальної тривалості відповідної фази. Завдяки всім цим комплексним маніпуляціям керівник отримує максимально реалістичний та життєздатний проектний план. З таким інструментом можна аргументовано захищати перед замовником фінальні терміни реалізації, а також мати чітке уявлення про загальні трудовитрати. Якщо ж виникає потреба в екстреному скороченні термінів, менеджер може оперативно змодельовати нову ситуацію, перебалансувавши наявний план із залученням нових виконавців.

З моменту офіційного запуску проекту в розробку, створений та затверджений план перетворюється на основний інструмент моніторингу та контролю. Робоча рутинна керівника зводиться до безперервного циклу: постановка конкретних завдань членам команди, фіксація прогресу та статусу

виконання в програмі, а також оперативне коригування графіків у разі виявлення критичних відхилень. Способи комунікації завдань можуть варіюватися від використання гнучких коротких ітерацій (спринтів) до надання роздрукованих діаграм Гантта або ж глибокої автоматизованої інтеграції MS Project із системами управління життєвим циклом додатків, наприклад TFS. Ключовим фактором успіху є не обраний засіб, а суворе регулярність оновлення даних у системі — щонайменше один або два рази на тиждень. Це дозволяє керівнику тримати руку на пульсі та вчасно виявляти так звані «вузькі місця». Дуже ефективним прийомом для локалізації проблем є використання групування завдань за функціональними компонентами продукту. Нерідко трапляються ситуації, коли загальна статистика проекту демонструє випередження графіка, проте глибокий аналіз в розрізі компонентів виявляє серйозне локальне відставання одного з розробників через складну системну помилку. Використання лише усереднених показників приховало б цю проблему до самого фіналу етапу, коли вживати антикризові заходи було б уже запізно.

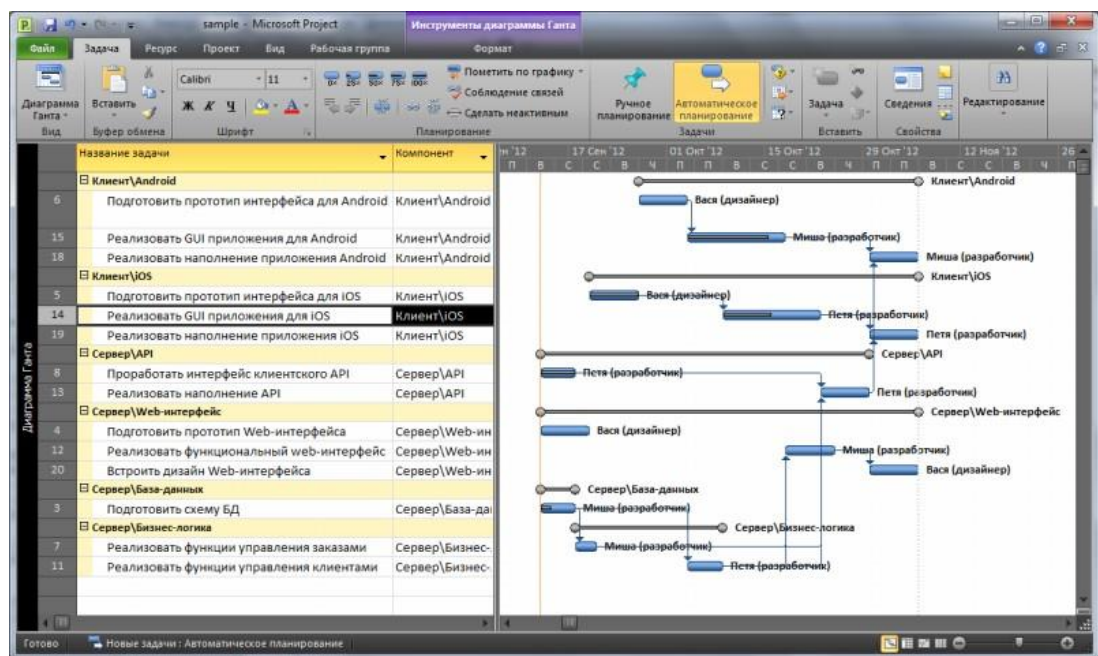


Рисунок 3.3 Моніторинг статусу виконання з використанням структурного групування за компонентами системи

В процесі актуалізації статусу робіт перевага віддається фіксації відсотка їх виконання, уникаючи безпосереднього фізичного зсуву завдань по сітці

календаря. Глобальні відхилення від графіка найпростіше відстежувати, аналізуючи положення головної сумарної задачі всього проекту відносно поточної дати. Альтернативна стратегія передбачає постійне «виштовхування» незавершених робіт уперед, у майбутнє. У такому разі для контролю змін критично необхідно використовувати функцію базового плану — збереженого еталонного знімка початкового стану розкладу. Для візуального порівняння поточного стану справ із цим еталоном застосовується спеціальний вид відображення «Діаграма Гантта з відстеженням». Проте в динамічних проектах із високим ступенем мінливості вимог такий підхід може ускладнити роботу. Тому часто раціональнішим є впровадження жорстких контрольних точок (віх), які символізують завершення важливих етапів, і моніторинг відхилень від базового плану здійснюється виключно по цих критичних індикаторах.

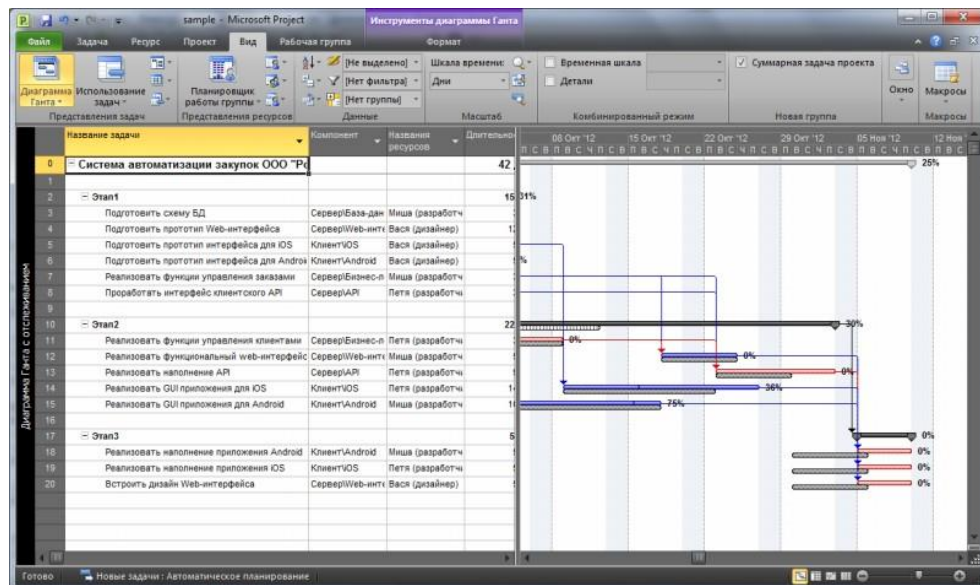


Рисунок 3.4 Візуалізація відхилень на діаграмі Гантта з функцією відстеження базового плану

Окремої уваги заслуговує підхід до структурування та категоризації робочого масиву. Зважаючи на те, що архітектура сумарних завдань в MS Project занадто жорстко прив'язана до хронологічної послідовності їх виконання, спроба змінити кут зору на проект може призвести до руйнування всієї візуальної структури. Вирішенням цієї проблеми є активне використання кастомних, користувацьких полів. Програма надає доступ до пулу резервних колонок із гнучкими налаштуваннями, поведінку яких можна задавати

самостійно. Зокрема, на базі стандартного текстового поля «Текст1» можна легко створити новий атрибут «Компонент», налаштувавши для нього випадаючий список із переліком архітектурних модулів системи.

