Вінницький національний технічний університет
Факультет інформаційних технологій та комп’ютерної інженерії
Кафедра комп’ютерних наук

Пояснювальна записка

до магістерської кваліфікаційної роботи

на тему: «ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ УПРАВЛІННЯ ІТ ПРОЕКТАМИ»

Виконав: студент 2 курсу,
групи 1КН-20м
спеціальності 122 «Комп'ютерні науки»
Колесников Ю.С.

Керівник: к.т.н., доц.кафедри КН
Барабан С. В.

Рецензент: д.т.н., проф Ліщинська Л.Б.

Вінниця
2021

ЗАТВЕРДЖУЮ
Завідувач кафедри____КН____
д.т.н., проф.. Яровий А.А.

_____
(підпис)
“____” _____2021 року

ЗАВДАННЯ

на магістерську кваліфікаційну роботу на здобуття кваліфікації магістра зі спеціальності: 122 – «Комп'ютерні науки»

08-22.МКР.008.20.539.ПЗ

Магістранта групи «1КН-20м» Колесникова Юрія Сергійовича

Тема магістерської кваліфікаційної роботи: «Інформаційна технологія управління ІТ проектами»

Вхідні дані: повний склад команди (Middle Java Back-end розробник; Junior Java + JavaScript FullStack розробник; Senior JS React Front-End розробник; Senior QA інженер; Бізнес аналітик; Middle DevOps інженер; Керівник проекту), тривалість тестування – 8 тижнів, покращення ефективності – не менше 10%.

Короткий зміст частин магістерської кваліфікаційної роботи:

1. Графічна: схематичне зображення Scrum процесів, Офіційний логотип Інституту управління проектами, класична візуалізація Канбан-дошки, етапи каскадної методології управління проектами Waterfall.
2. Текстова (пояснювальна записка): вступ, обгрунтування актальності теми дослідження, проектування інформаційної технології управління ІТ проектами, програмна реалізація та тестування інформаційної технології управління ІТ проектами, економічна частина, висновки, перелік використаних джерел, додатки.

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ МКР

№ етапу	Назва етапу	Термін виконання		Очікувані результати
		початок	кінець	
1	Обґрунтування актуальності теми дослідження			Аналіз предметної області та скріншоти аналогів
2	Проектування та реалізація інформаційної технології управління ІТ проектами			Архітектура додатку та готова методологія
3	Підготовка економічної частини			Економічне обґрунтування
4	Апробація результатів дослідження			тези доповідей
5	Оформлення пояснювальної записки, графічного матеріалу та презентації			Пояснювальна записка, графічний матеріал, презентація

Консультанти з окремих розділів магістерської кваліфікаційної роботи

1. Науковий керівник _____ к.т.н, доц., доц., кафедри КН
(підпис) наук. ступінь, вчене звання (посада)
“____” _____20___ р. _____С.В. Барабан_____
ініціали та прізвище

2. Економічна частина _____ канд. екон. наук, доц., доц. каф. ЕПВМ
(підпис) наук. ступінь, вчене звання (посада)
_____М. В. Бальзан_____
ініціали та прізвище
“____” _____20___ р.

Дата попереднього захисту роботи “____” _____20___ р.

Рецензент _____ _ д.т.н., проф.кафедри ПЗ_____
(підпис) наук. ступінь, вчене звання (посада)
_____Л.Б. Ліщинська_____
ініціали та прізвище

Завдання видав
науковий керівник _____ к.т.н, доц., доц., кафедри КН
(підпис) наук. ступінь, вчене звання (посада)
_____С.В. Барабан_____
ініціали та прізвище
“____” _____20___ р.

Завдання отримав магістрант _____ _____Ю.С. Колесников_____
(підпис) ініціали та прізвище
“____” _____20___ р.

АНОТАЦІЯ

У даній магістерській кваліфікаційній роботі був реалізован фреймворк для управління IT проектами. Проаналізовано та здійснено порівняння сучасних методологій управління IT проектами, шо використовуються в IT галузі при створенні програмних продуктів. На підставі цього було запропоновано інформаційну технологію, що дозволяє покращити ефективність управління командами розробників.

Запропоновано удосконалені принципи управління IT проектами та удосконалені івенти для управління командою.

При реалізації фреймворку використано сучасну Agile методологію Scrum, як основу для покращенної.

В роботі проаналізовано розроблену методологію та приведені результати досліджень.

ABSTRACT

In this master's qualification work, a framework for IT project management was implemented. The comparison of modern methodologies of IT project management, which are used in the IT field when creating software products, is analyzed and compared. Based on this, information technology was proposed to improve the management of development teams.

Improved principles of IT project management and improved team management events are proposed.

The implementation of the framework uses the modern Agile Scrum methodology as a basis for improvement.

The developed methodology is analyzed in the work and the results of researches are given.

ЗМІСТ

ВСТУП

Актуальність теми. Будь-яка IT компанія коли-небудь задавалися питанням: «Чому важливий менеджмент проектів?»

Якщо компанія ніколи раніше не працювала з менеджером проекту, може виникнути спокуса відмовитися від витрат і керувати проектом самостійно. Зрештою, управління проектами може бути серйозною фінансовою інвестицією, то чому б просто не зробити це самостійно? Але це було б помилкою.

Хороший менеджер проекту – це клей, який скріплює проект і забезпечує якість та досягнення цілей вчасно та в рамках бюджету. За даними Інституту управління проектами (PMI), організації, які недооцінюють управління проектами, повідомляють, що в середньому на 50% більше їхніх проектів зазнають невдачі. П'ятдесят відсотків.

Невдалі проекти можуть швидко зруйнувати ваші ініціативи та затримати або навіть запобігти зростанню бізнесу.

Управління проектом – це більше, ніж просто відстеження термінів і встановлення бюджету. Хороший менеджер проекту контролює проект від початку до кінця, гарантуючи, що ініціативи та цілі стратегічно узгоджені, проект має підтримку зацікавлених сторін і всі знаходяться на одній сторінці.

Головний інструмент в руках менеджера проекту – це методологія розробки програмного забеспечення.

Методологія розробки ПЗ - це система (фреймворк або методологія), яка визначає порядок виконання завдань, методи оцінки та контролю.

Підходи до розробки програмного забезпечення визначають успіх проекту, адже без правильно підібраною методології складно досягти стабільності в роботі продукту, безпеки і стійкості функціональних особливостей. Керівники проекту намагаються знайти оптимальний варіант з безлічі.

Нажаль, не існує ідеальної методології, яка б вирішувала усі задачі під час процесу розробки програмного забеспечення. Кожна має свою вузконаправленність, і може підходити для одного проекту, а для іншого – ні.

Тому створення універсального фреймворку сприятиме поліпшеню налаштування процесів при розробці ПЗ.

Зв'язок роботи з науковими програмами, планами, темами. Магістерська робота виконана відповідно до напрямку наукових досліджень кафедри комп'ютерних наук Вінницького національного технічного університету 22 К1 «Розробка спеціалізованих засобів штучного інтелекту на основі інтелектуального аналізу даних та машинного навчання».

Мета та завдання дослідження. Метою даної магістерської дипломної роботи є підвищення ефективності управління командами розробників сучасних IT проектів не менше ніж на 10%.

Для досягнення поставленної мети необхідно розв'язати такі завдання:

1. здійснити аналіз предметної області та об'єкту проектування;
2. здійснити аналіз існуючих методологій та їх характеристик;
3. пояснити особливості проектування методологій управління проектами;
4. виконати реалізацію методології;
5. провести тестування та проаналізувати отримані результати.

Об'єкт дослідження – процес управління IT проектами з використанням Agile методологій.

Предмет дослідження є сучасні Agile методології управління проектами розробки програмного забезпечення.

Методи дослідження. У роботі використані наступні методи наукових досліджень: аналізу структури інформаційної технології, теорії методів керування командами для інформаційної технології управління IT проектами, методи експерименту для проведення тестування та аналізу результатів.

Наукова новизна одержаних результатів полягає в наступному:

Розроблено інформаційну технологію управління IT проектами, що забеспечує можливість більш ефективно керувати сучасними командами розробників за допомогою вспоміжних івентів.

Практичне значення одержаних результатів полягає у розробленому на основі проведених досліджень Agile – методології, що збільшує ефективність IT

команд. Розроблена інформаційна технологія підвищить рівень зручності управління IT-проектами.

Достовірність теоретичних положень магістерської кваліфікаційної роботи підтверджується коректністю постановки завдання, коректністю валідації недоліків сучасних методологій, експериментальнимим тестуванням в існуючих командах рохробників.

Особистий внесок магістранта. Результати даної магістерської кваліфікаційної роботи отримані самостійно.

Апробація результатів роботи. Результати досліджень апробовані на науково-технічній конференції «Молодь в науці: дослідження, проблеми, перспективи (МН-2022)».

Публікації. За результатами досліджень опубліковано тези [1].

1 ОБҐРУНТУВАННЯ АКТУАЛЬНОСТІ ТЕМИ ДОСЛІДЖЕННЯ

1.1 Аналіз предметної області управління ІТ проектами

Управління проектами – це використання специфічних знань, навичок, інструментів і прийомів для надання людям чогось цінного. Розробка програмного забезпечення для покращення бізнес-процесу, будівництво будівлі, надання допомоги після стихійного лиха, розширення збуту на новий географічний ринок – усе це приклади проектів.

Усі проекти – це тимчасова спроба створити цінність за допомогою унікального продукту, послуги чи результату. Усі проекти мають початок і кінець. У них є команда, бюджет, графік і набір очікувань, яким команда повинна відповідати. Кожен проект унікальний і відрізняється від рутинних операцій – поточної діяльності організації – тому що проекти досягають завершення, коли мета досягнута.

Змінний характер роботи через технологічні досягнення, глобалізацію та інші фактори означає, що все частіше робота організовується навколо проектів, коли команди об’єднуються на основі навичок, необхідних для виконання конкретних завдань.

Керують цими проектами менеджери – люди, яких навмисно або через обставини просять переконатися, що команда проекту досягає своїх цілей. Менеджери проекту використовують багато різних інструментів, прийомів і підходів, щоб задовольнити потреби проекту.

Деякі проекти необхідні для швидкого вирішення проблем з розумінням того, що покращення будуть зроблені протягом певного періоду часу. Інші проекти мають більшу тривалість і/або виробляють продукт або інші результати, які не потребуватимуть суттєвих покращень за межами запланованого обслуговування, наприклад, шосе.

Інші будуть поєднанням обох цих типів проектів. Проектні менеджери використовують різноманітні навички та знання, щоб залучити та мотивувати

інших для досягнення цілей проекту. Також вони мають вплив на успіху проектів і дуже затребувані, щоб допомогти організаціям досягти своїх цілей.

Протягом історії людства управління проектами завжди практикувалося неформально, але воно почало з’являтися як окрема професія в середині 20-го століття, коли група далекоглядних людей з аерокосмічної, інженерної, фармацевтичної та телекомунікаційної сфер усвідомили, що світ змінюється. потрібні нові інструменти. Мотивовані потребою вирішити проблеми планування та ресурсів, пов’язаних із все більш складними проектами, вони зібралися, щоб розпочати встановлення та стандартизацію інструментів для нової професії. А в 1969 році зародився Інститут управління проектами (PMI – Project Management Institute) [1].

Рисунок 1.1 – Офіційний логотип Інституту управління проектами

Сьогодні ми живемо в епоху проектної економіки (The Project Economy), де проекти є рушійною силою того, як виконується робота, реалізуються зміни та надається вартість.

