МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет

Навчально-науковий інститут бізнесу, економіки і менеджменту

Кафедра управління імені Олега Балацького

«До захисту допущено»
завідувач кафедри

_____ Ігор РЕКУНЕНКО

____ _____ 20__ р.

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на здобуття освітнього ступеня бакалавр

зі спеціальності 073 Менеджмент

освітньо-професійної програми «Менеджмент»

на тему:

«Ефективне управління проектами за допомогою методології SCRUM в IT-компаніях»

Здобувачки групи ЕН-93/1м ***Діденко Марії Іванівни***

Кваліфікаційна робота бакалавра містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело _____ Марія ДІДЕНКО

Керівник_____ ст. викладач, к.е.н., доцент, Тетяна МАЙБОРОДА

Керівник _____ директор ТОВ «МАЙНДКЕЙ», Олег НЕСТЕРОВ

Суми – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
Кафедра управління імені Олега Балацького

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____
(підпис) (Ім'я та ПРІЗВИЩЕ)
_____ _____20_____ р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавр

зі спеціальності 073 Менеджмент, освітньо-професійної програми «Менеджмент»

Здобувача (ки) групи ЕН-93/1м Діденко Марії Іванівни

1. Тема роботи "Ефективне управління проектами за допомогою методології SCRUM в IT-компаніях" затверджена наказом №0569-VI від 25.05.2023р.
2. Термін подання здобувачем закінченої роботи 10.06.2023р.
3. Мета кваліфікаційної роботи: дослідження ефективності методології Scrum для управління IT-проектами.
4. Об'єкт дослідження: ефективність методології Scrum для реалізації IT-проектів.
5. Предмет дослідження: фактори впливу на успішну реалізацію проекту за використанням методології Scrum та основні проблеми, які виникають в процесі реалізації.
6. Кваліфікаційна робота виконується на підставі Scrum та досліджень щодо результатів і кращих практик її застосування в IT-проектах.
7. Орієнтовний план кваліфікаційної роботи, терміни подання розділів керівникові та зміст завдань для виконання поставленої мети.

№ пор.	Назва розділу	Термін подання
I	«Загальні методології управління проектами»	29.04.23
II	«Методологія управління проектами Scrum»	15.05.23
III	«Удосконалення процесу управління IT-проектом за методологією Scrum»	05.06.23

Зміст завдань для виконання поставленої мети кваліфікаційної роботи:

У розділі 1 студент повинен дослідити загальні методології управління проектами, їх історію, види та визначити сучасні тенденції в управлінні проектами.

У розділі 2 студент повинен ознайомитися з методологією Scrum, основними її принципами та артефактами, визначити особливості застосування та проаналізувати дослідження щодо результатів її використання в різних галузях.

У розділі 3 студент повинен провести власне дослідження щодо ефективності методології Scrum та розробити способи удосконалення результатів від використання даної методології.

8. Консультації щодо виконання роботи:

Розділ	Прізвище, ініціали та посада керівника/консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
1			
2			
3			

9. Дата видачі завдання 17.04.23

Керівник_____ ст. викладач, к.е.н., доцент, Тетяна МАЙБОРОДА

Керівник _____ директор ТОВ «МАЙНДКЕЙ», Олег НЕСТЕРОВ

Завдання до виконання одержав _____ Марія ДІДЕНКО

АНОТАЦІЯ

Структура та обсяг випускної кваліфікаційної роботи бакалавра. Робота складається зі вступу, трьох розділів, висновків, списку використаних джерел, який містить 40 найменувань. Загальний обсяг кваліфікаційної роботи становить 56 с., зокрема 11 рисунків, 1 таблиця, список використаних джерел із 4 сторінок.

Актуальність. Тенденція швидкого розвитку ринку спонукає зростанню популярності гнучких методів управління проектами, які дозволяють швидко розробляти і впроваджувати нові продукти. Дослідження методології Scrum, як найбільш розповсюдженого методу управління реалізацією проектів, допоможе визначити основні фактори ефективності даного фреймворку та виявити проблеми на шляху до успіху.