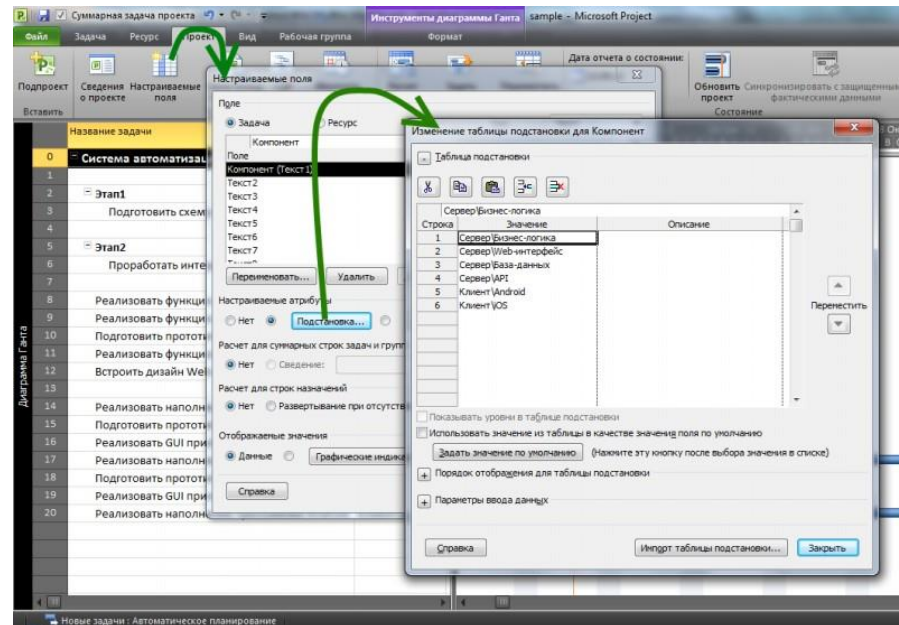


Рисунок 3.5 Процесс наладки та створення нового користувацького атрибута

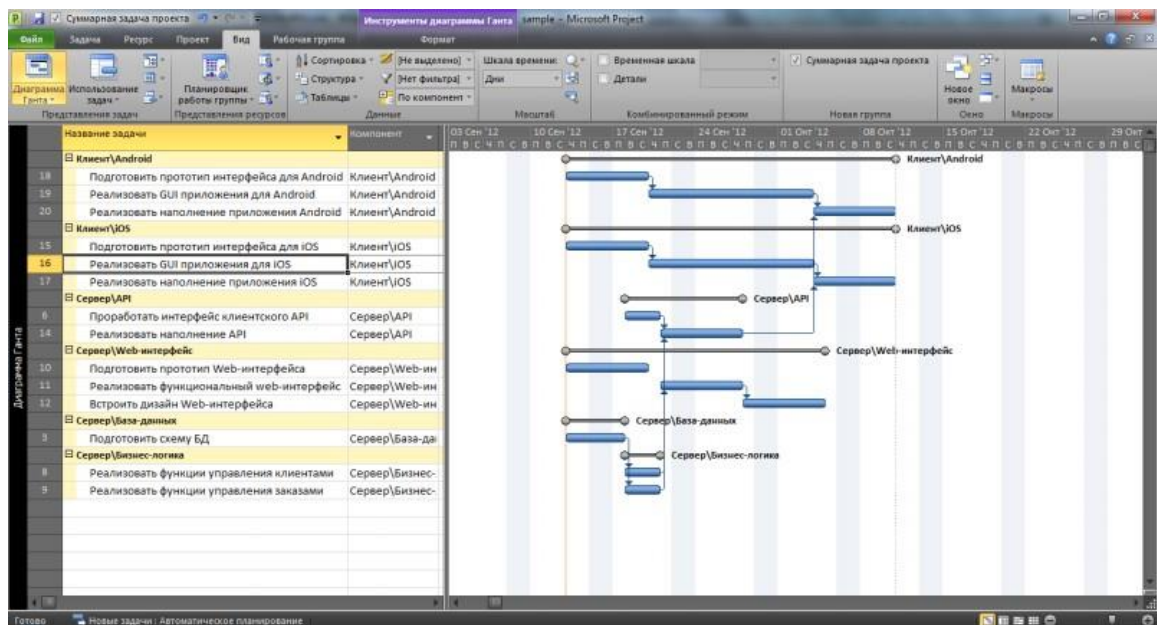


Рисунок 3.6 Приклад групування робочого простору за створеним атрибутом компонента

Такий підхід відкриває можливості для глибокої багатовимірної аналітики, дозволяючи категоризувати роботи одразу за кількома критеріями, наприклад,

розмежовуючи безпосередню розробку коду, тестування та написання технічної документації. Більше того, розширені налаштування MS Project дозволяють автоматизувати процес візуального оформлення: наприклад, можна задати алгоритми, за якими завдання, що належать до різних підсистем, будуть автоматично зафарбовуватися у відповідні кольори на діаграмі Гантта. Це не вимагає написання додаткових макросів і виконується виключно штатними засобами системи. Загалом, комбінація користувацьких атрибутів та вбудованих механізмів фільтрації і групування формує потужний аналітичний інструмент, здатний надати керівнику вичерпні відповіді на будь-які операційні запити.

На фінальній стадії, коли всі роботи успішно виконано, актуальність оперативного планування в середовищі MS Project дещо знижується, оскільки фокус уваги зміщується з планових метрик на аналіз фактичних результатів та витрат. Для збору та обробки фактичних трудовитрат значно ефективнішим є використання серверних рішень рівня MS Project Server, хоча це вже виходить за рамки поточного опису. Підсумовуючи, запропонована методологічна модель базується на реальному практичному досвіді та доведеній ефективності в процесі управління розробкою складного програмного забезпечення. Її простота, логічність та адаптивність дозволяють менеджерам успішно контролювати хід виконання проектів та дотримуватися затверджених дедлайнів. Безумовно, жодна система не гарантує абсолютної безпомилковості, і невдачі іноді трапляються. Проте, як показує практика, першопричиною таких збоїв переважно стають не недоліки самого інструментарію планування, а фатальні помилки на етапі попереднього збору вимог та постановки технічного завдання, що призводить до створення продукту, який концептуально не відповідає первинним очікуванням.

Висновки до розділу 3

У третьому розділі було запропоновано та обґрунтовано практичну методологію використання програмного середовища MS Project для ефективного управління проектами з розробки програмного забезпечення.

Визначено, що основними завданнями керівника проєкту на різних етапах його життєвого циклу є: точне прогнозування термінів і бюджету на етапі ініціації, забезпечення дотримання графіків і своєчасне реагування на відхилення під час активної фази виконання, а також ретроспективний аналіз результатів при завершенні робіт.

Детально розглянуто алгоритм підготовки базового плану проєкту. Доведено експлуатаційну ефективність використання плоского лінійного списку завдань (відмовившись від сумарних завдань для ієрархічної декомпозиції), явної розстановки логічних залежностей (без використання Drag & Drop) та жорсткого обмеження тривалості однієї задачі до двох тижнів із закріпленням за нею виключно одного виконавця. Центральним етапом підготовки плану визначено ручне балансування розкладу. Цей процес дозволяє мінімізувати терміни виконання етапів за рахунок рівномірного розподілу навантаження між членами команди (передача завдань від перевантажених співробітників до вільних). Крім того, обов'язковим елементом планування є закладання буферних часових резервів (фіктивних завдань) наприкінці кожного етапу для управління ризиками та форс-мажорними обставинами.