У The Project Economy світове зростання рівня управління проектами доводить його цінність як:

- як визнана та стратегічна організаційна компетентність;
- як предмет навчання та виховання;
- як кар'єрний шлях.

Зараз загальновизнано, що базові знання з управління проектами можуть бути корисними для людей з різними ролями в широкому діапазоні починань. Навички управління проектами можуть допомогти молодому студенту, який працює над науковим проектом, досягти успіху, або керівнику компанії врегулювати особисті суперечки. Ці навички можуть допомогти медсестрі впорядкувати зміни, щоб покращити час реагування пацієнтів у своїй палаті. Вони можуть допомогти IT-фахівцям розробляти інноваційне програмне забезпечення в рекордно короткі терміни або допомогти державній установі покращити послуги, які вони надають, у більш економічний спосіб[1;2].

1.2 Медологія розробки программного забеспечення

Перед стартом кожного проекту для менеджера наступним кроком буде з’ясувати, які методології управління проектами підходять цьому проекту і команді розробників.

Ландшафт методологій управління проектами може здатися дещо приголомшливим.

Незалежно від того, чи маєте менеджер офіційну сертифікацію з управління проектами, або навчається стати менеджером проектів з досвіду, на вибір є абсолютний вибір методологій проектів. І вони часто мають свої правила, списки, принципи та нескінченні скорочення.

Методологія управління проектами – це набір принципів і практик, які допоможуть вам організувати свої проекти для забезпечення їх оптимальної ефективності. [3]

По суті, це фреймворк, який допомагає менеджеру найкращим чином керувати проектом.

Управління проектами дуже важливе для організацій і команд, але для того, щоб воно було дійсно ефективним, потрібно переконатися, що правильно зіставлена методологія управління проектами з типом команди, проектом, організацією та цілями.

Немає двох абсолютно однакових проектів (навіть якщо компанія використовує такі зручні функції, як шаблони проектів, щоб повторити свої минулі успіхи).

І якщо врахувати різні цілі, ключові показники ефективності (KPI) та методи виробництва не лише різних типів команд, але й різних типів галузей, має сенс, що не існує універсального підходу до управління проектом. Те, що найкраще працює для одного типу команди, може стати абсолютним кошмаром для іншого. Наприклад, багато розробників програмного забезпечення почали виявляти, що традиційні методи управління проектами заважають, а не допомагають, їх робочим процесам і негативно впливають на їхню продуктивність і результати.

В результаті команди програмного забезпечення почали розробляти новий тип методології управління проектами, яка була розроблена для вирішення їхніх конкретних проблем [4;5;6].

Невдовзі інші команди та галузі почали адаптувати ці нові методи управління проектами відповідно до своїх унікальних потреб і проблем. І так далі й далі, коли різні методології управління проектами переробляються та адаптуються для різних галузей та налаштовуються відповідно до конкретних випадків використання.

Існує багато факторів, які впливають на те, яка методологія управління проектами підходить для вашого проекту, команди та організації:

- Вартість і бюджет: з яким бюджетом ви працюєте? Чи можна це змінити, якщо це необхідно, чи важливо, щоб воно залишалося в цих наперед визначених межах?
- Розмір команди: скільки людей задіяно? Скільки зацікавлених сторін? Чи є ваша команда відносно компактною та самоорганізованою, чи більш розгалуженою, з потребою в більш суворому делегуванні?
- Здатність йти на ризик: чи це величезний проект із великим впливом, яким потрібно ретельно керувати, щоб досягти дуже серйозних результатів? Або це проект меншого масштабу з трохи більше простору для гри?

- Гнучкість: чи є можливість змінювати масштаби проекту під час процесу? А як щодо готового продукту?
- Хронометраж: скільки часу відведено на виконання? Вам потрібен швидкий оборот, чи важливіше, щоб у вас був прекрасний результат, незалежно від того, скільки часу це займе?
- Співпраця між клієнтами/зацікавленими сторонами: наскільки клієнт/зацікавлена сторона має потребу або хоче брати участь у процесі? Наскільки ви потребуєте – чи хочете – щоб вони були залучені [8]?

1.3 Аналіз актуальних методологій управління IT проектами

1.3.1 Каскадна модель управління IT проектами. Методологія Waterfall— це лінійний підхід до управління проектом, коли вимоги зацікавлених сторін і замовників збираються на початку проекту, а потім створюється послідовний план проекту, щоб задовольнити ці вимоги. Модель водоспаду названа так тому, що кожна фаза проекту переходить каскадом у наступну, плавно спускаючись вниз, як водоспад.

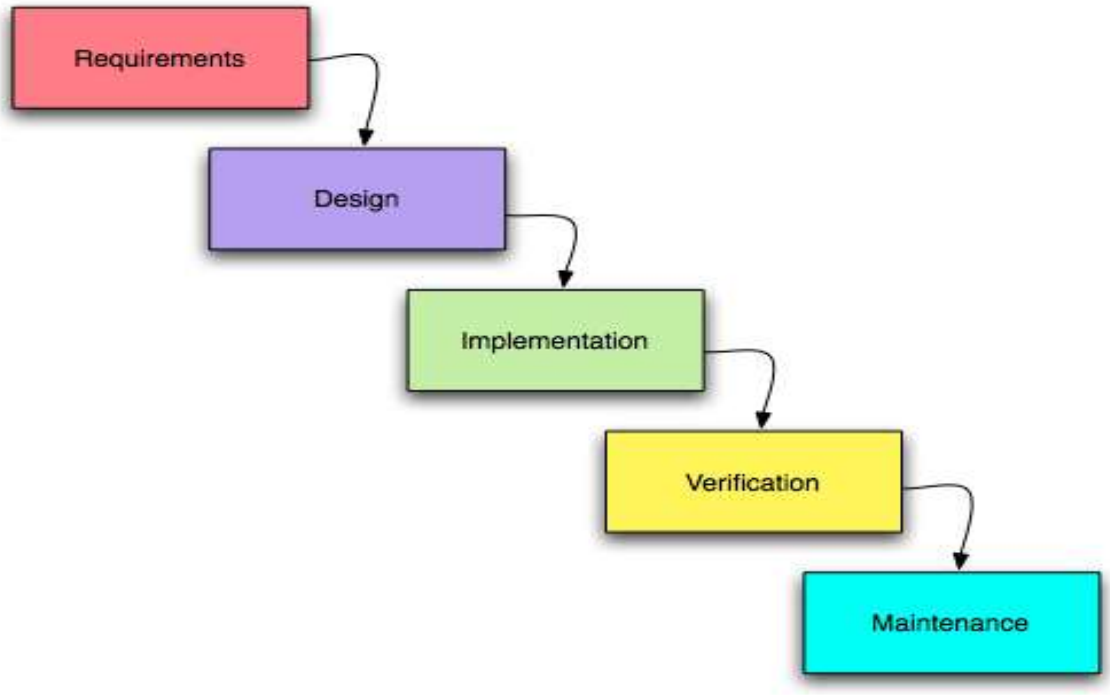

Рисунок 1.2 – Стадії каскадної методології Waterfall

Це ретельна, структурована методологія, яка існує вже давно, тому що вона працює. Деякі галузі, які регулярно використовують модель водоспаду, включають будівництво, IT та розробку програмного забезпечення. Як приклад, життєвий цикл розробки програмного забезпечення каскаду, або водоспад SDLC, широко використовується для управління проектами програмної інженерії.

Водоспадний підхід має принаймні п’ять-сім фаз, які слідують у строгому лінійному порядку, де фаза не може розпочатися, доки не буде завершена попередня фаза. Конкретні назви сходинок водоспаду відрізняються, але спочатку їх винахідник Вінстон В. Ройс визначив у такий спосіб:

Вимоги: ключовим аспектом методології водоспаду є те, що всі вимоги замовника збираються на початку проекту, що дозволяє планувати кожну наступну фазу без подальшого листування з клієнтом, поки продукт не буде завершено. Передбачається, що всі вимоги можуть бути зібрані на цьому етапі управління водоспадом.

Проектування: фазу проектування процесу водоспаду найкраще розділити на дві підфази: логічний дизайн і фізичний дизайн. Підфаза логічного проектування — це коли обговорюються та теоретично обговорюються можливі рішення. Підфаза фізичного проектування – це коли ці теоретичні ідеї та схеми перетворюються на конкретні специфікації.

Реалізація: фаза впровадження — це коли програмісти засвоюють вимоги та специфікації попередніх етапів і створюють фактичний код.

Перевірка: на цьому етапі клієнт переглядає продукт, щоб переконатися, що він відповідає вимогам, викладеним на початку проекту водоспаду. Це здійснюється шляхом відпуску готової продукції замовнику.

Технічне обслуговування: Клієнт регулярно використовує продукт на етапі технічного обслуговування, виявляючи помилки, неадекватні функції та інші помилки, що виникли під час виробництва. Виробнича група застосовує ці виправлення за необхідності, доки клієнт не буде задоволений[8].

1.3.2 Agile управління проектами. Agile – це ітеративний підхід до управління проектами, який зосереджується на розбивці великих проектів на більш керовані завдання, які виконуються за короткі ітерації протягом життєвого циклу проекту. Команди, які використовують методологію Agile, можуть виконувати роботу швидше, адаптуватися до мінливих вимог проекту та оптимізувати свій робочий процес.

Як випливає з назви, Agile дозволяє командам бути краще оснащеними, щоб швидко змінювати напрямок і фокус. Компанії програмного забезпечення та маркетингові агентства особливо знають про тенденцію до змін від зацікавлених сторін проекту щотижня. Методологія Agile дозволяє командам переоцінювати роботу, яку вони виконують, і коригувати з певними кроками, щоб переконатися, що в міру того, як змінюється робота та клієнтський ландшафт, змінюється і фокус для команди.

Якщо ви новачок в управлінні проектами Agile, на перший погляд, це може виглядати як складна і складна в управлінні система. Але, усвідомлюєте ви це чи ні, ви вже робите багато речей, яких вимагає Agile. За допомогою кількох налаштувань ви будете на шляху до коротших циклів розробки та менших, більш частих випусків продуктів.

Хто використовує Agile управління проектами?

Спочатку створений для розробки програмного забезпечення, Agile підхід до управління проектами швидко адаптується не тільки IT-командами. Маркетологи, університети, військові та навіть автомобільна промисловість також розглядають методологію Agile та інші структури Agile для доставки інноваційних продуктів у невизначених умовах. Багато організацій можуть отримати вигоду від управління проектами Agile, і його легко налаштувати та використовувати.

У світі програмного забезпечення, коли приймається рішення про створення або подальший розвиток існуючої технології, кінцевий продукт може бути важко визначити. Agile допускає цю неоднозначність через свою гнучкість, щоб змінювати напрям проекту, коли робота рухається в майбутнє.