Мета роботи. Мета роботи полягає в дослідженні ефективності методології Scrum для управління проектами в IT-компаніях.

У роботі розглянуто теоретичні і практичні аспекти методології управління IT-проектами Scrum. За допомогою використання емпіричних методів дослідження на основі реального проекту було досліджено ефективність даної методології і виявлено основні проблеми її впровадження в управління реалізацією проекту. Було проаналізовано причини виникнення відхилень та наведено методи запобігання.

Визначено напрямки підвищення ефективності методології та запропоновано способи покращення результатів на основі проведеного дослідження.

Апробація результатів. Результати дослідження будуть використані для подальшого впровадження існуючого проекту.

Ключові слова. МЕТОДОЛОГІЯ, SCRUM, УПРАВЛІННЯ ПРОЕКТОМ, ГНУЧКА МЕТОДОЛОГІЯ, ІТЕРАЦІЯ, IT-ПРОЕКТ.

ЗМІСТ

ВСТУП

Актуальність. Гнучкі методології розробки програмного забезпечення набувають дедалі більшої популярності через значні переваги та цінність, яку вони надають. Agile методології на сьогодні використовуються провідними компаніями не лише в сфері IT, а й в медицині, освіті, фінансовій сферах. Дослідження ефективності Scrum-фреймворку допоможе висвітлити основні проблеми та фактори, які впливають на успішність його впровадження в IT-проекти.

Метою роботи є дослідження ефективності методології Scrum для управління IT-проектами. Для досягнення мети необхідно виконати такі завдання:

- проаналізувати дослідження щодо результатів застосування методології Scrum в реалізації проектів в IT-компаніях;
- провести впровадження методології в тестовий IT-проект;
- дослідити ефективність методології на основі отриманих результатів проекту.

Предметом дослідження є фактори впливу на успішну реалізацію проекту за використанням методології Scrum та основні проблеми, які виникають в процесі реалізації.

Об'єктом дослідження є ефективність методології Scrum для реалізації IT-проектів.

Методи дослідження. Методологічною основою роботи є емпіричні методи дослідження, такі як: спостереження, опис та вимірювання. У роботі було використано порівняльний аналіз, який дозволив побачити переваги і недоліки даної методології в порівнянні з іншими методологіями проектного менеджменту; метод синтезу, який дозволив розглянути основні сильні та слабкі

сторони впровадження даної методології в ІТ-проект. Теоретичною основою дослідження є роботи зарубіжних дослідників та вчених, присвячені аналізу впливу використання методології на успішність проектів різних сфер.

Результати проведеного дослідження висвітлюють проблеми впровадження гнучкої методології розробки в проект та визначають основні фактори, які обумовлюють ефективність даної методології. Отримані результати будуть корисними в подальшій реалізації досліджуваного стартап-проекту.

1. ЗАГАЛЬНІ МЕТОДОЛОГІЇ УПРАВЛІННЯ ПРОЕКТАМИ

1.1 Еволюція методологій проектного менеджменту

Дослідження історичних етапів розвитку проектного управління дозволяє повернутися в період формування та зародження проектного менеджменту, вивчити процес його еволюції та визначити можливості розвитку у майбутньому.

Процес зародження управління проектами (УП) розпочався на початку XX століття і був пов'язаний з розвитком космічної інженерії США. Потреба у використанні інструментів аналізу процесу управління створенням ракет стала ключовою причиною появи терміну «управління проектами». Вперше, невелика матрична розробка для організації проектів була застосована в 1953 році в проектах повітряних сил США. У 1957 році американською компанією DuPont, перед якою стояла задача реконструкції хімічних заводів з використанням нового стандарту виробництва, був створений метод критичного шляху (Critical Path Method, CPM) як механізм розробки найкоротшого шляху до отримання результату [1, 8-9 с].