Для етапу практичної реалізації проєкту запропоновано дієві механізми моніторингу та контролю. Обґрунтовано доцільність регулярної (1–2 рази на тиждень) актуалізації відсотка виконання завдань без їхнього фізичного зсуву по сітці календаря. З метою гнучкої аналітики та раннього виявлення локальних затримок розроблено підхід із використанням користувацьких полів (наприклад, атрибута «Компонент») для групування завдань за функціональними модулями системи. Для відстеження ж глобальних відхилень рекомендовано застосовувати базовий план та відображення «Діаграма Гантта з відстеженням», контролюючи розбіжності за ключовими контрольними точками (віхами).

У підсумку доведено, що використання такої структурованої, простої та логічної методології в середовищі MS Project дозволяє успішно контролювати хід виконання робіт і дотримуватися затверджених дедлайнів. Практичний досвід підтверджує, що можливі збої у розробці найчастіше пов'язані не з

недоліками інструментарію планування, а з фатальними помилками на початковому етапі постановки технічного завдання та збору вимог до продукту.

РОЗДІЛ 4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Розрахунок штучного освітлення у приміщенні

Штучне освітлення застосовується при недостатньому природному освітленні, а також у вечірній та нічний час. Освітлення робочих місць сприяє створенню безпечних і нормальних умов роботи.

У разі неправильного вибору освітлення погіршується зір, а також виникають різні зорові захворювання.

Неправильний вибір світильників небезпечних приміщень може викликати нещасний випадок.

Освітлення буває загальне, місцеве та комбіноване. В свою чергу загальне підрозділяється на загальне рівномірне та загальне локалізоване освітлення.

Загальне рівномірне освітлення використовується для створення по всьому приміщенню рівномірного освітлення.

Загальне локалізоване освітлення необхідне в тих випадках, коли в місцях робіт треба забезпечити велику освітленість.

Місцеве освітлення створює потрібну освітленість тільки зони виконання робіт на невеликій площі. Комбіноване освітлення – поєднання загального та місцевого освітлення. За призначенням штучне освітлення поділяється на робоче, чергове, аварійне, евакуаційне, охоронне.

Робоче освітлення призначене для нормальної освітленості приміщень, частин відкритого простору для роботи, проходу людей та руху транспорту.

Охоронне освітлення застосовується при відсутності спеціальних технічних засобів охорони і передбачається вздовж границь територій, що охороняються в нічний час. Аварійне освітлення необхідно для продовження роботи у випадку виходу зі строю робочого освітлення.

Евакуаційне освітлення встановлюється в проходах промислових споруд з чисельністю робочих більше 50 чоловік.

Накреслимо в масштабі план і розріз приміщення, позначаємо розташування світильників. Визначаємо висоту їх підвісу h , проставляємо необхідні розміри (рис. 3.1 та 3.2). Лампа ЛД – 40 має номінальний світловий

потік $\Phi_d=2340$ лм; світильник ЛПО-01 відноситься до умовної групи 8.

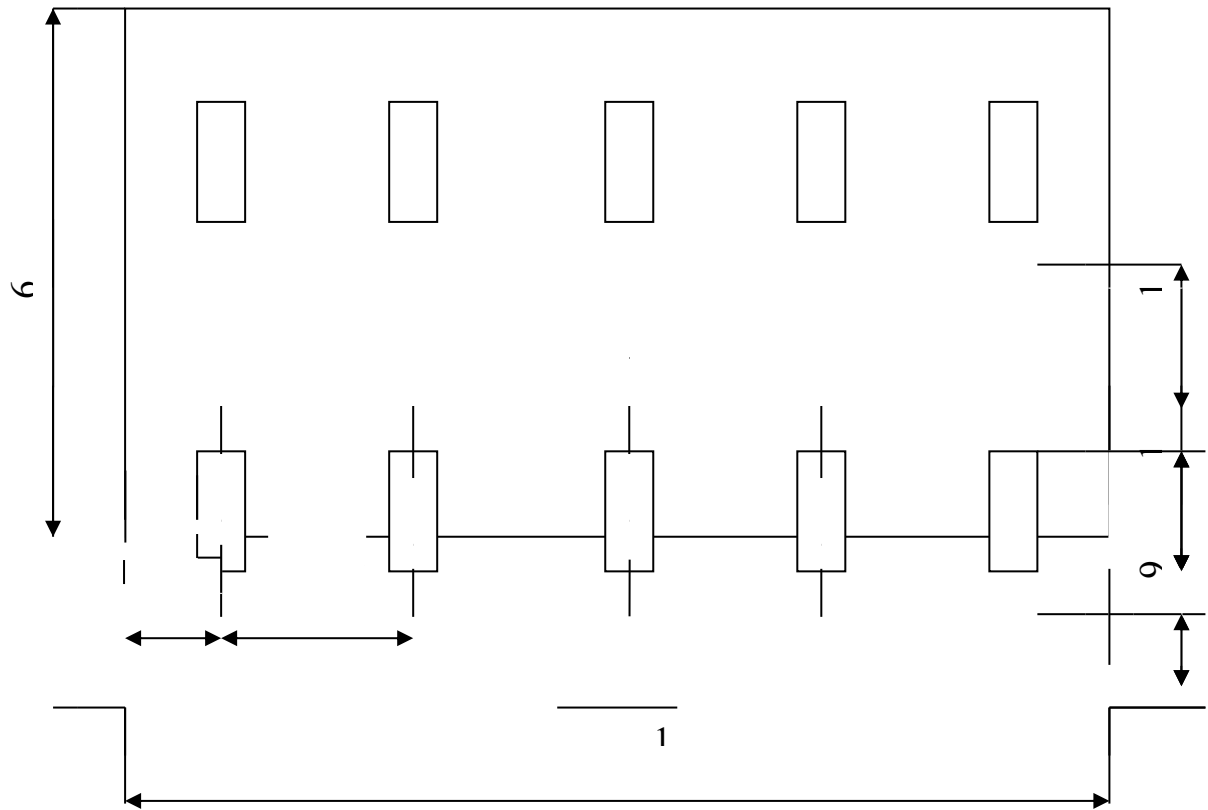


Рисунок 3.1 Розташування світильників у плані

Норма освітленості для загального освітлення за точністю виконуваної зорової роботи становить 200 лк (ІІІ розряд, підрозряд “в”).

Для розрахунку штучного освітлення застосовують в основному три методи: коефіцієнта використання світлового потоку, точковий та допустимої потужності.

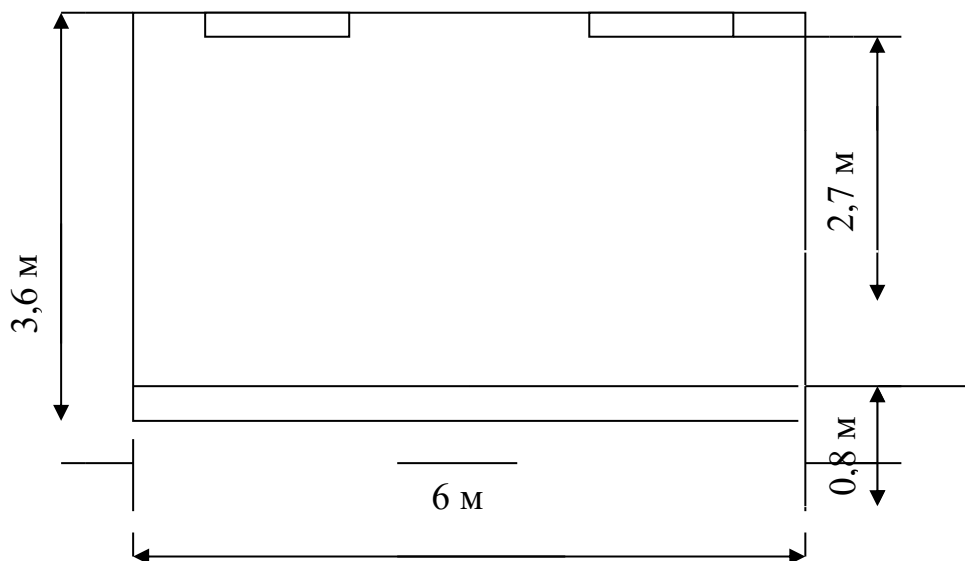


Рисунок 3.2 Поперечний розріз приміщення

Для розрахунку загального рівномірного освітлення при горизонтальній робочій поверхні основним є метод коефіцієнта використання світлового потоку, який враховує світловий потік, відбитий від стелі та стін. Відстань між світильниками у ряді $\lambda = 1,6$ м, висота підвісу $h = 2,7$ м, $0,5h = 0,5 \cdot 2,7 = 1,35$ м, тобто $\lambda > 0,5h$. За цієї умови для розрахунку горизонтальної освітленості можна застосувати метод коефіцієнта використання світлового потоку.