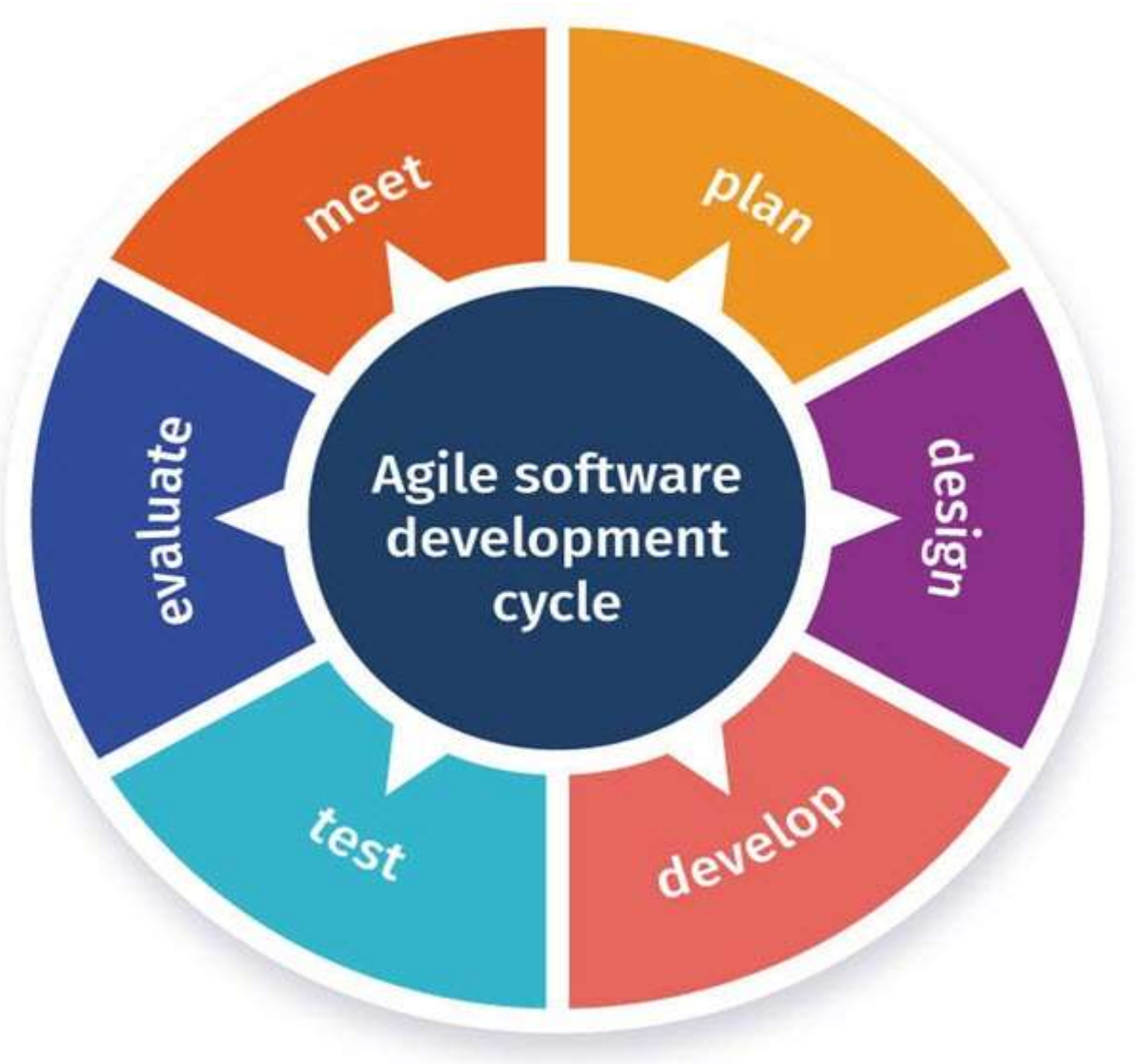

Рисунок 1.3 – Життєвий цикл Agile розробки

Хоча ви можете скористатися перевагами програмного забезпечення Agile, книг або Agile тренерів, кожна команда Agile унікальна, і розуміння основ може допомогти вам створити методологію Agile, яка працює для вас і вашої команди [9].

Agile - Маніфест окреслює 4 основні цінності та 12 керівних принципів, які служать полярною зіркою для будь-якої команди, яка використовує методологію Agile.

Чотири основні цінності Agile:

1. Люди та взаємодія важливіші за процеси та інструменти. Наскільки витонченішими стають технології, людський елемент завжди буде відігравати важливу роль у будь-якому управлінні проектами. Надто покладаючись на процеси та інструменти, ви не зможете адаптуватися до мінливих обставин.

2. Працюючий продукт важливіший за вичерпну документацію. Наскільки важливою є документація, робоче програмне забезпечення – це більше. Ця цінність полягає в тому, щоб дати розробникам саме те, що їм потрібно для виконання роботи, не перевантажуючи їх.

3. Співпраця із замовником важливіша за погодження умов контракту. Ваші клієнти є одним із ваших найпотужніших активів. Незалежно від того, чи є внутрішні або зовнішні клієнти, залучення їх протягом усього процесу може допомогти забезпечити більш ефективне задоволення кінцевого продукту їхнім потребам.

4. Готовність до змін важливіша за проходження початкового плану. Це значення є одним із найбільших відхилень від традиційного управління проектами. Історично зміни вважалися витратами, яких слід уникати. Agile дозволяє безперервно змінювати протягом життя будь-якого проекту. Кожен спринт дає можливість перегляду та корекції курсу[10].

Методології Agile можуть бути такими ж різноманітними та унікальними, як і кожна окрема команда, але 12 принципів Agile завжди повинні керувати вашими рішеннями та розробкою продукту.

– Наш найвищий пріоритет – задовольнити клієнта шляхом своєчасної та безперервної доставки програмного забезпечення (або іншого, що ви поставляєте).

– Вітаємо зміни вимог, навіть на пізніх стадіях розробки. Гнучкі процеси використовують зміни для конкурентної переваги клієнта.

– Розділяйте проект на часові відрізки, від кількох тижнів до кількох місяців, віддаючи перевагу коротшим термінам.

– Члени координаційної групи повинні працювати разом щодня протягом усього проекту.

– Створюйте проекти навколо мотивованих людей. Дайте їм необхідне середовище та підтримку та довірте їм виконання роботи.

– Бесіда віч-на-віч є найефективнішим і ефективним методом передачі інформації різним командам і всередині них.

– Кінцевий продукт є основним показником прогресу.

– Гнучкі процеси сприяють сталому розвитку. Усі зацікавлені сторони повинні мати можливість підтримувати постійний темп на невизначений термін.

– Постійна увага до технічної досконалості та гарний дизайн покращує маневреність.

– Простота – мистецтво максимізації обсягу невиконаної роботи – має важливе значення.

– Найкращі архітектури, вимоги та дизайн виходять із самоорганізуючих команд.

Через регулярні проміжки часу команда обмірковує, як стати ефективнішою, а потім відповідно налаштовує та коригує свою поведінку.

1.3.3 Гнучка методологія Scrum. Scrum – це методологія гнучкої розробки, яка використовується при розробці програмного забезпечення на основі ітераційних та інкрементних процесів. Scrum – це адаптивна, швидка, гнучка та ефективна гнучка структура, яка призначена для надання цінності клієнту протягом усього періоду розробки проекту. Основна мета Scrum – задовольнити потреби клієнта через середовище прозорості спілкування, колективної відповідальності та постійного прогресу. Розробка починається із загального уявлення про те, що потрібно побудувати, розробки переліку характеристик, упорядкованих за пріоритетом (відставання продукту), які хоче отримати власник продукту [9].

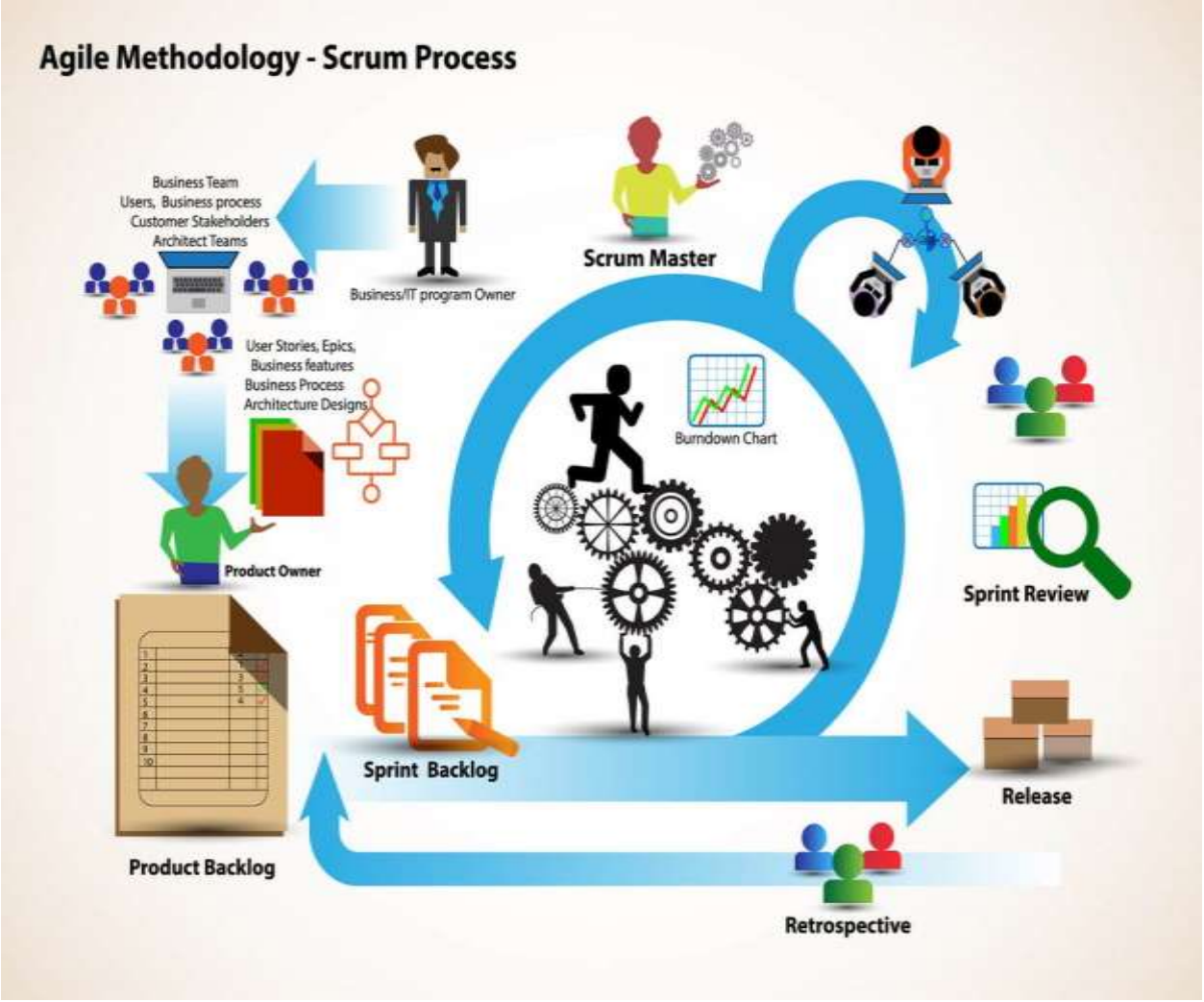

Рисунок 1.2 – Графічне відображення Scrum процессів

Scrum – це саме еволюція Agile-менеджменту. Методологія Scrum заснована на наборі чітко визначених практик і ролей, які повинні бути задіяні в процесі розробки програмного забезпечення. Це гнучка методологія, яка винагороджує застосування 12 принципів agile в контексті, узгодженому всіма членами команди продукту.

Scrum виконується у вигляді тимчасових блоків, які є короткими та періодичними, які називаються спринтами, які зазвичай тривають від 2 до 4 тижнів, що є терміном для зворотного зв’язку та роздумів. Кожен спринт є сутністю сам по собі, тобто він забезпечує повний результат, варіацію кінцевого продукту, який повинен бути доставлений клієнту з найменшими зусиллями, коли його запитують.