Наступною розробкою була система сіткового планування (Programme evaluation and review technique - PERT), розроблена компанією «Локхид» з метою реалізації ракетної розробки. Використання даної методології дозволило вдало реалізувати проект раніше визначеного терміну, що визначило подальший успіх застосування даного методу. У 1958 році ці дві розробки активно застосовувалися під час планування робіт за проектом, контролю якості виконання проекту, оцінювання ризиків та ефективного управління наявними ресурсами [2, 21с.].

В 60-х роках XX століття сфера застосування сіткових моделей починає активно розширюватись. Пошук способів оптимізації витрат на організацію проектів призвело до розробки першої системи управління проектами компанією IBM. Згодом починають створюватися професійні організації з управління

проектами, які об’єднують провідних фахівців Європи. Однією з таких була неприбуткова організація International Project Management Association (IPMA), створена в 1965 році, яка займалась розробкою стандартів з управління проектами. [2, 14 с.]

У 1969 році виник перший інститут управління проектами, як неприбуткова організація, метою якої було сприяння розвитку професійного управління проектами та розробка методології щодо комунікацій всередині проекту. У 1980-х роках вони розроблять низку стандартів проектного менеджменту, а у 1987 році вийшов перший довідник (PMBOK), в якому були зібрані фундаментальні практики для ефективного управління проектами в будь-якій галузі, управління проектами було сформовано як остаточну дисципліну.

На початку 70-х років методологія критичного шляху стає законодавчо обгрунтованою та застосовується в навчальних закладах в Сполучених Штатах, а також поширюється в Європу. Розробляються нові методи управління проектами та організаційні структури. В середині 80-х років починають розвиватися управління якістю та управління змінами в проекті, а згодом і управління ризиками, з’являється покращене технічне забезпечення, що дозволяє ефективніше використовувати методи УП [2].

Початок 90-х років був трансформуючим: управління проектами починають застосовувати в нетрадиційних на той час сферах, відбувається уніфікація та стандартизація процесів управління проектами, розробляється концепція проектно-орієнтованої організації, застосовуються нові інформаційні технології. IPMA випускає підручник з УП, який зібрав весь багаторічний досвід з управління проектами, що сприяє поширенню УП в інші сфери.

Як окрема сфера проектний менеджмент (Project Management) активно формувався протягом останніх 60-ти років. Сьогодні проектний менеджмент має широке розповсюдження серед різних організацій та підприємств, вдосконалюються стандарти з управління проектами, виникають нові школи

проектного менеджменту, які готують кваліфікованих спеціалістів, здатних успішно довести проект будь-якої складності від ініціації до завершення.

1.2 Гнучкі та каскадні методології управління проектами та особливості їх застосування

Методологія управління проектами є визначеним набором змістовно пов'язаних практик, процедур та процесів, які визначають спосіб планування, розробки, реалізації та контролю проекту протягом усього безперервного процесу від ініціації до успішного завершення. Це методичний, структурований і дисциплінований підхід до створення концепції, реалізації та закриття проекту, заснований на науково-доведених даних [4].

Метою застосування методології є контроль процесу реалізації проекту за допомогою прийняття ефективних рішень та усунення проблем завдяки визначеним технікам та підходам. Вона є свого роду скелетом, на основі якої відбувається розробка кроків щодо втілення проекту, визначається вартість та кінцева якість, а також часові проміжки.

В управлінні проектами існує багато різних методів, які можна поділити на каскадні та гнучкі, або, по-іншому – традиційні та сучасні. Найбільш традиційним підходом до управління проектами є методологія Waterfall (в перекладі з англ. – водоспад). Дана методологія була розроблена в 1970 році з метою розробки програмного забезпечення і отримала таку назву через каскадний спосіб управління проектом, коли кожний наступний етап починається лише після завершення попереднього.