Світловий потік лампи розраховують так, лм:

$$\Phi_{\Lambda} = \frac{E_H S z k}{n_{\Lambda} \eta}, \quad (4.1)$$

де E_H – мінімальна нормована освітленість, лк; S – площа приміщення, м²; z – коефіцієнт мінімальної освітленості, що характеризує нерівномірність освітлення (відношення мінімальної освітленості до середньої горизонтальної), приймається $z = 1,1 \dots 1,2$; k – коефіцієнт запасу, який враховує старіння ламп, запиленість та забруднення світильників, його значення знаходиться в межах $1,3 \dots 1,7$; n_{Λ} – кількість ламп; η – коефіцієнт використання світлового потоку освітлювальної установки ($0,2 \dots 0,7$), який залежить від індексу приміщення i і коефіцієнтів відбиття стелі ρ_{Π} та стін ρ_C .

Коефіцієнт запасу k для приміщень громадських споруд при штучному освітленні для газорозрядних ламп дорівнює 1,5 (СНиП II-4-79).

Коефіцієнт мінімальної освітленості z для люмінесцентних ламп дорівнює 1,1.

Для зменшення контрастів яскравості в полі зору коефіцієнти відбиття поверхонь повинні мати наступні значення: стелі $0,7 - 0,75$; стін $0,4 - 0,5$.

Значить, вибираємо η для $\rho_{\Pi} = 0,7$, $\rho_C = 0,5$.

Індекс приміщення

$$i = \frac{S}{h(A \cdot B)} \quad (4.2)$$

де h – відстань від світильника до робочої поверхні, м.

$$i = (10 \cdot 6) / 2,7(10 + 6) = 1,38.$$

Коефіцієнт використання для світильників групи 8, прийнятих коефіцієнтів відбивання ρ і для індексу приміщення 1,38 становить $\eta = 0,47$.

Визначаємо фактичну мінімальну освітленість з (3.1), лк:

$$E_n = (Nn\Phi_{\text{л}}\eta)/(Skz) = (10 \cdot 2 \cdot 2340 \cdot 0,47)/(10 \cdot 6 \cdot 1,5 \cdot 1,1) = 222,18 \text{ (лм)}.$$

Розрахункова фактична освітленість у приміщенні задовольняє вимогам норм за “СНіП” (перевищення становить лише 11%).

4.2 Первинні засоби пожежегасіння

Підприємства та громадські споруди повинні бути забезпечені первинними засобами пожежегасіння, до яких належать пожежні стволи (водні та повітряно-пінні), внутрішні пожежні водопроводи (крани), вогнегасники (хімічно-пінні, газові, порошкові), бочки з водою, лопати, відра, сухий пісок, азбестові ковдри, інструмент та пристрої для розбирання будівельних конструкцій під час гасіння (багри, лопати, сокири та ін.).

Первинні засоби пожежегасіння призначені для ліквідації невеликих осередків пожеж, а також для гасіння пожеж на початковій стадії їхнього розвитку силами персоналу об’єктів до прибуття штатних підрозділів пожежної охорони.

Залежно від вогнегасної речовини вогнегасники підрозділяють на: водяні (із зарядом води чи води з добавками); пінні (хімічно-пінні, повітряно-пінні); порошкові; газові (вуглекислотні, хладонові); комбіновані (піна, порошок).

Вогнегасний ефект води полягає в змочуванні поверхонь, зволоженні та охолодженні речовин, що горять, механічному збиванні полум’я струменем води.

Водою не можна гасити рідкі горючі речовини, електроустановки, що перебувають під напругою, лужні метали.

Хімічно-пінні вогнегасники призначені для гасіння легкозаймистих та горючих речовин та матеріалів.

Повітряно-пінні вогнегасники (ВПП-5, ВПП-10, ВПП-100) мають ту ж область застосування, що й хімічно-пінні. На відміну від хімічної, повітряно-механічна піна не викликає корозію, більш екологічна, однак має меншу стійкість (швидко розкладається).

Пінні вогнегасники (хімічно-пінні та повітряно-пінні) не можна застосовувати для гасіння електроустановок, що знаходяться під напругою, оскільки до складу піни входить вода.

Хладонові вогнегасники (ВАХ, ВХ-3, ВВБ-3А, ВХ-7) призначені для гасіння електроустановок, що знаходяться під напругою 380 В, різноманітних горючих твердих та рідких речовин, за винятком лужних та лужноземельних металів та їх карбідів, а також речовин, що здатні горіти без доступу повітря.

Порошкові вогнегасники (ВП-1, ВП-2, ВП-5, ВП-10, ВП-100) є універсальними і характеризуються широким діапазоном застосування. Однак для гасіння пожежі в лабораторії з точними приладами краще застосовувати інші, так як порошок може зіпсувати прилади, потрапивши в них.

Вуглекислотні вогнегасники (ВВ-2, ВВ-5, ВВ-8, ВВ-25, ВВ-80) застосовуються для гасіння легкозаймистих та горючих рідин, твердих горючих речовин та матеріалів, електропроводок, що знаходяться під напругою до 1000 В. Тому вони підходять для даних умов.

Пожежу найлегше ліквідувати в початковій стадії. Успішна ліквідація займань можлива тільки в результаті чітких та швидких дій. Для цього треба знати будову, принцип дії вогнегасників та вміти ними користуватися.

Газові вогнегасники ВВ-2, ВВ-5, ВВ-8 та інші (цифра вказує на місткість балона в літрах) наповнюють зрідженим діоксидом вуглецю до робочого тиску 6 МПа. За принципом будови та дії всі вогнегасники аналогічні.

Вогнегасник ВВ-2 складається зі сталевого суцільнотягнутого балона, в горловину якого на конусній різбі ввернуто латунний голковий вентиль з сифонною трубкою. На вихідний штуцер вентиля нагвинчено раструбовий снігоутворювач (у переносних вогнегасників він закріплений на кінці гнучкого броньованого шлангу). Щоб привести вогнегасник в дію, потрібно, тримаючи його за рукоятку, спрямувати раструбовий снігоутворювач на вогнище пожежі та відкрити вентиль, обертаючи маховичок проти годинникової стрілки (в останніх моделях цього вогнегасника замість вентиля з маховичком встановлюється поворотна рукоятка). Виходячи з

вентиля, рідкий діоксид вуглецю переохолоджується та заморожується в снігоутворювачі у вигляді снігоподібних пластівців. Довжина струменя – біля 2 м, час дії – 30 с, пластівці в зоні горіння відразу перетворюються на газ. Під час роботи вогнегасника не можна торкатись снігоутворювача, оскільки температура його знижується до 70 °С.

Вогнегасна дія діоксиду вуглецю ґрунтується на розбавленні ним повітря та зниженні концентрації кисню в зоні горіння, що призводить до припинення процесу горіння. Один кілограм зрідженого діоксиду вуглецю утворює 0,5 м³ газу. При його випаровуванні відбирається теплота від речовини, що горить, знижується температура цієї речовини і це сприяє припиненню горіння.