Відправною точкою процесу є перелік цілей/вимог, які складають план проекту. Саме клієнт проекту визначає пріоритетність цих цілей, враховуючи баланс вартості та вартості, таким чином визначаються ітерації та наступні поставки.

З одного боку, ринок вимагає якості, швидкої доставки при нижчих витратах, для чого компанія повинна бути дуже швидкою та гнучкою в розробці продуктів, щоб досягти коротких циклів розробки, які можуть задовольнити попит клієнтів, не підриваючи якість результату. . Це дуже проста в застосуванні методологія і дуже популярна завдяки швидким результатам.

Методологія Scrum використовується в основному для розробки програмного забезпечення, але інші сектори також користуються її перевагами, впроваджуючи цю методологію в свої організаційні моделі, такі як продажі, маркетинг, відділи персоналу тощо.

У Scrum команда зосереджена на створенні якісного програмного забезпечення. Scrum Product Owner зосереджується на тому, щоб визначити, які характеристики повинен мати продукт для створення (що створювати, що ні і в якому порядку) і подолати будь-які перешкоди, які можуть перешкодити команді розробників.

Скрам-команда складається з таких ролей:

- Scrum-майстер: людина, яка керує командою, направляючи її до дотримання правил і процесів методології. Scrum-майстер керує зменшенням перешкод проекту та працює з Власником продукту, щоб максимізувати рентабельність інвестицій. Скрам-майстер відповідає за те, щоб тримати Scrum в актуальному стані, забезпечуючи коучинг, наставництво та тренінги для команд у разі потреби.
- Власник продукту (PO): є представником зацікавлених сторін і клієнтів, які використовують програмне забезпечення. Вони зосереджуються на бізнес-частині та відповідають за рентабельність інвестицій проекту. Вони передають бачення проекту команді, перевіряють Story, які потрібно включити в Беклог продукту, і регулярно визначають їх пріоритети.

– Команда: Група професіоналів з необхідними технічними знаннями, які спільно розробляють проект, виконуючи Story, які вони зобов’язуються на початку кожного спринту.

1.3.4 Гнучка методологія Kanban. Kanban – це візуальна система для керування роботою під час її проходження через процес. Kanban візуалізує як процес (робочий процес), так і фактичну роботу, що проходить через цей процес. Мета Kanban – визначити потенційні вузькі місця у вашому процесі та виправити їх, щоб робота могла проходити через це економічно ефективно з оптимальною швидкістю або пропускною здатністю.

Рисунок 1.4 – Класична візуалізація Канбан-дошки

Методологія Канбан дотримується набору принципів і практик для управління проектами та покращення потоку роботи. Це еволюційний, не руйнівний метод, який сприяє поступовому вдосконаленню процесів організації. Якщо ви дотримуєтеся цих принципів і практик, ви зможете успішно використовувати Kanban для максимізації переваг для вашого бізнес-процесу –

покращення потоку, скорочення часу циклу, підвищення цінності для клієнта з більшою передбачуваністю – усе це має вирішальне значення для будь-якого бізнесу [10].

Нижче наведено чотири основоположні принципи та шість основних практик методології Канбан:

4 Основні принципи:

- **Почніть з того, що ви робите зараз**: метод Kanban (надалі іменований просто Kanban) наголошує, що не потрібно відразу вносити жодних змін у ваші існуючі налаштування/процеси. Kanban необхідно застосовувати безпосередньо до поточного робочого процесу. Будь-які необхідні зміни можуть відбуватися поступово протягом певного періоду часу у зручному для команди темпі.
- **Погодьтеся на поступові, еволюційні зміни**: Kanban заохочує вас вносити невеликі поступові зміни, а не вносити радикальні зміни, які можуть призвести до опору в команді та організації.
- **Спочатку поважайте поточні ролі, обов'язки та посади**: на відміну від інших методів, Kanban не нав'язує жодних організаційних змін сам по собі. Таким чином, не потрібно вносити зміни до ваших існуючих ролей і функцій, які можуть працювати добре. Команда спільно визначить та впровадить усі необхідні зміни. Ці три принципи допомагають організаціям подолати типовий емоційний опір і страх перед змінами, які зазвичай супроводжують будь-які ініціативи щодо змін в організації.
- **Заохочуйте акти лідерства на всіх рівнях**: Kanban заохочує постійне вдосконалення на всіх рівнях організації і говорить про те, що дії лідерства не повинні походити лише від керівників вищого рівня. Люди на всіх рівнях можуть надавати ідеї та демонструвати лідерство для впровадження змін, щоб постійно вдосконалювати спосіб надання своїх продуктів і послуг.

6 основних практик методу Канбан:

- Візуалізуйте хід роботи: це фундаментальний перший крок до прийняття та впровадження методу Канбан. Вам потрібно візуалізувати – на фізичній дошці або електронній дошці Kanban, етапи процесу, які ви зараз використовуєте для надання своєї роботи чи послуг. Залежно від складності вашого процесу та вашої робочої суміші (різних типів робочих елементів, над якими ви працюєте та надаючи), ваша дошка Kanban може бути дуже простою або дуже складною. Після того, як ви візуалізуєте свій процес, ви можете візуалізувати поточну роботу, яку виконуєте ви та ваша команда. Це може бути у вигляді наклейок або карток різних кольорів, які позначають різні класи обслуговування, або можуть бути просто різним типом робочих предметів. (У SwiftKanban кольори означають різні типи робочих елементів!) Якщо ви вважаєте, що це може бути корисно, ваша дошка Kanban може мати різні доріжки для плавання, по одній для кожного класу обслуговування або для кожного типу робочого елемента. Однак спочатку, щоб все було просто, ви також можете мати лише одну доріжку для плавання, щоб керувати всією своєю роботою – і згодом робити будь-який редизайн дошки.
- Limit WIP (Work in Progress): Обмеження незавершеного виробництва (WIP) є основоположним для впровадження Kanban – «системи витягування». Обмежуючи WIP, ви заохочуєте свою команду спочатку завершити роботу, перш ніж братися за нову роботу. Таким чином, роботи, які зараз виконуються, мають бути завершені та відзначені як виконані. Це створює потенціал у системі, тому команда може залучати нову роботу. Спочатку може бути нелегко визначити, якими мають бути ваші обмеження WIP. Насправді, ви можете почати без обмежень WIP. Великий Дон Рейнертсен пропонує (він зробив це на одній із конференцій Lean Kanban), що ви можете почати без обмежень WIP і просто спостерігати за початковою роботою, яка виконується, коли ваша команда починає використовувати Kanban. Коли у вас буде достатньо даних, визначте обмеження WIP для кожного етапу робочого процесу

(кожного стовпця вашої дошки Kanban) як рівні половині середнього WIP.

- Як правило, багато команд починають з ліміту WIP в 1–1,5 рази більше, ніж кількість людей, які працюють на певному етапі. Обмеження WIP та встановлення лімітів WIP на кожній колонці ради не тільки допомагає членам команди спочатку закінчити те, що вони роблять, перш ніж братися за нові справи, але також повідомляє клієнту та іншим зацікавленим сторонам, що існує обмежена здатність виконувати роботу для будь-якого команда – і вони повинні ретельно спланувати, яку роботу вони просять виконати команду.
- Керування потоком: Керування та покращення потоку є основою вашої системи Kanban після того, як ви впровадили перші 2 практики. Система Kanban допомагає вам керувати потоком, висвітлюючи різні етапи робочого процесу та статус роботи на кожному етапі. Залежно від того, наскільки чітко визначено робочий процес і встановлені ліміти WIP, ви спостерігатимете або плавний потік у межах WIP, або роботу, яка накопичується, коли щось затримується і починає затримувати ємність. Все це впливає на те, наскільки швидко робота проходить від початку до кінця робочого процесу (деякі люди називають це потоком створення цінностей). Kanban допомагає вашій команді аналізувати систему та вносити зміни, щоб покращити процес, щоб скоротити час, необхідний для виконання кожної частини роботи.
- Ключовим аспектом цього процесу спостереження за вашою роботою та вирішенням/усуненням вузьких місць є перегляд проміжних етапів очікування (проміжних етапів Готово) і перегляду, як довго робочі елементи залишаються на цих «стадіях передачі». Як ви дізнаєтеся, скорочення часу, витраченого на цих етапах очікування, є ключем до скорочення часу циклу. У міру того, як ви покращуєте потік, робота вашої команди стає більш гладкою та передбачуваною. Оскільки це стає більш передбачуваним, вам стає легше давати надійні зобов’язання

перед клієнтом щодо того, коли ви закінчите роботу, яку для нього виконуєте. Покращення вашої здатності надійно прогнозувати час завершення є важливою частиною впровадження системи Kanban!

– Зробіть політику процесу чіткою: як частина візуалізації вашого процесу, має сенс також чітко визначити та візуалізувати ваші політики (правила процесу або вказівки) щодо того, як ви виконуєте роботу, яку ви виконуєте. Формулюючи чіткі вказівки щодо процесу, ви створюєте загальну основу для розуміння всіх учасників, як виконувати будь-який тип роботи в системі. Правила можуть бути на рівні дошки, на рівні доріжок для плавання та для кожної колонки. Вони можуть бути контрольним списком кроків, які необхідно виконати для кожного типу робочого елемента, критеріїв входу-виходу для кожного стовпця або взагалі чогось, що допомагає членам команди добре керувати потоком роботи на дошці. Приклади явних політик включають визначення того, коли завдання завершено, опис окремих доріжок чи колон, хто коли тягне тощо. Політики мають бути чітко визначені та відображені зазвичай у верхній частині [11].

1.4 Життєвий цикл розробки программного забеспечення

Планування проекту буде неможливим, якщо немає життєвого циклу проекту, про який можна говорити. Це стосується ряду заходів, які необхідно виконати або виконати для досягнення цілей проекту. Зрозуміло, проекти відрізняються за розміром, спрямованістю та складністю. Проте всі вони мають один і той же життєвий цикл [12].

Життєвий цикл проекту включає п'ять фаз. Це також часто називають традиційним управлінням проектом, яке має п’ять основних етапів процесу:

1. Ініціювання

Це передбачає визначення характеру та обсягу проекту. Для цього необхідно вивчити бізнес-середовище та зрозуміти, як воно працює. Деякі з ключових заходів на цьому етапі:

– аналіз вимог бізнесу;

– оцінка історичних та поточних даних про діяльність бізнесу, включаючи фінансові звіти та бюджети;

– визначення зацікавлених сторін та аналіз їх ролі та впливу;

– визначення потреб зацікавлених сторін;

– визначення цілей проекту.

Саме на цьому етапі часто проводяться техніко-економічні обґрунтування. Це чудові інструменти для визначення можливих варіантів, які можуть вирішити наявні проблеми та допомогти досягти мети проекту.

Також часто на цьому етапі обирають і призначають керівника проекту, а також членів проектної групи та інших робочих груп.

2. Планування

Переходимо до більш детальної фази проекту. План проекту або блок-схема готується для планування часу, графіка, витрат і розподілу ресурсів для виконання заходів у проекті. Це передбачає врахування вартості супутніх ризиків під час реалізації заходів проекту. Також на цьому етапі команда проекту отримає остаточне схвалення для продовження проекту. Заходи, що виконуються на етапі планування, включають:

– створення команди планування;

– визначення результатів проекту, включаючи цілі якості та заходи контролю;

– визначення заходів, які необхідно виконати;

– розробка структури розбивки робіт та відображення їх взаємозв’язків;

– отримання кошторису витрат на матеріали, обладнання, оплату праці та інші витрати;

– складання бюджету проекту;

– розробка графіка виконання заходів у розкладі робіт;

– ідентифікація потенційних загроз, проблем або ризиків і формулювання відповідних заходів, якщо ці загрози, проблеми або ризики виникнуть під час реалізації проекту.