Waterfall methodology – лінійний підхід до управління проектом, який полягає в послідовній реалізації проекту, згідно з планом та у відповідності до попередньо визначених вимог зацікавлених сторін [5]. Даний підхід є простим для розуміння, адже передбачає складення чіткого плану та поступове слідування

йому. Зазвичай вимоги до проекту з використанням методології водоспаду визначаються заздалегідь, ще до того, як проект розпочнеться, на відміну від гнучких методологій, де вимоги та план реалізації можуть змінюватися в процесі виконання проекту. Каскадний підхід включає близько п’яти фаз, кожна з яких розпочинається лише після завершення попередньої [6]:

1. Вимоги. Дана фаза спрямована на визначення основних вимог замовника та планування кожної наступної фази;
2. Фаза проектування, під час якої створюється план розробки продукту і визначається кінцева мета, а також кроки на шляху досягнення цієї мети;
3. Фаза реалізації розпочинається під час перших дій команди в рамках втілення завдань проекту і триває до моменту фактичного завершення основного обсягу роботи;
4. Фаза перевірки необхідна для того, щоб клієнт міг переконатися у відповідності результату його вимогам, які були обговорені на початку реалізації проекту;
5. Фаза технічного обслуговування допомагає виявити помилки на перших етапах застосування готового продукту замовником і вчасно їх виправити, а також забезпечити вчасне оновлення працюючих систем.

Серед переваг методології водоспаду варто зазначити фіксованість бюджету та графіку реалізації проекту, адже цілі проекту та вимоги залишаються незмінними до кінця його розробки, що полегшує його реалізацію. Недоліком даної методології є те, що для внесення певних змін в проект, необхідно повертатися до попередніх фаз циклу, що подовжує час реалізації проекту [7].

Сучасні методології управління проектами базуються не на лінійних процесах, а на гнучких методах розробки, які дозволяють вносити зміни в процесі реалізації проекту та можуть не мати чітко визначеного кінцевого результату. До сучасних гнучких методів управління проектами відносять Agile. Основним принципом його роботи є поділ проекту на короткі цикли, вкінці кожного з яких, команда отримує певний продукт, який можна презентувати

замовнику. На відміну від каскадних методологій управління проектами, в Agile етапи розробки можуть виконуватися паралельно, а ключовим моментом є наявність готового самостійного рішення на кожному етапі розробки. Нижче наведені ключові цінності Agile [6]:

1) Управління процесом за допомогою команди, а не інструментів;
2) Фокус на забезпеченні працюючих рішень замість вичерпної документації. Створення рішення та досягнення результату важливіше за розгорнуту документацію, яка затримує процес розробки;
3) Співпраця та взаємодія з клієнтом замість підписання контрактів. Клієнт має бути залучений до кожного етапу розробки проекту для повнішого задоволення його вимог та якісної співпраці;
4) Заплановані реагування на зміни. Внесення змін під час коротких ітерацій допомагають перетворити зміни на покращення, а не додаткові витрати.

Перевагами даної методології є постійна взаємодія між членами команди проекту, гнучкість, тобто можливість легко вносити зміни в проект під час розробки без повернення до попередніх етапів та швидкий результат. Недоліками Agile є невизначена вартість проекту, яку не можна порахувати відразу через відсутність точного кінцевого результату та велику кількість змін, що постійно вносяться, а також наявність цих змін, які можуть бути безрезультатними та відтягувати завершення проекту. Але все залежить від рівня кваліфікації команди та менеджменту в конкретному проекті.

Найбільш відомою гнучкою методологією управління проектами та підвидом Agile є Scrum (у перекладі з англ. означає «боротьба за м’яч у регбі»). Scrum має наступні ключові особливості:

- Наявність структурованих команд, кожний учасник якої має свою роль (замовник, скрам-майстер, розробник);

- Поділ проекту на спринти – окремі етапи розробки проекту, тривалістю один-два тижні, в рамках яких плануються результати та основні завдання по проекту;
- На початку роботи над проектом, створюється перелік всіх завдань по проекту (беклог). Кожний наступний спринт формується на основі цього беклогу;
- Результатом завершення кожного спринту має бути працездатний продукт, який можна представляти замовнику.