Газові вогнегасники застосовуються для гасіння легкозаймистих та горючих рідин, твердих речовин, електроустановок, що перебувають під напругою. Ними не можна гасити речовини, котрі можуть горіти без доступу повітря (терміт тощо).

Вуглекислотно-брометилові вогнегасники ВВБ-3 та ВВБ-7 застосовуються для гасіння всіх первинних пожеж (крім горіння лужноземельних металів) при температурі навколишнього середовища від -60 до +60 °С.

За будовою та принципом дії ці вогнегасники подібні до газових, але мають тонкостінні сталеві балони та замість снігоутворювача – пряму насадку. Вогнегасники заряджають бромистим етилом : ВВ3 – 3,4 кг, ВВ7 – 7,76 кг і діоксидом вуглецю відповідно 0,1 кг та 0,24 кг.

Для викиду заряду з балона при відкритому вентилі у вогнегасник накачують повітря під тиском 0,86 МПа при температурі 20 °С.

Вогнегасна дія бромистого етилу (галогенового вуглеводню) заснована на гальмуванні хімічних реакцій горіння, тому вони часто називаються антикаталізаторами, або інгібіторами горіння. Бромистий етил – рідина, що легко випаровується, має змочувальні властивості та добре гасить горючі матеріали.

Рекомендується застосовувати в закритих приміщеннях. Для гасіння горючих матеріалів достатня концентрація брометилу 4,6 і 4,8 %.

Порошкові вогнегасники застосовуються для гасіння лужних металів, двигунів внутрішнього згорання, електроустановок. Для викиду порошкового заряду через конічну або циліндричну насадку у вогнегаснику є балончик із стисненим повітрям або зрідженим діоксидом вуглецю. Для запобігання зволоженню та змочуванню сухого порошку у вогнегасниках насадку герметично закривають спеціальною пробкою.

Для приведення вогнегасника в дію виймають пробку з насадки, потім вогнегасник спрямовують на вогнище, розгерметизовується газовий балончик шляхом відкривання вентиля або протикання герметизуючої перетинки. Газ потрапляє в корпус вогнегасника, створює тиск та виштовхує порошок, який містить 96,5 % кальцинованої соди, 1 % графіту, 1 % стеарату заліза, 1 % стеарату алюмінію, 0,5 % стеаринової кислоти. Порошок являє собою білі аморфні дрібнодисперсні частинки, що плавляться під дією тепла, та виділяє діоксид вуглецю. Розплавлений порошок сполучається, покриває речовину, що горить, плівкою низької теплопровідності, заважаючи поширенню вогню, ізолює речовину, що горить, гальмує реакцію горіння як інгібітор (антикаталізатор).

4.3 Кондиціонування повітря в лабораторії

Кондиціонування повітря застосовується для створення найбільш оптимальних умов і автоматичного забезпечення в закритих приміщеннях та спорудах наступних якостей повітря робочої зони: температури, відносної вологості, чистоти, газового та іонного складу, швидкості руху повітря.

Комплекс технічних засобів, що забезпечує бажану обробку повітря (фільтрацію, підігрів, охолодження, підсушування та зволоження), транспортування і розподілення його у приміщенні, влаштування засобів ізоляції шуму, що виникає під час роботи обладнання, джерела тепло- та холодопостачання, засоби автоматичного регулювання, контролю і керування, а також допоміжне обладнання складають систему кондиціонування повітря.

Кондиціонування повітря має суттєве значення для безпеки

життєдіяльності людини, а також для забезпечення режимів технологічних процесів, під час яких не допускається коливання температури та вологості. Системи кондиціонування поділяються на: місцеві (обслуговують одне приміщення); центральні (обслуговують кілька приміщень або будівлю в цілому). В лабораторіях використовують місцеві кондиціонери, так як вони мають меншу продуктивність, яка підходить для їх роботи у відносно невеликих приміщеннях.

У приміщеннях, де застосовують місцеві системи кондиціонування, агрегат для кондиціонування повітря розташовують безпосередньо в приміщенні, що обслуговується або в безпосередній близькості від нього. В центральних системах кондиціонування повітря обробляється в кондиціонерах, які виносяться в окремі приміщення.

Кондиціонування повітря може бути повним і неповним.

Повне кондиціонування повітря – це коли забезпечується сталість температури, відносної вологості, рухомості, чистоти та інших якостей повітря. Неповне кондиціонування забезпечує лише частину з цих параметрів.

У виробничих приміщеннях, де люди перебувають певний час, використовується кондиціонування повітря з врахуванням санітарно-гігієнічних норм. Це стосується і лабораторій.

Кондиціонери слід вибирати, виходячи з розрахункової кількості повітря (продуктивності) по даним заводу-виробника (СНіП 2.04. 05–91). Продуктивність по повітрю (повітрообмін) систем кондиціонування – це кількість повітря, котре подається в приміщення. Її слід визначати розрахунком і приймати більше значення.

Розрахунок повітря можна робити по: а) залишковим тепловиділенням; б) залишковим вологовиділенням; в) виділенням парів та газів шкідливих речовин. Це найбільш точні та поширені розрахунки.

Оскільки більшу кількість повітря можна в нашому випадку отримати по залишкам теплоти, то скористаємося формулою:

$$L_T = \frac{Q_{ИЗБ}}{\gamma c (t - t_{ПР})}, \quad (4.3)$$

де $Q_{изб}$ – залишкові теплопритоки в приміщенні, ккал/год; γ – питома вага приточного повітря, (кг/м³); c – питома теплоємність повітря, ккал/кг К); t – температура повітря в приміщенні, °С; $t_{пр}$ – температура приточного повітря.

Температура в приміщенні $t = 25$ °С. Питома вага приточного повітря при даній температурі $\gamma = 1,2$ кг/м³. Питома теплоємність повітря $c = 1$ кДж/(кг*°С). Температура приточного повітря $t_{пр} = 20$ °С (норми ГОСТ).

Повна кількість тепла в приміщенні:

$$Q_{изб} = Q_1 + Q_2 + Q_3 + Q_4, \quad (4.4)$$

де Q_1 – кількість тепла, що потрапляє від сонячної радіації; Q_2 – кількість тепла, що потрапляє через непрозорі поверхні; Q_3 – кількість тепла від нагрітих матеріалів; Q_4 – тепловиділення від людей.

Q_1 для вікон з подвійним зашкленням та дерев'яним переплетенням, що виходять на північно-західну сторону з широтою 45° становить 270 кДж/(м³год).

Q_2 по тим же рекомендаціям становить 120 кДж/(м³год).

Q_3 приймаємо 95 кДж/(м³год).

При температурі повітря в приміщенні 25 °С повне тепловиділення від людини, яка виконує легку працю, становить 520 кДж/год. Якщо врахувати, що людей в приміщенні 20, то Q_4 буде дорівнювати 1400 кДж/год.

Надлишок тепла в приміщенні: $Q_{изб} = 270 + 120 + 95 + 1400 = 1885$ кДж/год.

Отже, кількість повітря: $L_T = 1885 / 1,2 \cdot (25 - 20) = 197,05$ м³/год.

Кратність повітрообміну n :

$$n = \frac{L}{V}, \quad (4.5)$$

де V – об'єм приміщення: $n = 197,05 / 216 = 0,9$ год⁻¹.

По каталогах вибираємо для лабораторії місцевий кондиціонер з продуктивністю ≈ 200 м³/год.

4.4 Безпека в надзвичайних ситуаціях

Надзвичайна ситуація в умовах науково-дослідної лабораторії або офісного приміщення, де здійснюється розробка, тестування програмного забезпечення та проєктування автоматизованих систем, може бути спричинена пожежею, раптовим відключенням електроенергії, коротким замиканням апаратури, а також зовнішніми факторами техногенного, природного чи військового характеру. Головною метою заходів безпеки у таких випадках є збереження життя та здоров'я персоналу, зведення до мінімуму матеріальних збитків і запобігання втраті критично важливої корпоративної інформації. Для забезпечення готовності до дій у надзвичайних ситуаціях на підприємстві розробляється та затверджується план ліквідації аварій, а також схеми евакуації, які розміщуються на видних місцях у коридорах та робочих приміщеннях.