У проекті немає фіксованої кількості заходів, які необхідно виконати. Деякі проекти можуть мати лише кілька завдань, тоді як інші проекти складаються з довгого списку заходів. У цьому немає нічого поганого, якщо ви стежите за метою.

3. Виконання або впровадження

Це етап впровадження, на якому виконуються або виконуються ключові дії проекту, щоб отримати результати проекту, визначені раніше. Іншими словами, план проекту зараз запущений.

Тут проводяться такі заходи:

– розподіл ресурсів на відповідні заходи або фази проекту;

– координація з ключовими зацікавленими сторонами;

– виконання завдань, зазначених у плані;

– звітування про хід проекту на регулярних зустрічах.

4. Моніторинг і контроль

На кожному кроці проекту необхідно відстежувати його прогрес. Це корисно, тому проблеми можна виявити та вирішити, щоб мінімізувати ризики. Важливу роль на цьому етапі відіграє зворотній зв'язок. Можуть бути відхилення від базової лінії або цілі, встановленої раніше. Якщо їх неможливо виправити, щоб повернути речі до початкового плану, їх слід задокументувати як відхилення.

Цей етап вимагає:

– вимірювання діяльності в міру їх проведення;

– слідкувати за змінними проекту та постійно порівнювати їх із планом;

– вживання відповідних дій для усунення проблем і вирішення проблем;

– регулярна звітність перед зацікавленими сторонами;

– документування ходу та оновлення плану, якщо є;

– огляд результатів проекту відповідно до базової лінії або цілей.

5. Закриття або завершення

На цьому етапі проект офіційно оголошується завершеним. Це відбудеться лише тоді, коли зацікавлені сторони приймуть і будуть задоволені кінцевим результатом або результатом.

Воно включає:

– випуск або доставка остаточних результатів;

– документація та архівування файлів проекту та інших відповідних документів, які використовуються та створюються протягом усього проекту;

– проведення огляду після впровадження, де обговорюються набуті уроки з метою їх застосування в майбутніх проектах.

Також необхідно офіційно повідомити всіх зацікавлених сторін про закриття проекту [13;14].

1.5 Висновок до розділу 1

В даному розділі проаналізовано предметну область управління IT проектами, розглянуті популярні методолігії розробки ПЗ та розібрано типовий життєвий цикл розробки. Виходячи з проведеного аналізу гнучких методів розробки програмного забезпечення можна зробити такі висновки:

– жодна з сучасних гнучких методологій не є ідеальною;

– не існує методології, яка зможе бути єфективною на проектах різних типів.

2 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ УПРАВЛІННЯ ІТ ПРОЕКТАМИ

2.1 Вибір методології як основу для покращенної

Аналіз предметної області управлення ІТ проектами вказує на те, що в галузі розробки программного забеспечення уснують багато різноманітних фреймворків управління. Не існує фреймворка який підходив би кожній команді розробників та будь-якому проекту.

Порівняльна характеристика методологій подана у таблиці 2.1.

Таблиця 2.1 – Порівняльна характеристика методологій Scrum та Kanban

Назва методології	Плюси	Мінуси
Scrum	– Scrum може допомогти командам швидко й ефективно завершити результати проекту – Scrum забезпечує ефективне використання часу та грошей – Великі проекти поділяються на легко керовані спринти – Розробки кодуються та перевіряються під час огляду спринту – Добре працює для швидко розвиваються проектів	– Scrum часто призводить до скорочення обсягу через відсутність певної кінцевої дати – Шанси на провал проекту високі, якщо люди не дуже віддані чи співпрацюють – Застосування фреймворку Scrum у великих командах є складним завданням – Фреймворк може бути успішним лише з досвідченими членами команди

	– Команда отримує чітку видимість через зустрічі Scrum – Scrum, будучи гнучким, сприймає відгуки клієнтів і зацікавлених сторін – Короткі спринти набагато легше вносять зміни на основі зворотного зв'язку – Індивідуальні зусилля кожного члена команди видно під час щоденних зустрічей Scrum	– Щоденні зустрічі іноді розчаровують членів команди – Якщо хтось із членів команди покине проект у середині проекту, це може мати величезний негативний вплив на проект – Якість важко реалізувати, поки команда не пройде агресивний процес тестування
Kanban	– Простота використання: Канбан – це дуже простий і зрозумілий підхід, що робить його практичним для ефективного застосування керівництва компанії. Вам не потрібно бути експертом, щоб працювати з підходом Kanban. – Сприяє постійним і стійким удосконаленням різних функцій компанії: Канбан-підхід складається не тільки з ручних інструкцій або карток, а й візуалізації результатів процесу, що полегшує аналіз	– Не вписується в динамічне середовище: підхід Канбан передбачає стабільні та послідовні плани. – Неможливість ітерації: створення програмного забезпечення на ітераціях є основою для більшості процесів розробки, що не є невід'ємною частиною Kanban. – Відсутність часу: з кожною фазою не закріплені часові рамки, що може бути невигідним

	роботи. Це також може виділити інші потенційно невизначені області, на яких необхідно зосередитися. – Адаптивність: Канбан заохочує до максимальної адаптивності, що неймовірно для більш масштабних підприємств, які потребують постійних змін. – Співпраця: Kanban сприяє розвитку співпраці та змушує всю команду працювати разом, щоб досягти ідеальних результатів. – Низькі накладні витрати: Нагляд за використанням канбан-дошки, карток та аналіз результатів є легшим у порівнянні з більшістю методологій. – Зменшує витрати та втрати: Канбан виділяється тим, що виділяє проблеми процесу та їх вирішення. Система Kanban покращує потік і управління, безпосередньо допомагаючи компанії задіяти існуючі системи	для замовника або самої компанії.

	компанії, наприклад, «точно вчасно» (JIT) і зробити на замовлення тощо, що зменшує витрати на перевезення або зберігання.	
Waterfall	– Наявність інструкцій та правил по всьому процесу. Робота починається з докладного аналізу вимог замовника та того, як буде реалізовано проект. Плани, етапи та процеси затверджуються заздалегідь, фіксуються у документах та питань не викликають. Виконавцю потрібно просто їм слідувати. – Визначеність у термінах та бюджеті. Вартість товару та терміни здачі проекту розраховані та затверджені на самому початку і не змінюються в процесі. – Відсутність додаткових витрат на комунікацію у команді. Навіть якщо прийде новий розробник або тестувальник, зрозуміти завдання і почати роботу	– Відсутність гнучкості. Неможливо передбачити усі проблеми у проекті заздалегідь. Через жорстку послідовність етапів недоліки стануть відомі тільки в кінці проекту, доведеться робити додаткові ітерації і починати роботу наново, а це нові витрати і зайві робочі години. – Замовник не допускається до розробки та тестування. Він не може коментувати макети чи прототипи і бачить результат лише наприкінці проекту. Якщо змінилися вимоги або умови, це неможливо заздалегідь врахувати. – Проблеми спливають лише під час тестування.

	вийде швидко: для всіх процесів є описані правила.	Зробити частину роботи і відразу протестувати чи поєднати розробку та тестування, щоб знайти вразливість, не можна. Тестування починається після закінчення розробки, тому часто недоліки виявляються надто пізно.

Scrum частіше використовують на нових проектах, де розробка буде проводитись з самого початку і до виходу в Live, в той час як Kanban популярний на саппорт проектах, де сам продукт вже створений і приносить прибуток, але існує необхідність додавати новий функціонал. Kanban також часто використовують в командах дизайнерів та бізнес аналітиків. Waterfall найменш популярний серез проаналізованих фреймворків, тому що не підходить для швидко змінних бізнес сфер. Проте часто використовуються в медичній та воєнній сферах, тому що вимоги зафіксовані заздалегідь і не будуть змінюватись під час розробки.

Основою для розробки інформаційної технології управління IT проектами буде методологія Scrum [8;9].

2.2 Практичний досвід використання Scrum методологій відчизняними компаніями

Світ переживає дуже складний час. Епідемія короновірусу повністю змінила підхід к робочим процесам не тільки в IT галузі, а і в будь-якому іншому бізнесі. Локдауни та обемеження спонукають до повної зупинки усіх робочих процессів, або до переходу на віддалену роботу. Навчання в освітніх закладах стало онлайн,

усюди можлива доставка на дім будь-яких товарів. І навіть ті професії, які раніше сприймалися лише як оффлайнові, також змогли перейти в онлайн. І після такого людство точно не стане таким як раніше!

Сфера розробки ПЗ вже давно звикла до ремоуту. Розробники з України можуть спокійно працювати в команді з європейцями та американцями, та разом створювати програмні продути для східних країн, не покидаючи при цьому рідної оселі.

ІТ-сфера в Україні – одна з найдинамічніших і найперспективніших. Її вже давно називають локомотивом розвитку української економіки. За даними сайту DOU.UA, зараз в українській ІТ-індустрії працює понад 190 000 фахівців. А компаній близко 4000. І більшість з них працює с закордонними замовниками, на європейських та американських стартапах в режимі віддаленної роботи.

До епідемії розробники, дизайнери, бізнес аналітики та тестувальники все одно знаходились в одному офісі, працювали командами в одній кімнаті та постійно спілкувались. А зараз усього цього немає, кожен учасник команди теперь повинен працювати з дому. І це створює досить багато проблем, тому що жодна з методологій не адаптована під повний ремоут. Навіть в офіційних маніфестах по методологіям Scrum (Scrum Guide) та Kanban (Kanban Manifesto) вказано, що команди мають територіально знаходитись в одній кімнаті для максимальної продуктивності.

Під час аналізу предметної області управління ІТ проектами, були розглянуті найпопулярніші Agile методології управління проектами по розробці програмного забеспечення. Всі ці методології чітко сформовані та задокументовані, перевірені часом і не підлягають змінам. Проте, дуже мала кількість відчизняних ІТ компаній, при використанні данних фреймворків, дотримуються усіх принципів Agile. Співпрацючи з різними українськими підприємствами та ознайомившись з їх робочими процессами, стало зрозуміло що в більшості випадків необхідно нехтувати багатьмя законами фреймворків задля покращення робочих процессів. Кожен окремий проект унікальний, склад

команд завжди різний, рівень розробників також не однаковий. Тому і зміни в процессах постійно різні.

Проте компанії на співбесідах з кандидатами все одно кажуть, що працють по методології Scrum. Але це не відповідає дійсності, і працюють вони за неіснуючою методологією без назви, яка чимось нагадує Scrum.

2.3 Проектування робочих процесів поліпшеної методології управління IT проектами на основі фреймворку Scrum

Scrum як методологія позиціонує себе як ітеративна і циклічна, тобто усі процеси повторюються з визначенним інтервалом. Ці процеси можуть тривати нескінченно, до моменту завершення проекту. При роботі за цією методологією команда матиме 5 кроків, які також називають Scrum-фази.

1. Створення Беклогу продукту.

Це створення списку функціоналу, який необхідно виконати команді розробників на цьому проекті.

На цьому кроці функціонал декомпозується та візуалізуються як стікери на дошці. Кожен такий стікер ще називають тікетами.