Детальніше методологію Scrum буде розглянуто в наступному розділі.

Ще одним підвидом Agile є методологія Kanban. Дана методологія була створена як проста система планування для контролю і управління роботою та запасами в компанії Toyota. Мета Канбан – «виявити потенційні вузькі місця в процесі розробки проекту та виправити їх, щоб робота могла протікати ефективно та з оптимальною швидкістю» [8].

Методологія Канбан заснована на певних принципах, які покликані покращити робочий процес за допомогою поступового вдосконалення організаційних процесів. Основоположні принципи Канбан [8]:

- Внесення змін до поточного робочого процесу відбувається поступово протягом певного часу;
- Поступові зміни замість радикальних;
- Відсутність потреби у внесенні змін до існуючих ролей та функцій, якщо вони добре працюють;
- Заохочення до вдосконалення на всіх рівнях керівництва.

Канбан – це система управління поступовими змінами в організації, яка націлена на вдосконалення існуючих процесів невеликими кроками. Якщо порівняти її зі Scrum, то ключовою відмінністю є відсутність спринтів та чіткого планування періоду здачі робіт – вони можуть почати виконуватися в будь-який момент. На відміну від Скрам, Канбан визначається не як самостійний метод

управління проектом, а як спосіб удосконалення існуючих процесів. Поєднання Scrum та Kanban утворює ще один гібридний підхід до управління проектами – Scrumban.

Scrumban поєднує в собі структуру методології Scrum і гнучкість та безперервний процес Kanban. Спочатку Scrumban був призначений для спрощення переходу команди від Scrum до Kanban, але саме поєднання обох методологій дозволяє отримати надзвичайно гнучку форму управління проектами. Унікальними особливостями Scrumban є відсутність ієрархії в команді, тобто кожен учасник може приймати рішення, а також відсутність кінцевого терміну завершення проекту, який тепер залежить від кількості завдань та мети [9].

До переваг застосування даної методології можна віднести: заощадження часу за допомогою запобігання повторюваності роботи та усунення завдань, які не відповідають ключовій меті проекту; можливе застосування в довгострокових проектах завдяки поступовим змінам протягом великих періодів часу; автономність команди та можливість її членів приймати певні рішення самостійно. До недоліків Scrumban належать: відсутність контролю за командою, що може стати причиною дезорганізації; відсутність стандартизованої структури, яка визначає правила організації процесів; нижчий рівень впливу менеджера проекту на процес роботи через відсутність конкретних ролей кожного учасника проекту [10].

Метод критичного шляху – метод визначення завдань, які необхідно виконати для реалізації проекту та планування термінів їх виконання. Ідея полягає в тому, щоб визначити найдовшу послідовність робіт, які необхідно виконати щоб закінчити проект вчасно. Даний метод був розроблений вкінці 1950-х років з метою зменшення витрат, що пов’язані з неефективним плануванням проекту [11]. В основі методу лежить побудова моделі проекту, що містить перелік всіх завдань, які необхідні для реалізації проекту та залежності між цими завданнями і оцінкою часу на їх виконання.

До основних причин застосування даного методу в проекті належить: ефективне планування проекту на основі фактичного перебігу робіт; більш ефективний розподіл ресурсів на основі пріоритетності завдань; визначення залежностей в проекті та уникнення перешкод, які є причиною втрати часу.