У разі виникнення пожежі або задимлення першочерговою дією є негайне припинення всіх робіт та знеструмлення комп'ютерної техніки і серверного обладнання шляхом вимкнення загального рубильника або щитка. Персонал зобов'язаний негайно повідомити про інцидент пожежну охорону, керівництво підприємства та розпочати евакуацію згідно із затвердженим планом. За умови відсутності прямої загрози життю, працівники, які пройшли відповідний інструктаж, можуть приступити до гасіння осередку займання за допомогою первинних засобів пожежегасіння. Як було обґрунтовано раніше, для гасіння комп'ютерної техніки та електроустановок під напругою необхідно застосовувати виключно вуглекислотні вогнегасники, щоб уникнути ураження струмом та незворотного пошкодження електронних плат.

Оскільки робота ІТ-спеціалістів та інженерів з автоматизації нерозривно пов'язана з інтенсивним використанням електрообладнання, високим є ризик виникнення аварійних ситуацій через коротке замикання або пробій ізоляції. У випадку ураження працівника електричним струмом необхідно негайно звільнити потерпілого від дії струму, вимкнувши електроустановку найближчим вимикачем. Якщо швидко знеструмити лінію неможливо, слід відтягнути потерпілого за одяг (якщо він сухий), використовуючи діелектричні

рукавиці або інші ізолюючі матеріали. Після цього необхідно викликати швидку медичну допомогу та до її прибуття надати першу долікарську допомогу: забезпечити приплив свіжого повітря, а за відсутності дихання — розпочати проведення штучної вентиляції легень та непрямого масажу серця.

Для мінімізації наслідків раптового зникнення напруги в електромережі, що може призвести до втрати незбережених програмних кодів, пошкодження баз даних та збою в роботі серверів, уся критично важлива обчислювальна техніка повинна бути підключена через джерела безперебійного живлення (UPS). У разі аварійного відключення електроенергії ці пристрої подають звуковий сигнал і забезпечують резервне живлення протягом часу, достатнього для коректного збереження робочих проектів, штатного завершення роботи операційних систем та безпечного вимкнення обладнання. Крім того, на випадок відключення основного освітлення, у приміщеннях та на шляхах евакуації повинна автоматично вмикатися система аварійного освітлення, яка живиться від незалежних автономних джерел.

Окрему увагу слід приділити діям персоналу при отриманні сигналів цивільного захисту, зокрема сигналу «Повітряна тривога» або «Увага всім!». При надходженні такого сигналу через засоби масової інформації або внутрішню систему оповіщення, працівники зобов'язані негайно зберегти дані, заблокувати робочі станції, відключити електроприлади від мережі, зачинити вікна та організовано, без паніки покинути приміщення, слідуючи за вказівниками до найближчого захисної споруди (укриття). Повернення на робочі місця та відновлення проектної діяльності дозволяється виключно після офіційного повідомлення про відбій тривоги та отримання відповідного дозволу від керівника підрозділу або особи, відповідальної за цивільний захист на підприємстві.

Висновки по розділу 4

У четвертому розділі було проведено комплексний аналіз умов охорони праці та безпеки життєдіяльності в робочому приміщенні лабораторії, що є критично важливим етапом для забезпечення безпечної роботи персоналу та

надійного функціонування апаратного і програмного забезпечення.

Зокрема, здійснено розрахунок системи штучного загального рівномірного освітлення із застосуванням методу коефіцієнта використання світлового потоку. Встановлено, що використання люмінесцентних ламп типу ЛД-40 у світильниках ЛПО-01 забезпечує розрахункову фактичну освітленість на рівні 222,18 лк. Цей показник повністю задовольняє нормативні вимоги (СНіП) для виконання точної зорової роботи, які вимагають 200 лк (перевищення норми становить прийнятні 11%), що гарантує комфортні умови для зору працівників та запобігає виробничому травматизму.

Під час опрацювання питань пожежної безпеки було проаналізовано первинні засоби пожежегасіння. Обґрунтовано, що для умов лабораторії з точними електронними приладами та електрообладнанням застосування порошкових вогнегасників є недоцільним, оскільки порошок може потрапити в обладнання та безповоротно зіпсувати його. Натомість найоптимальнішим вибором визначено вуглекислотні вогнегасники (типу ВВ-2, ВВ-5 тощо), які ефективно справляються із загорянням електропроводок під напругою і не залишають слідів на апаратурі.

Також для забезпечення оптимального мікроклімату та дотримання санітарно-гігієнічних норм було розраховано необхідні параметри системи кондиціонування повітря. На основі детального розрахунку залишкових тепловиділень у приміщенні (від сонячної радіації, поверхонь, працюючого обладнання та людей), загальний надлишок тепла склав 1885 кДж/год. Відповідно до цього, необхідний повітрообмін становить 197,05 м³/год. Для забезпечення цих потреб було обрано місцевий кондиціонер із продуктивністю 200 м³/год, що повністю перекриває розрахункову потребу.

Таким чином, розраховані системи штучного освітлення, підібрані установки кондиціонування повітря та обрані первинні засоби пожежегасіння повністю відповідають чинним нормам безпеки та охорони здоров'я. Їх впровадження гарантує підтримання необхідних мікрокліматичних і безпекових умов для безперебійного функціонування обчислювальної техніки та збереження здоров'я персоналу лабораторії.

ЗАГАЛЬНИЙ ВИСНОВОК

Підсумовуючи та глибоко узагальнюючи всю наведену вище інформацію щодо якісних трансформацій у процесі конструювання програмного забезпечення, можна чітко виокремити низку фундаментальних тенденцій, які визначають вектор автоматизації цієї галузі. Насамперед, спостерігається масштабна уніфікація технологічних прийомів та методів проектування. Цей процес супроводжується активною популяризацією найкращих світових практик, які сьогодні стандартизуються та пропонуються розробникам у зручному форматі готових архітектурних шаблонів, загальновизнаних методик і потужних програмних бібліотек. Тісно пов'язаною з цим є концепція контейнеризації та модульності розробки. Сучасна інженерія йде шляхом глибокої інкапсуляції, коли складні внутрішні деталі реалізації приховуються всередині окремих, незалежних компонентів. Ці блоки згодом легко інтегруються в єдину систему за принципом конструктора, що суттєво пришвидшує випуск готового продукту.

Паралельно з цим відбувається глобальний перехід до сервісно-орієнтованого підходу та масова міграція робочих процесів у хмарні середовища. Традиційні десктопні програмні продукти, локальні засоби розробки та інструменти розгортання все частіше поступаються місцем гнучким хмарним моделям надання послуг, таким як SaaS, PaaS, IaaS та DaaS. Ця технологічна міграція створює ідеальні умови для реалізації наступної важливої тенденції — наскрізної інтеграції процесів та даних. Сьогодні життєвий цикл програмного продукту спирається на використання комплексних баз знань, які агрегують вичерпну інформацію на всіх без винятку етапах розробки. Ці бази безпосередньо і безперервно взаємодіють із різноманітними прикладними інструментами, серед яких спеціалізовані системи управління проектами, баг-трекери та інтегровані середовища розробки (IDE).