Загалом уснує декілька типів тікетів:

- епік (epic) - велике завдання, на вирішення якої команді потрібно кілька спринтів;
- історія (story) - частина великого завдання (епіка), яку команда може вирішити за 1 спринт;
- завдання (task) - технічне завдання, яке робить один із членів команди;
- під-завдання (sub-task) - частина історії/завдання, яка описує мінімальний обсяг роботи члена команди;
- баг (bug) - завдання, яке описує помилку в системі.

Історії користувачів перетворюються з великих елементів і стають меншими, які можна помістити в беклог продукту. Епіки також можуть бути включені до

бэклогу продукту, але не можуть бути включені до відставання спринту без перетворення його в історію користувача.

Типовий приклад історії користувача: як адміністратор, я хочу додавати, змінювати та видаляти завдання для користувачів на веб-сайті.

2. Планування спринту

Тривалість спринту дуже важлива, щоб історій користувачів було якомога менше. Типова середня тривалість спринту триває близько 2 тижнів. Якщо тривалість спринту невелика, перевага полягає в тому, що можна отримати більше відгуків від клієнтів, а більшість помилок і помилок можна усунути раніше. Якщо тривалість спринту велика, це дозволяє розробнику працювати ретельно.

3. Створення беклогу спринту.

На даному етапі команда scrum має вибрати важливі історії користувачів і перетворити їх на менші завдання. Вони повинні спланувати, як виконати завдання. Крім того, важливо визначити пріоритети необхідних завдань.

По завершенню спринта, він може отримати два статуси:

- Спринт успішний: весь розроблений функціонал розроблений, протестований, продемонстрований замовнику та виконує свою бізнес ціль;
- Спринт провальний: функціонал не був розроблений до кінця в задані часові рамки, дефектний або не виконує свою бізнес ціль. Тоді усі розробки за спринт необхідно видалити та почати розробку спочатку.

4. Робота над спринтом

Фактичні історії користувачів переміщуються у вигляді невеликих завдань у відставання спринту, де починається фактична робота. Тут починається реалізація програмного додатка, наприклад, розробка веб-сайту.

Для початку використовується дошка завдань, яку також називають дошкою Канбан, з великою кількістю карток. На картках вказуються такі відомості про завдання, як доручена особа, деталі роботи, термін виконання або тривалість тощо. Дошка завдань складається з стовпців «Завдання», «Робота виконується»,

а потім стовпці «Тестування» та «Виконана робота». Картки можна переміщати зліва направо в порядку бажаного та залежно від завершення.

На цьому кроці важливі scrum мітинги, оскільки вони проводяться для відстеження статусу прогресу та того, хто який статус виконує. Тому щоденно на початку робочого дня проводять DSM .

DSM або Daily Scrum Meeting – це 15-хвилинний мітинг для Scrum команди. Зустрічі, як правило, проводяться в одному місці й в один і той же час щодня. В ідеалі щоденні збори Scrum проводяться вранці, оскільки це допомагає встановити контекст для роботи майбутнього дня. Основною ціллю мітингу є засинхронізуватися. DSM не використовується як нарада для вирішення проблем. Питання, які піднімаються, розглядаються окремо, щом не затримувати учасників команди, кого ця проблема не стосується. Під час щоденної сутички кожен член команди відповідає на наступні три запитання:

– Що ви робили вчора?
– Що ви будете робити сьогодні?
– Чи є якісь перешкоди на вашому шляху?

Зосереджуючись на тому, що кожна людина досягла вчора і досягне сьогодні, команда отримує відмінне розуміння того, яка робота була виконана, а яка робота залишилася. DSM – це не зустріч з оновленням статусу, на якій бос збирає інформацію про те, хто відстає від графіка. Скоріше, це зустріч, на якій члени команди беруть зобов’язання один перед одним.

Нажаль, більшість учасників цих мітингів надають перевагу спілкуванню без увімкненної веб-камери, що руйнує командну єдність і відштовхує членів команди один від одного.

5. Тестування та демонстрація продукту

Виконані завдання мають бути реалізовані як робочий продукт з випробуванням повного життєвого циклу. Вартість тестування може бути мінімізована за допомогою додавання Quality Assurance Engineer (тестувальника ПЗ) або меншої кількості історій, однак перше є найкращим рішенням. Кожен

виконаний спринт повинен бути продемонстрований замовнику для того, щоб він погодився і його погляд на повне рішення

6. Ретроспектива та планування наступного спринту

Результатом цього кроку є обговорення того, що пройшло добре, а що можна покращити для наступного рівня. Крім того, вам потрібно обговорити отримані уроки та підводні камені будь-яких конкретних питань чи проблем. Потім слід розпочати планування наступного спринту на основі знань, які ми маємо про поточні процеси та минулі проекти.

Склад команди та ролі мають величезний вплив на процесси та їх формування. Класична скрам команда складається з Product Owner, Scrum Master та розробників. Але наразі команда, зазвичай, включає ще керівників проектів, тестувальників, дизайнерів UI/UX, бізнес аналітиків та DevOps. Scrum Guide не розраховний на сучасний склад команди, тому процеси для них не адаптовані. Також Scrum Guide передбачає, що усі члени команди досвідченні розробники, у яких низька вірогідність помилки. Але в реальності це далеко не так. Сучасна різнороллева та різнорівнева команда при класничному скрамі не зможе працювати в повну силу.

2.4 Висновок до розділу 2

В даному розділі проаналізовані робочі процеси методології Scrum, описано особливості проектування методологій управління IT проектів, розглянуто адаптації методології відчизняними компаніями. Також в результаті порівняльної характеристики вибрано методологію для вдосконалення.

3 РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ УПРАВЛІННЯ ІТ ПРОЕКТАМИ

3.1 Постановка задачі

Більшість команд, особливо ті, які були сформовані нещодавно, відчують деяке падіння продуктивності. Це не повинно бути приводом для занепокоєння: це просто ще один доказ того, що спільне розташування дійсно має значення. Також було виявлено, що Agile значно полегшує віддалену роботу. Багато компаній, які працюють agile, кажуть, що наступні принципи допомагають їм стати ефективнішими віддалено:

1. Жорстке визначення пріоритетів. Створення відставання та постійне їх уточнення забезпечили можливість окремих людей і команд працювати над найціннішими речами, навіть коли вони віддалені.
2. Робота в невеликих, міжфункціональних автономних командах. Невеликі команди розвивають стійкість і полегшують зміну напрямку, коли це необхідно.
3. Регулярні мітинги. Допомагають переконатися, що всі отримують необхідну інформацію, і дає змогу співпрацювати, коли люди працюють вдома.
4. Agile менеджмент. Хороші проектні менеджери зосереджуються на конкретних результатах і цілях, дозволяючи командам мати спільне бачення, над яким можна працювати.

По суті, agile — це набір культурних цінностей, принципів і поведінки, а не набір конкретних практик. Команди, які залишаються вірними принципам, все ще можуть ефективно працювати, поки життя та бізнес не повернуться до більш нормальних моделей. Ось як команди повинні зосередити свої зусилля.

Невеликі, міжфункціональні команди з повноваженнями є ядром кожної гнучкої організації. Здатність діяти автономно стимулює мислити креативно та дає змогу командам швидко приймати рішення. Але високий ступінь автономії працює лише тоді, коли існує також високий ступінь узгодженості в командах і

між ними, і важливість цього узгодження підвищується, коли члени команди працюють віддалено.

Agile-менеджери повинні подвійно переконатися, що команди узгоджуються із загальною метою, стратегією та пріоритетами компанії. Керівники повинні повідомляти про наміри, пояснюючи як причини, так і що, щоб члени залишалися зосередженими на цілях своєї команди та зв'язку з більшими бізнес-цілями. У звичайні часи це вирівнювання є попередником автономії; у часи відволікання, коли команди фізично розділені, вирівнювання стає важливішим, ніж будь-коли.

Отже, для реалізації інформаційної технології управління IT проектами необходно виконати наступні задачі:

- сформувати список івентів;
- модифікувати робочі процеси під актуальні склади команд;
- зберегти принципи Agile.

3.2 Реалізація інформаційної технології управління IT проектами

3.2.1 Реалізація поліпшениих робочих процесів. В розробленій методологій сформовані такі принципи:

- спринт триває два тижні;
- два спринта формують одну ітерацію проекту;
- наповненість беклогу ітерації проводить керінник проекту разом з бізнес аналітиком на спільному мітингу з клієнтом;
- формування планування на ітерацію робиться з урахування рівня кожного розробника.
- кожна оцінка функціоналу розробниками автоматично отпримають 15% додаткового часу, на випадок зміни вимог, на ліквідацію багів або у випадку помилкової оцінки.

Об’єднання двух Спринтів в Ітерацію грає важливу роль в новій методології. Завдяки коммунікації бізнес аналітика, керівника проекту та клієнта у команди буде можливість сформувати беклог на довший термін, у бізнес аналітика – більше часу на пропрацювання документації з вимогами по функціоналу. При тестуванні розробленого функціоналу у QA інженерів буде більше часу на пошук та валідацію багів, а у розробників більше часу на їх ліквідацію. Це автоматично підвищить якість розробленого функціоналу.

В свою чергу замовник матиме змогу корегувати функціонал, який розробляти необхідно в найблищий час, що підвищує лояльність до команди і компанії.

Керівник проекту, в свою чергу, зможе детально спланувати наступну ітерацію завдяки Sprint Grooming і отриманим з цього івенту даними.

Також, буде можливість демонструвати замовнику розроблений функціонал в кінці кожної ітерації, що дає можливість заощадити час розробників на процесі деплоя коду на необхідні середовища. Також це дає можливість ефективно використовувати години DevOps інженера.

Додавання «буферу» 15% до кожної оцінки дає можливість команді бути спокійними при будь-яких непередбачуваних ситуаціях. Якщо спринт завершено раніше дати закінчення, заощаджений час переноситься на наступний спринт. По завершенню ітерації проводиться аналіз зайвого часу(якщо такий є) і коригується буфер на наступну Ітерацію.

В кінці кожної ітерації обов’язкові Iteration Review Meeting та Iteration Retrospective.

3.2.2 Реалізація івентів нового фреймворку. В поліпшенній методології використовуються наступні івенти:

- Daily Standup Meeting. Цей івент слід модифікувати правилом, завжди включенної веб камери. Усі інші функції цього мітингу слід залишити незмінними.

- Refinement Meeting. В сучасних командах за формування вимог до розроблюваного функціоналу відповідають бізнес аналітики, які постійно спілкуються з замовником та кларифікують відкриті питання по проекту. В класичному скрамі цю роль на себе приймає Product Owner, але через велику кількість обов’язків він це робить на високому рівні деталізації. Саме тому, у розробників виникають безліч питань по функціоналу, який вони розробляють. А відповіді на запитання неймовірно сильно затримають розробку. Бізнес аналітик, в свою чергу, максимально детально описує функціонал, і завдяки Refinement meeting зможе повністю покрити необхідність в повному розумінні функціоналу розробниками. Детальне ознайомлення дає змогу надавати більш точну оцінку в годинах розробки. На момент проведення Refinement Meeting документація з описаним функціоналом переглянуто та затверджено клієнтом. Мітинг буде проводитись двічі на спринт, на якому бізнес аналітик буде ознайомлювати команду з функціоналом, який буде в розробці наступного спринта. Хронометраж данного івенту – 1 година.
- Sprint Grooming. Івент ціллю якого буде дати оцінку функціоналу який буде в розробці наступного спринта. Проводитись буде наступного дня після останнього Refinement meeting, хронометраж – 30 хвилин.
- Функціонал буде оцінюватися в годинах розробки кожним членом команди. На основі отриманих годин, керівник проекту зможе створити короткотривале планування, враховуючи рівень кожного розробника і отримати очікуванні дати завершення спринта.
- Iteration Review Meeting. Процесс демонстрації розробленого функціонала замовнику, проводиться один раз на ітерацію. Хронометраж – не більше 1.5 години.