Методологія Lean (в перекладі з англ. – бережливий) – це набір практик та принципів управління проектами, спрямованих на оптимізацію бізнес-процесів шляхом виключення діяльності, яка не створює цінності для клієнта. Методологія бережливого виробництва заснована на повазі до людей та постійному вдосконаленні. Історія даної методології бере свій початок в 1950-х роках в компанії Toyota, як спосіб знизити витрати та зберегти високі прибутки. Нижче наведено п’ять ключових принципів даної методології [12]:

1. Ідентифікація цінності та усунення відходів (діяльність), які не приносять дану цінність;
2. Відображення потоку створення цінності – процесу, який проходять продукти під час реалізації;
3. Організація ефективного робочого процесу, що допомагає усунути вузькі місця, які створюють затримки в роботі;
4. Впровадження ощадливого методу для зменшення перевиробництва та скорочення часу, необхідного для виконання роботи командою;
5. Безперервне вдосконалення, що включає оцінку продуктивності та залучення команд на всіх етапах роботи до вдосконалення

1.3 Огляд сучасних тенденцій в управлінні IT- проектами

За останні роки відбувся значний технологічний розвиток, пов'язаний з поширенням використання штучного інтелекту, появою нових інструментів управління та перебудовою організаційних процесів для роботи у віддаленому режимі, що безпосередньо вплинуло й на управління проектами.

Першою тенденцією в управлінні проектами на 2023 рік є використання штучного інтелекту (ШІ). Поява штучного інтелекту, наприклад ChatGPT, вже спростило роботу працівників багатьох галузей. В управлінні проектами ШІ буде корисним в організації та розподілу ресурсів, оновленні плані проекту та управлінні ризиками [13]. Штучний інтелект вже використовується для автоматизації рутинних завдань, оптимізації робочих процесів та звільнення проектного менеджера для виконання більш важливих завдань.

Наступною тенденцією в управлінні проектами є поширення гібридних підходів до управління проектами, тобто поєднання гнучких та традиційних методологій в межах одного проекту. Застосування гібридних підходів означає використання основних переваг кожного виду методологій та мінімізації їх недоліків. При гібридному підході керівники проектів обирають найбільш доцільну методологію управління для кожної фази проекту або навіть для певних завдань в межах кожної фази. Наприклад, методологія Agile часто застосовується на етапі розробки продукту, а Waterfall на етапі тестування, що підвищує ефективність та покращує результати кожної фази [14].

Невід'ємною частиною управління проектами також стали цифрові інструменти, використання яких зумовлено їх зростаючою доступністю Програмне забезпечення використовується для планування проекту, впорядкування проектної документації, відстеження прогресу та покращення співпраці і комунікації між проектними командами. Наприклад такі інструменти як Asana, Trello та Microsoft Project допомагають менеджерам ефективніше керувати термінами, бюджетом та завданнями. Використання технології хмарних обчислень дозволяє проектним командам безпечно зберігати та поширювати проектну інформацію між командою проекту та стейкхолдерами, що сприяє прозорості та покращенню якості відділеної роботи.

Більшої актуальності також набуває тенденція ощадливого управління проектами. Ощадливе управління - це підхід в управлінні проектами, що

наголошує на усуненні відходів та оптимізації процесів для підвищення ефективності, зниження витрат ресурсів та збільшення цінності. Ця тенденція бере свій початок з виробництва компанії Toyota та поширюється на різні сфери і галузі. Метою ощадливого управління проектами є зосередження на підвищенні цінності за рахунок зниження витрат в процесі управління проектами. Це досягається за допомогою пріоритезації процесів, усунення вузьких місць, постійного вдосконалення і спільного прийняття рішень.

Тенденція сталого розвитку в проектному менеджменті фокусується на зменшенні негативного впливу проектів на навколишнє середовище. Вона враховує вірогідність впливу проекту на довкілля та суспільство в довгостроковій перспективі й узгоджує цілі проекту відповідно до інтересів зовнішнього середовища. Дотримання сталого розвитку передбачає звернення уваги на три ключові сфери: економічної, екологічної та соціальної стійкостей. Економічна стійкість передбачає забезпечення фінансової стабільності проекту та економічної вигоди як для організації, так і для економіки. Під екологічною стійкістю мається на увазі зменшення негативного впливу на довкілля: усунення шкідливих викидів, вторинну переробку відходів, охорону природних ресурсів. Соціальна стійкість передбачає забезпечення статусу соціально відповідального проекту, тобто врахування впливу на суспільство, соціальну справедливість.