Окремої уваги заслуговує еволюція самого інструментарію написання коду. Чітко простежується тренд на популяризацію декларативних підходів до

програмування, а також стрімкий розвиток мов, спеціально орієнтованих на складні математичні обчислення та статистичну обробку великих масивів даних (Big Data). Сучасні мови програмування стають дедалі більш мультипарадигмальними, пропонуючи розробникам безпрецедентно високий рівень абстракції. У перспективі всі ці інновації ведуть до глобальної мети — поступового пакування всіх розрізнених етапів створення програмного забезпечення в єдину, монолітну сервісну модель. Така парадигма передбачає можливість автоматичної генерації кінцевого продукту за принципом «чорного ящика», де від користувача вимагається лише введення базових параметрів та вимог.

Такі глибокі технологічні зсуви неминуче провокують і трансформацію людського фактора в ІТ-галузі. Відбувається поступове стирання меж та об'єднання класичних професійних ролей. У сучасних реаліях часто виникають ситуації, коли функції замовника, кінцевого користувача, менеджера проєкту, системного аналітика, програміста та інженера з тестування акумулюються в особі одного спеціаліста. З іншого боку, всередині самої професійної ІТ-спільноти поглиблюється диференціація фахівців залежно від технологічного стеку, який вони використовують. Водночас середовище звичайних користувачів також сегментується, але вже за критерієм рівня обізнаності та практичних навичок володіння популярними цифровими інструментами.

Спираючись на проведений ґрунтовний аналітичний огляд технологій, які є рушійною силою перелічених вище тенденцій, можна сформулювати базову, інтегральну характеристику всього сучасного інструментарію розробки програмного забезпечення. Її сутність полягає у безперервному та невідворотному підвищенні рівня абстракції. У рамках цієї концепції глибокі прикладні та апаратні технічні особливості — наприклад, алгоритми прямого доступу до внутрішньої пам'яті пристрою, управління потоками чи низькорівнева оптимізація — повністю приховуються від розробника. Виконання цих складних функціональних операцій автоматично забезпечується самими засобами розробки (компіляторами, віртуальними машинами, фреймворками) і фактично виводиться із зони безпосередньої відповідальності

людини.

З одного боку, подібна тотальна автоматизація надзвичайно спрощує та прискорює рутинний процес створення програмного забезпечення, відкриваючи двері в індустрію навіть для людей без профільної інженерної освіти. Проте, з іншого боку, вона парадоксальним чином висуває значно жорсткіші вимоги до фундаментальних компетенцій справжніх ІТ-фахівців. Підсумовуючи, варто наголосити: незважаючи на те, що сучасні методичні та прикладні інструменти здатні автоматизувати практично всі кроки життєвого циклу проєкту (від початкового аналізу вимог до фінального впровадження і післягарантійного супроводу), цей факт жодним чином не звільняє професійного розробника від відповідальності. Справжній інженер повинен володіти глибокими практичними та теоретичними знаннями щодо кожного автоматизованого етапу і, у разі виникнення критичної необхідності, бути здатним виконати всі ці процеси самостійно, у повністю ручному режимі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Барковський В. В. Застосування удосконалень метода найменших квадратів (МНК) до визначення закономірностей в сучасних телекомунікаційних мережах / В. В. Барковський, Т. В. Майсак // XII Міжнар. наук. конф. ім. академіка М. Кравчука : матеріали конф. (Київ, 15–17 трав. 2008 р.). – К. : Задруга, 2008. – С. 21.
2. Беркман Л. Н. Застосування вдосконалених модифікацій методу найменших квадратів до оптимізації телекомунікаційних мереж / Л. Н. Беркман, Т. В. Майсак // Зв'язок. – 2008. – № 7–8. – С. 46–52.
3. Беркман Л. Н. Рівні конвергенції і їх реалізація – ситуація у світі і в Україні / Л. Н. Беркман, Т. В. Майсак // Зв'язок. – 2009. – № 1 (85). – С. 19–22.
4. Волочій Б. Ю. Надійнісна модель відмовостійкої багатопроцесорної систем з відновленням працездатності програмного забезпечення / Б. Ю. Волочій, Л. Д. Озірковський, О. В. Муляк та ін. // Вісник НТУУ «КПІ». Серія Радіотехніка. Радіоапаратобудування. – 2013. – № 54. – С. 33–43.
5. Вишнівський В. В. Основи надійності та діагностики телекомунікаційних і радіотехнічних систем : конспект лекцій / В. В. Вишнівський. – К. : ННІТІ ДУТ, 2015. – 142 с.
6. Майсак Т. В. Оптимізація параметрів телекомунікаційних систем та мереж на базі економетричних методів : автореф. дис. канд. тех. наук : 05.12.02 / Т. В. Майсак ; Держ. ун-т інформ.-комунік. технологій. – К., 2009. – 16 с.
7. Математичні моделі та методи аналізу надійності радіоелектронних, електротехнічних та програмних систем : монографія / Ю. Я. Бобало, Б. Ю. Волочій, О. Ю. Лозинський та ін. – Львів : Вид-во Львівської політехніки, 2013. – 300 с.
8. Муляк О. В. Вплив моделей надійності програмних засобів на показники надійності програмно-апаратних систем / О. В. Муляк, В. С. Яковина, Б. Ю. Волочій // Східно-Європейський журнал передових технологій. – 2015. – Т. 4, № 9(76). – С. 53–57.
9. Нитребич О. О. Моделі та методи оцінювання надійності програмного

- забезпечення з урахуванням його архітектури : дис. канд. техн. наук : 01.05.03 / О. О. Нитребич ; Нац. ун-т "Львівська політехніка". – Львів, 2015. – 153 с.
10. Притула А. В. Практично-орієнтовані методи викладання в галузі вбудованих систем / А. В. Притула, А. В. Пархоменко, Г. В. Табунщик // Сучасні проблеми і досягнення в галузі радіотехніки, телекомунікацій та інформ. технологій : зб. тез доп. VII міжнар. наук.-практ. конф. (Запоріжжя, 17–19 верес. 2014 р.). – Запоріжжя : ЗНТУ, 2014. – С. 216–217.
 11. Сенів М. М. Система оцінювання та прогнозування надійності програмного забезпечення / М. М. Сенів, Д. В. Федасюк, Ю. І. Парфенюк та ін. // Відбір і обробка інформації. – 2010. – № 33 (109). – С. 123–129.
 12. Табунщик Г. В. Віддалений та віртуальний інструментарій в інжинірингу : монографія / Г. В. Табунщик, Т. І. Каплієнко. – Запоріжжя : ЗНТУ, 2015. – С. 154–204.
 13. Табунщик Г. В. Інженерія якості програмного забезпечення : навч. посіб. / Г. В. Табунщик, Р. К. Кудерметов, Т. І. Брагіна. – Запоріжжя : Дике поле, 2013. – 173 с.
 14. Тимошенко Ю. О. Узагальнена модель негомогенного пуассонівського процесу для оцінювання надійності програмного забезпечення / Ю. О. Тимошенко, М. В. Дідковська // Проблеми програмування. – 2004. – № 2–3. – С. 480–489.
 15. Федасюк Д. В. Метод побудови сценаріїв тестування програмного забезпечення на основі аналізу його змінних / Д. В. Федасюк, В. С. Яковина, П. В. Сердюк, О. О. Нитребич // Інформаційні технології та комп'ютерна інженерія. – 2014. – № 2 (30). – С. 50–58.
 16. Франчук В. М. Web-орієнтовані навчальні комп'ютерні системи / В. М. Франчук // Актуальні проблеми методології та методики навчання фізико-математичних дисциплін : матеріали Міжнар. наук. конф. (Київ, 18–19 січ. 2013 р.). – К. : Вид-во НПУ ім. М. П. Драгоманова, 2013. – С. 124–125.
 17. Франчук В. М. Вибір системи управління вмістом сайту / В. М. Франчук, О. В. Галицький // Науковий часопис НПУ ім. М. П. Драгоманова. Серія 2. Комп'ютерно-орієнтовані системи навчання. – 2014. – № 14(21). – С. 19–28.