– Iteration Retrospective. Івент повністю ідентичний версії методології Scrum, але проводиться один раз на ітерацію. Хронометраж – не більше 1 години.

3.3 Тестування розробленої методології управління IT проектами

Тестування проводилось на базі діючого проекту, в два етапи:

– 4 тижні використовуючи класичний Scrum;

– 4 тижні використовуючи розроблену методологію.

При проведенні тестування розробленої методології управління IT проектів маємо наступний склад команди:

– Middle Java Back-end розробник;

– Junior Java + JavaScript FullStack розробник;

– Senior JS React Front-End розробник;

– Senior QA інженер;

– Бізнес аналітик;

– Middle DevOps інженер ;

– Керівник проекту (PM).

Команда на час тестування не змінювалась, робочий день – 8 годин.

Тестовий проект: розробка маркетплейсу для оптових B2B продажів.

Критерії оцінки результатів: сумарна кількіть витраченого часу за ітерацію; суб'єктивні враження членів команди.

Витрачені години под час тестування подані у таблиці 3.1

Таблиця 3.1 – Витрачені години под час тестування

Член команди	Scrum		Розроблена методологія	
	Оригінальна оцінка(год.)	Фактичний витрачений час(год.)	Оригінальна оцінка(год.)	Фактичний витрачений час(год.)

Middle Java Back-end розробник	145	160	160	154
Junior Java + JavaScript FullStack розробник	140	162	150	154
Senior JS React Front-End розробник	128	140	130	129
Senior QA інженер	120	140	160	120
Бізнес аналітик	160	160	160	160
Middle DevOps інженер	24	24	24	12
Всього	717	786	784	729

За перший етап тестування з використанням класичної методології Scrum, команда виконала заданний обсяг задач за 784 робочі години, при запланованих 717. При такій ситуації, ітерація була виконувалася довше ніж було заплановано на 9.34% довше, що несе за собою наступні наслідки:

- розробникам необхідно було працювати понаднормово для дотримання запланованих дат релізу, що привело до погіршення робочої атмосфери в команді та вигоранню декількох її учасників;
- через більший термін розробки функціоналу, тестування було розпочато пізніше запланованого часу, що могло вплинути на якість фінального продукту і несе за собою фінансові втрати;
- ІТ компанія зазнала фінансових втрат, тому що додатковий час в розмірі 69 годин не був обговорений з замовників проекту;

– понаднормові години оплучуються розробнику по збільшенному тарифу.

За другий етап тестування з використовуванням покращенної методології управління IT проектами, команда завершила розробку функціоналу за 729 годин, при запланованих 784. Тобто на 7% швидше:

– Додаткові івенти з бізнес аналітиком дали можливість більше поринути в функціонал та заощадило час розробників на кларифікації функціональних вимог;
– Ефективне використання робочих годин дало можливість заощадити 55 робочих годин компанії, та використати цей час у наступній ітерації;
– Успішно завершена ітерація підняла загальний настрій команди та надала додаткової мотивації для подальшої розробки;
– Завдяки завчасній розробці, тестування було розпочато раніше запланованого, що дало можливість більш детально все протестувати та зекономити час команди;
– Меньша кількість релізів дала можливість заощадити робочі години DevOps інженера вдвічі.

При аналізі результатів було визначено, що покращена методологія управління IT проектами збільшила єфективність розробки на 16% в порівнянні з класичним Scrum. Зміни в івентах були позитивно прийняті командою та на ретроспективі затверджені для використання при наступних ітераціях проекту.

Суб’єктивні враження членів команди, які були озвучені під час ретроспективи, подані у таблиці 3.2

Таблиця 3.2 – Враження команди на зміни в методології

Член команди	Враження
Middle Java Back-end розробник	«Набагато легше надавати оцінку функціоналу, який детально розібрано з бінес аналітиком»

Junior Java + JavaScript FullStack розробник	«Щоденні мітинги стали цікавішими, і користі від них стало більше; тепер бачучи щоранку веселі обличчя своїх колег на дейлі, з'явилось більше мотивації»
Senior JS React Front-End розробник	«Об'єм розробленного функціоналу став трішки меньмим, проте якість коду стала вище»
Senior QA інженер	«З'явилось більше часу на тестування; за рахунок буферу, до кінця ітерації майже не залишилось відкритих багів(залишились низькоприорітетні), якість вище і морально спокійніше»
Бізнес аналітик	«Незвично проводити мітинги на всю команду, звик тет-а-тет розповідати про функціонал; з'явилось більше часу на підготовку специфікацій»
Middle DevOps інженер	«За рахунок того, що реліз тепер один раз в місяць, з'явилось більше часу на інші проекти»

3.4 Висновок до розділу 3

В даному розділі було розроблено методологію управління IT проектами та протестовано на діючому проекті. Результати тестування вказують на те, що:

- ефективність команди розробників зросла на 16%;
- процесс оцінки функціоналу став точнішим;
- якість фінального продукту стала вищою;
- мітинги по кларифікаціям вимог дають ясність по специфікації функціоналу, що буде найблищим часов в розробці;
- використання покращенної методології дає можливість IT компаніям заощаджувати кошти за рахунок ефективного використання робочих годин команд розробників.

4. ЕКОНОМІЧНА ЧАСТИНА

Економічна частина є завершальним розділом магістерської дипломної роботи, в якому розробляються остаточні висновки щодо економічної ефективності запропонованої розробки. В даному розділі розглянемо основні питання конкурентоспроможності продукту та комерційного потенціалу розробки.

4.1 Проведення комерційного та технологічного аудиту науково-технічної розробки

Метою проведення комерційного і технологічного аудиту є оцінювання науково-технічного рівня та рівня комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності, тобто під час виконання магістерської кваліфікаційної роботи.

Для проведення комерційного і технологічного аудиту залучимо 3-х незалежних експертів. У нашому випадку такими експертами будуть провідні викладачі випускової та споріднених кафедр.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу будемо здійснювати за 12-а критеріями згідно рекомендацій.
Результати оцінювання комерційного потенціалу розробки заносимо до таблиці 4.1.

Таблиця 4.1 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки

Критерії	Експерти		
	Експерт 1	Експерт 2	Експерт 3
	Бали, виставлені експертами		
Технічна здійсненність концепції	4	3	4

Ринкові переваги (наявність аналогів)	4	3	3
Ринкові переваги (ціна продукту)	4	4	4
Ринкові переваги (технічні властивості)	4	3	4
Ринкові переваги (експлуатаційні витрати)	3	3	3
Ринкові перспективи (розмір ринку)	3	4	4
Ринкові перспективи (конкуренція)	3	4	3
Практична здійсненність (наявність фахівців)	4	3	3
Практична здійсненність (наявність фінансів)	3	4	4
Практична здійсненність (необхідність нових матеріалів)	3	3	4
Практична здійсненність (термін реалізації)	3	3	3
Практична здійсненність (розробка документів)	4	4	4
Сума балів	42	40	42
Середньоарифметична сума балів $\overline{СБ}$	41,66		

За даними таблиці 4.1 робимо висновок щодо рівня комерційного потенціалу розробки. При цьому користуємося рекомендаціями, наведеними в таблиці 4.2.

Таблиця 4.2 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів, розрахована на основі висновків	Рівень комерційного потенціалу розробки
0 – 10	Низький
11 – 20	Нижче середнього
21 – 30	Середній
31 – 40	Вище середнього
41 – 50	Високий

Оскільки середньоарифметична сума балів складає 41,66, то рівень комерційного потенціалу розробки високий, тому дана розробка є реальною для подальшої її реалізації та впровадження.

4.2 Розрахунок витрат на здійснення науково-дослідної роботи

4.2.1 Витрати на оплату праці. Витрати на основну заробітну плату дослідників розраховують відповідно до посадових окладів працівників, за формулою:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \quad (4.1)$$

де M_{ni} – місячний посадовий оклад конкретного розробника (інженера, дослідника, науковця тощо), грн.; T_p– середня кількість робочих днів в місяці, $T_p \approx 21 \ldots 23$ дні; t_i– кількість днів роботи конкретного дослідника.

Дану розробку буде проводити інженер, величина окладу буде становити 11000 грн. на місяць. Кількість робочих днів у місяці складає 21, а кількість робочих днів дослідника складає 55.

Зведемо сумарні розрахунки до таблиці 4.3.

Таблиця 4.3 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Кількість днів роботи	Витрати на заробітну плату, грн
Керівник проекту	13500	642,85	10	6428,5
Інженер	11000	523,80	55	28809,52
Всього				35238,02

4.2.2 Розрахунок додаткової заробітної плати робітників. Додаткова заробітна плата $З_д$ розраховується як 10-12% від суми основної заробітної плати дослідників та робітників за формулою

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \qquad (4.2)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати.

На даному підприємстві додаткова заробітна плата начисляється в розмірі 10% від основної заробітної плати.

$$З_д = 0{,}10 \cdot 35238{,}02 = 3523{,}80(\text{грн.})$$

4.2.2 Відрахування на соціальні заходи. Нарахування на заробітну плату $Н_{ЗП}$ дослідників та робітників, які брали участь у виконанні даного етапу роботи, розраховуються за формулою :

$$З_{дод} = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%}, \qquad (4.3)$$

де $Н_{зп}$ – норма нарахування на заробітну плату.

Дана діяльність відноситься до бюджетної сфери, тому ставка єдиного внеску на загальнообов'язкове державне соціальне страхування буде складати 22%, тоді:

$$Н_{зп} = (35238{,}02 + 3523{,}80) \cdot \frac{22}{100} = 8527{,}60 \text{ (грн.)}$$

Отже, нарахування на заробітну плату складають 8527,60 грн.

4.2.3 Розрахунок витрат на комплектуючі. Витрати на комплектуючі, які використовують при дослідженні нового технічного рішення, розраховуються, згідно з їхньою номенклатурою за формулою:

$$K_{\text{в}} = \sum_{j=1}^{n} H_j \cdot \text{Ц}_j \cdot K_j, \qquad (4.4)$$

де H_j – кількість комплектуючих j-го виду, шт.; Ц_j – покупна ціна комплектуючих j-го виду, грн; Kj - коефіцієнт транспортних витрат, (1,1...1,15). Проведені розрахунки зводимо до таблиці 4.4.

Таблиця 4.4 – Витрати на комплектуючі

Найменування комплектуючих	Кількість, шт.	Ціна за штуку, грн.	Разом
Папір	1	160	160
Ручка	1	10	10
Флешка	1	285	285
Всього			500,5

4.2.4 Амортизація обладнання, програмних засобів та приміщення. У спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо можуть бути розраховані з використанням прямолінійного методу амортизації за формулою:

$$\text{А}_{\text{обл}} = \frac{\text{Ц}_{\text{б}}}{T_{\text{в}}} \cdot \frac{t_{\text{вик}}}{12}, \qquad (4.5)$$

де $\text{Ц}_{\text{б}}$ –балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн.; $t_{\text{вик}}$ – термін корисного використання обладнання, програмних засобів, приміщень під час досліджень,

місяців; $T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

Проведені розрахунки зводимо до таблиці 4.5.