2. МЕТОДОЛОГІЯ УПРАВЛІННЯ ПРОЕКТАМИ SCRUM

2.1 Історія виникнення, ключові особливості та сфери застосування методології Scrum

Історія створення Scrum починається на початку 1980-х років, коли група дослідників з університету Оулу в Фінляндії почала вивчати методології розробки програмного забезпечення. Одним з цих дослідників був американський програміст на ім'я Джефф Сазерленд, який раніше служив пілотом-винищувачем у військово-повітряних силах США. Він був зацікавлений у пошуку більш ефективного способу управління проектами з розробки програмного забезпечення. Джефф черпав натхнення зі свого військового досвіду і почав розробляти новий підхід, який наголошував на співпраці, швидких ітераціях та зосередженні на створенні цінності для клієнта.

Завданням Джеффа в компанії Easel, куди його взяли на посаду віце-президента об'єктних технологій, було за шість місяців розробити нову серію продуктів програмного забезпечення для Ford Motor та інших великих компаній-клієнтів. Однак реалізувати це за допомогою каскадної моделі, яка на той час використовувалась в Easel, було майже неможливо. Кілька місяців команда розробників на чолі з Сазерлендом витратила на те, щоб знайти рішення цієї непростої задачі. Стаття з Harward Business Review «Розробка нового продукту. Нові правила гри» надихнула їх застосувати ступеневий процес розробки проекту, який був заснований на згуртуванні команди заради досягнення спільної мети, що порівнювався з грою в регбі (звідси назва - scrum). Проект був успішно реалізований на шість місяців раніше призначеного терміну в рамках визначеного бюджету, що підтвердило ефективність даної методології [15].

Перша офіційна презентація Scrum відбулася на конференції OOPSLA в 1995 році, де Джефф Сазерленд та Кен Швабер представили доповідь під назвою «Процес розробки програмного забезпечення SCRUM», яка містила в собі

ключові принципи методології. У наступні роки Сазерленд і Швабер продовжували вдосконалювати та популяризувати Scrum, публікуючи додаткові статті та книги про методологію і навчаючи все більше практиків [15].

Сьогодні Scrum є однією з найпоширеніших гнучких методологій, яка використовується в широкому спектрі галузей і дисциплін. Його орієнтація на співпрацю, постійне вдосконалення та створення цінності для клієнта зробила його популярним вибором для управління складними проектами у світі, що швидко змінюється.

Scrum має наступні ключові відмінності від інших гнучких методологій: гнучке управління проектами зосереджене на наданні цінності для замовника шляхом безперервної ітерації та вдосконалення, в той час як Scrum фокусується на розробці програмного забезпечення; методології Agile не мають визначеної структури чи рамок і можуть бути адаптованими до потреб проекту, а Scrum має чітку структуру з конкретними ролями, етапами та артефактами; в Scrum є чітко визначені ключові ролі в проекті, до яких зазвичай належать: Scrum -майстер, власник продукту та команда розробників; Scrum має специфічні артефакти, такі як відставання продукту, відставання спринту та мета спринту, які використовуються для управління проектом; гнучке управління проектами може використовувати деякі з цих артефактів, але вони не є обов’язковими; особливістю Scrum є наявність специфічних зустрічей, таких як щоденний Scrum, планування спринту, огляд спринту та ретроспектива спринту, які використовуються для управління проектом.

Загалом, якщо Scrum – конкретна методологія з певною структурою та ролями, то гнучке управління проектами – це ширший підхід, який можна адаптувати до потреб проекту. І Scrum і Agile мають спільну мету – надання цінності для замовника шляхом безперервної ітерації та вдосконалення.