18. Франчук В. М. Управління інформаційними ресурсами засобами web-орієнтованих комп'ютерних систем в освітніх закладах / В. М. Франчук, О. В. Галицький // Тези доповідей II Міжнар. наук.-практ. конф. «Інформ. технології в освіті, науці і техніці» (ІТОНТ-2014) : Черкаси, 24–26 квіт. 2014 р. – Черкаси : ЧДТУ, 2014. – Т. 2. – С. 28–29.
19. Чабанюк Я. М. Побудова і дослідження моделі надійності програмного забезпечення з індексом величини проекту / Я. М. Чабанюк, В. С. Яковина, Д. В. Федасюк та ін. // Інженерія програмного забезпечення. – 2010. – № 1. – С. 24–29.
20. Яковина В. С. Методи та засоби аналізу надійності функціонування програмного забезпечення з урахуванням етапів життєвого циклу : автореф. дис. доктора тех. наук : 01.05.03 / В. С. Яковина ; Нац. ун-т "Львівська політехніка". – Львів, 2015. – 325 с.
21. Яковина В. С. Огляд основних підходів до аналізу надійності програмного забезпечення / В. С. Яковина, В. Смірнов // Вісник Нац. ун-ту "Львівська політехніка". Комп'ютерні науки та інформ. технології. – 2011. – № 719. – С. 278–282.
22. Arras P. E-learning environment for the remote study in material properties courses / P. Arras, Y. Kolot, T. Galyna, T. Kozik // International Journal of Computing. – 2014. – Vol. 12 (3). – P. 233–238.
23. Henke K. Using Interactive Hybrid Online Labs for Rapid Prototyping of Digital Systems / K. Henke, G. Tabushchyk, H-D Wuttke et al. // REV2014 Conference : proceedings (Porto, 26–28 Feb. 2014). – Porto, 2014. – P. 48–59.
24. Koziolk H. Operational Profiles for Software Reliability / H. Koziolk // Dependability Engineering, volume 2 of Trustworthy Software Systems. – 2006. – P. 119–142.
25. Fedasyuk D. V. Variable state-based software usage-model based on its variables / D. V. Fedasyuk, V. Yakovyna, P. Serdyuk, O. Nytrebych // Econtechmod. – 2014. – Vol. 3, № 2. – P. 15–20.
26. Mirmomeni M. Fuzzy descriptor systems and spectral analysis for chaotic time series prediction / M. Mirmomeni, C. Lucas, M. Shafiee et al. // Neural Computing

and Applications. – 2009. – Vol. 18. – P. 991–1004.

27. Salmeron M. Parallel Computation of an Adaptive Optimal RBF Network Predictor / M. Salmeron, J. Ortega, C. G. Puntonet, M. Damas // LNCS. – 2003. – Vol. 2687. – P. 425–432.

28. Yakovyna V. S. Influence of the activation function of RBF neural network on software reliability time series prediction / V. S. Yakovyna, V. Ketsman // VIIIth Intern. Conf. on Perspective Technologies and Methods in MEMS Design MEMSTECH'2012 : proceedings (Polyana, 18–21 Apr. 2012). – Polyana, 2012. – P. 91–92.

29. Яковина В. С. Основи теорії надійності програмних систем / В. С. Яковина, М. М. Сенів. – Львів : Вид-во Львівської політехніки, 2020. – 248 с.

30. Федасюк Д. В. Методи та засоби автоматизованого тестування програмного забезпечення / Д. В. Федасюк, В. С. Яковина // Вісник Нац. ун-ту «Львівська політехніка». Серія : Комп'ютерні науки та інформ. технології. – Львів : Вид-во Львівської політехніки, 2017. – № 864. – С. 112–119.

31. Харченко В. С. Верифікація та оцінювання безпеки інформаційно-керуючих систем / В. С. Харченко, О. А. Ілляшенко // Радіoeлектронні і комп'ютерні системи. – Харків : НАКУ «ХАІ», 2018. – № 3(87). – С. 54–62.

32. Голего В. М. Основи технічної діагностики та надійності радіoeлектронних об'єктів / В. М. Голего. – К. : НАУ, 2015. – 320 с.

33. Романюк О. Н. Інструментальні засоби опрацювання даних про відмови програмних систем на етапі тестування / О. Н. Романюк, Т. О. Савчук // Інформаційні технології та комп'ютерна інженерія. – Вінниця : ВНТУ, 2021. – № 2(51). – С. 76–84.

34. Волошин О. Ф. Оцінювання надійності програмного забезпечення на основі марковських моделей та компонентного підходу / О. Ф. Волошин, І. В. Козак // Проблеми програмування. – К. : Ін-т програмних систем НАН України, 2019. – № 1. – С. 22–31.

35. Стрілкова Т. В. Моделювання та оцінка надійності сучасних телекомунікаційних мереж / Т. В. Стрілкова, О. М. Ковальчук // Телекомунікаційні та інформаційні технології. – К. : ДУТ, 2018. – № 3. – С. 15–

22.

36. Мельник А. О. Автоматизація промислових процесів на базі програмованих логічних контролерів / А. О. Мельник, О. І. Мороз // Автоматизація виробничих процесів у машинобудуванні та приладобудуванні. – Львів : Вид-во Львівської політехніки, 2019. – Вип. 53. – С. 45–51.

37. Коваленко І. М. Застосування математичних пакетів для розрахунку показників надійності інфокомунікаційних систем / І. М. Коваленко, С. В. Сидоренко // Зв'язок. – К. : ДУТ, 2016. – № 4. – С. 38–43.

38. Петренко О. І. Метод нелінійного регресійного аналізу для прогнозування кількості помилок у програмних системах / О. І. Петренко, В. О. Мухін // Системні дослідження та інформаційні технології. – К. : ННК «ІПСА» НТУУ «КПІ», 2020. – № 4. – С. 89–98.

39. Катренко А. В. Управління ІТ-проектами : навч. посіб. / А. В. Катренко. – Львів : Новий Світ-2000, 2011. – 540 с.

40. Кобиляк Г. І. Управління проектами : навч. посіб. / Г. І. Кобиляк, О. І. Кобиляк. – К. : КНЕУ, 2018. – 320 с.

41. Кравченко О. П. Інженерія програмного забезпечення : курс лекцій / О. П. Кравченко. – Черкаси : ЧДТУ, 2016. – 180 с.

42. Саченко А. О. Системи штучного інтелекту : навч. посіб. / А. О. Саченко. – Тернопіль : Економічна думка, 2015. – 240 с.

43. Тесля Ю. М. Вступ до управління проектами : навч. посіб. / Ю. М. Тесля. – К. : Кондор, 2011. – 184 с.

44. Wysocki R. K. Effective Project Management: Traditional, Agile, Extreme / R. K. Wysocki. – 7th Edition. – Indianapolis : Wiley, 2014. – 768 p.

45. Kerzner H. Project Management: A Systems Approach to Planning, Scheduling, and Controlling / H. Kerzner. – 11th Edition. – Hoboken : Wiley, 2013. – 1296 p.

46. Martin R. C. Clean Architecture: A Craftsman's Guide to Software Structure and Design / R. C. Martin. – Boston : Prentice Hall, 2017. – 432 p.

47. Guide to the Project Management Body of Knowledge (PMBOK Guide). – 6th Edition. – Pennsylvania : Project Management Institute, 2017. – 756 p.

48. Sommerville I. Software Engineering / I. Sommerville. – 10th Edition. – Harlow :

Pearson Education Limited, 2016. – 816 p.

49. Pressman R. S. Software Engineering: A Practitioner's Approach / R. S. Pressman, B. R. Maxim. – 8th Edition. – New York : McGraw-Hill Education, 2015. – 960 p.

50. Schwaber K. The Scrum Guide / K. Schwaber, J. Sutherland. – [Electronic resource]. – URL: <https://scrumguides.org/scrum-guide.html>.