Таблиця 4.5 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн.	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн.
Комп'ютер	15000	5	6	1800
Всього				1800

4.2.5 Паливо та енергія для науково-виробничих цілей. Витрати на силову електроенергію розраховують за формулою:

$$В_е = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_е \cdot К_{впі}}{\eta_i}, \quad (4.6)$$

де W_{yi} – встановленя потужність обладнання на певному етапі розробки, кВт; t_i – тривалість роботи обладнання на етапі дослідження, год; $Ц_е$ – вартість 1 кВт-години електроенергії, грн; $К_{впі}$ – коефіцієнт, що враховує використання потужності; η_i – коефіцієнт корисної дії обладнання.

Проведені розрахунки зведемо до таблиці 4.6.

Таблиця 4.6 – Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Комп'ютер	0,5	75	136,5
Освітлення приміщення	0,6	55	120,12

Всього	256,62

Витрати на проведення науково-дослідної роботи розраховуються як сума всіх попередніх статей витрат за формулою:

$$В = З_о + З_д + З_н + К + А_{обл} + В_е, \quad (4.7)$$

$$В = 35238{,}02 + 3523{,}80 + 8527{,}60 + 500{,}5 + 1800 + 256{,}62 = 49846{,}54$$ (грн)

Загальні витрати на завершення науково-дослідної роботи та оформлення її результатів розраховуються за формулою:

$$ЗВ = \frac{В_{заг}}{\eta}, \quad (4.8)$$

Загальні витрати складають

$$ЗВ = \frac{49846{,}54}{0{,}9} = 55385{,}04 \text{ (грн.)}$$

4.3 **Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором**

Розрахуємо можливе збільшення чистого прибутку у потенційного інвестора для кожного із років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки за формулою:

$$\Delta П_i = (\pm \Delta Ц_0 \cdot N + Ц_0 \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot \left(1 - \frac{\vartheta}{100}\right), \quad (4.9)$$

де $\pm \Delta Ц_0$ – зміна основного якісного показника від впровадження результатів науково-технічної розробки в аналізованому році; N – основний кількісний показник, який визначає величину попиту на аналогічні чи подібні розробки у році до впровадження результатів нової науково-технічної діяльності; $Ц_0$ – основний якісний показник, який визначає ціну реалізації нової науково-технічної розробки в аналізованому році; ΔN – зміна основного кількісного показника від впровадження результатів науково-технічної розробки в аналізованому році; λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість; ρ – коефіцієнт, який враховує рентабельність інноваційного продукту (послуги), рекомендується приймати 0,2...0,5; ϑ – ставка податку на прибуток.

В середньому в рік продається 300 розробок. Середня вартість такої розробки становить 1000 грн.

Впровадження зразка розробки дозволяє збільшити ціну кожного зразка на 200 грн, враховуючи ціни конкурентів. Також прогнозується, що попит на даний продукт зросте, оскільки даний продукт відрізняється якістю від конкурентних.

Попит збільшиться за перший рік на 200 примірників, за наступний на 250 та протягом третього року – ще на 150 примірників.

Ставка податку на додану вартість в 2021 році залишилась на рівні 20% , а коефіцієнт λ=0,8333. Ставка податку на прибуток складає 18%.

Коефіцієнт, який враховує рентабельність продукту, дорівнює 0,3.

Отже, розрахуємо збільшення чистого прибутку підприємства на 2022 -2024 рр.:

$$\Delta П_{2022} = (1000 \cdot 300 + (1000 + 200) \cdot 200) \cdot 0{,}8333 \cdot 0{,}3 \cdot \left(1 - \frac{18}{100}\right)$$

$$= 49259{,}52 \text{ (грн.)}$$

$$\Delta П_{2023} = (1000 \cdot 300 + (1000 + 200) \cdot (200 + 250)) \cdot 0{,}8333 \cdot 0{,}3 \cdot \left(1 - \frac{18}{100}\right) = 172193{,}11 \text{ (грн.)}$$

$$\Delta П_{2024} = (1000 \cdot 300 + (1000 + 200) \cdot (200 + 250 + 150)) \cdot 0{,}8333 \cdot 0{,}3 \cdot \left(1 - \frac{18}{100}\right) = 209091{,}63 \text{ (грн.)}$$

Далі розрахуємо приведену вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \qquad (4.10)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн; T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки; τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$$ПП = \frac{49259{,}52}{(1 + 0{,}1)^1} + \frac{172193{,}11}{(1 + 0{,}1)^2} + \frac{209091{,}63}{(1 + 0{,}1)^3} = 344183{,}36 \text{ (грн.)}$$

Далі розрахуємо величину початкових інвестицій, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки. Для цього можна використати формулу:

$$PV = k_{\text{інв}} \cdot \text{ЗВ}, \quad (4.11)$$

де $k_{\text{інв}}$ – коефіцієнт, що враховує витрати інвестора на впровадження науко-во-технічної розробки та її комерціалізацію; *ЗВ* – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, грн.

$PV = 2 \cdot 55385{,}04 = 110770{,}08$ (грн)

Тоді абсолютний економічний ефект або чистий приведений дохід для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$\text{Е}_{\text{абс}} = \text{ПП} - PV, \quad (4.12)$$

де *ПП* – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, грн;
PV – теперішня вартість початкових інвестицій, грн.

$$\text{Е}_{\text{абс}} = (344183{,}36 - 110770{,}08) = 233413{,}28 \text{ (грн.)}$$

Внутрішня економічна дохідність інвестицій , які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки, розраховується за формулою:

$$\text{Е}_{\text{в}} = \sqrt[\text{Т}_{\text{ж}}]{1 + \frac{\text{Е}_{\text{абс}}}{PV}} - 1, \quad (4.13)$$

де $\text{Е}_{\text{абс}}$ – абсолютний економічний ефект вкладених інвестицій, грн; *PV* – теперішня вартість початкових інвестицій, грн; $\text{Т}_{\text{ж}}$– життєвий цикл науково-

технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, роки.

$$\text{Е}_\text{в} = \sqrt[3]{1 + \frac{233413{,}28}{110770{,}08}} - 1 = 0{,}45 = 45\%$$

Далі визначимо бар'єрну ставку дисконтування, тобто мінімальну внутрішню економічну дохідність інвестицій, нижче якої кошти у впровадження науково-технічної розробки та її комерціалізацію вкладатися не будуть.

Мінімальна внутрішня економічна дохідність вкладених інвестицій визначається за формулою:

$$\tau_{min} = d + f, \quad (4.14)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках;
f – показник, що характеризує ризикованість вкладення інвестицій.

$$\tau = 0{,}15 + 0{,}05 = 0{,}20$$

Далі розрахуємо період окупності інвестицій, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{\text{Е}_\text{в}}, \quad (4.15)$$

де $\text{Е}_\text{в}$ – внутрішня економічна дохідність вкладених інвестицій.

$$\text{Т}_\text{ок} = \frac{1}{0{,}45} = 2{,}22 \text{ року}$$

Термін окупності складає 2,22 року, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження цієї розробки та виведення її на ринок.

ВИСНОВКИ

В магістерській кваліфікаційній роботі проведено розробку покращенної інформаційної технології управління ІТ проетами.

В першому розділі проаналізовано предметну область управління ІТ проектами, розглянуті популярні методолігії розробки ПЗ та розібрано типовий життєвий цикл розробки. Виходячи з проведеного аналізу гнучких методів розробки програмного забезпечення були зроблені такі висновки:

- жодна з сучасних гнучких методологій не є ідеальною;
- не існує методології, яка зможе бути єфективною на проектах різних типів.

В другому розділі були проаналізовані робочі процеси методології Scrum, описано особливості проектування методологій управління ІТ проектів, розглянуто адаптації методології відчизняними компаніями. Також в результаті порівняльної характеристики вибрано методологію для вдосконалення.

В третьому розділі було розроблено методологію управління ІТ проектами та протестовано на діючому проекті. Результати тестування вказують на те, що:

- ефективність команди розробників зросла на 16%;
- процесс оцінки функціоналу став точнішим;
- якість фінального продукту стала вищою;
- мітинги по кларифікаціям вимог дають ясність по специфікації функціоналу, що буде найблищим часов в розробці;
- використання покращенної методології дає можливість ІТ компаніям заощаджувати кошти за рахунок ефективного використання робочих годин команд розробників.

Для проведення комерційного і технологічного аудиту залучено 3-х незалежних експертів. Визначено, що середньоарифметична сума балів складає 41,66, тому рівень комерційного потенціалу розробки високий, тому дана розробка є реальною для подальшої її реалізації та впровадження.

Визначено, що термін окупності складає 2,22 року, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження цієї розробки та виведення її на ринок.

Результати досліджень апробовані на науково-технічній конференції «Молодь в науці: дослідження, проблеми, перспективи (МН-2022)»[1]. За результатами магістерської кваліфікаційної роботи опубліковано тези доповідей.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Колесников Ю.С., Барабан С.В. Вплив Covid-19 на робочі процеси IT команд розробників. // Тези доповідей на науково-технічній конференції «Молодь в науці: дослідження, проблеми, перспективи (МН-2022)». – Вінниця: ВНТУ, 2021. Електронний ресурс (режим доступу https://conferences.vntu.edu.ua/index.php/mn/mn2022/paper/view/14153)
2. Вплив agile методологій на розробку ПЗ: [Електронний ресурс]. – Режим доступу: https://www.ccsenet.org/journal/index.php/cis/article/view/44383/ (дата звертання 02.09.2021)
3. Agile маніфест: [Електронний ресурс]. – Режим доступу: https://agilemanifesto.org/ (дата звертання 05.09.2021)
4. 12 принципів Agile: [Електронний ресурс]. – Режим доступу: https://agilemanifesto.org/principles.html (дата звертання 05.09.2021)
5. Найкращі інструменти для спільної роботи у 2021 році: [Електронний ресурс]. – Режим доступу: https://blog.jetbrains.com/space/2021/07/16/best-collaboration-tools/ (дата звертання 05.09.2021)
6. Scrum управління проектами: [Електронний ресурс]. – Режим доступу: https://www.digite.com/agile/scrum-methodology/ (дата звертання 05.09.2021)
7. Scrum Guide: [Електронний ресурс]. – Режим доступу: https://scrumguides.org/ (дата звертання 05.09.2021)
8. Все про управління IT проектами: [Електронний ресурс]. – Режим доступу: https://www.cleverism.com/everything-about-project-management/(дата звертання 05.09.2021)
9. IT в Україні: куди ми рухаємося? [Електронний ресурс]. – Режим доступу: https://dou.ua/lenta/columns/future-of-it-ukraine/ (дата звертання 05.09.2021)
10. Що таке Канбан: [Електронний ресурс]. – Режим доступу: https://www.digite.com/kanban/what-is-kanban/(дата звертання 05.09.2021)

11. Що таке Project Management: [Електронний ресурс]. – Режим доступу: https://www.pmi.org/about/learn-about-pmi/what-is-project-management (дата звертання 05.09.2021)
12. Чому Project Management важливий для сучасних проектів: [Електронний ресурс]. – Режим доступу: https://www.lucidchart.com/blog/why-is-project-management-important/ (дата звертання 05.09.2021)
13. Плюси та мінуси Scrum: [Електронний ресурс]. – Режим доступу: https://www.simplilearn.com/scrum-project-management-article (дата звертання 05.09.2021)
14. Що не так з сучасним Agile: [Електронний ресурс]. – Режим доступу: https://www.workamajig.com/blog/project-management-methodologies (дата звертання 05.09.2021)