Scrum може бути застосований до широкого спектру проектів та індустрій, а не лише до розробки програмного забезпечення. Окрім ІТ є приклади успішної

імплементації Scrum в інших сферах. Наприклад, є досвід використання Scrum у сфері освіти в середній школі Blueprint High School, розташованій у Чандлері, штат Арізона. В інтерв'ю, проведеному InfoQ з фахівцями Blueprint Education вони говорять, що Agile та Scrum створюють середовище для того, щоб студенти відчули, як це – по-справжньому співпрацювати, працювати разом, бути підзвітними, використовувати свою креативність, вирішувати проблеми, мислити нестандартно, самоорганізовуватися та розвивати мислення вищого рівня [16].

Інший приклад успішного застосування Scrum – сфера охорони здоров'я. Клініка психічного здоров'я Monash Health у Мельбурні (Австралія) з 2017 року запровадила гнучкі методи лікування складних захворювань. Щотижня лікарі разом обговорювали випадки хвороб своїх пацієнтів щоб побачити, що можна зробити інакше, щоб забезпечити постійний прогрес лікування. Ці зустрічі дали лікарям можливість помічати проблеми, які виникають та стали форумом для ідей, які спираються на глибину знань і досвіду всіх учасників цих зборів. Мелліса Кейсі, директор відділу психології клініки, також зазначила, що з моменту впровадження гнучких практик відбулося 46-відсоткове покращення роботи лікарів та їх задоволеності своєю роботою [17]

Проте, спроби використання та адаптації Scrum у фінансовому секторі мали не такі позитивні результати. Дослідження показують, що фінансові установи, які впровадили гнучкі методології управління стикнулися з проблемами, пов'язаними зі складністю координації роботи окремих команд, які можуть знаходитися в різних країнах, або навіть континентах. Іншою причиною, чому використання Scrum викликає сумніви є те, що в фінансовому секторі найменша помилка може спричинити втрату тисяч або мільйонів, тому операції повинні фіксуватися з надзвичайною точністю. Вимоги щодо обачності змусили багато компаній, які надають фінансові послуги залишити каскадний спосіб управління, який передбачає дотримання великої кількості нормативів та документацій [18].

Але є також приклад успішного впровадження Scrum в фінансовій сфері. Наприклад BBVA Compass розробляє 60% свого програмного забезпечення за допомогою Agile, що допомогло банку скоротити час розробки з двох років до шести місяців. Адріана Кеведо-Прайс, виконавчий директор організації Agile Projecct-based BBVA Compass, заявила, що гнучка структура буде організаційним зрушенням, яке базується на компетенціях співробітників і враховує потреби кінцевих користувачів [19].

З цих досліджень можна зробити висновки, що Scrum може бути ефективною методологією для широкого спектру проектів, однак це не найкращий вибір для кожного проекту. При прийнятті рішення, щодо впровадження Scrum варто враховувати, що: найкраще він працює в невеликих крос-функціональних командах, дозволяючи їм використовувати сильні сторони кожного члена команди і створювати якісні продукти; використання Scrum в коротких проектах з фіксованими дедлайнами може бути не найкращим рішенням, адже ітеративний процес, на якому побудований Scrum може зайняти багато часу, що неприпустимо у випадку визначених термінів завершення проекту; дана методологія вимагає високого рівня залученості клієнта, тому його краще не використовувати в проектах, де клієнт не приймає активної участі в процесі реалізації.

Загалом, запровадження Scrum поза сферою інформаційних технологій тільки зростає та є менш зрілим ніж у сфері IT. Але в усьому світі все більше галузей звертаються до його принципів, щоб стимулювати зміни на робочому місці та впроваджувати інноваційні підходи.

2.2 Процес управління проектом на основі Scrum. Основні принципи, етапи та артефакти

Емпіризм та ощадливе мислення слугують основою для визначення Scrum. Згідно з емпіризмом, рішення ґрунтуються на спостереженнях, а знання здобуваються через досвід. Ощадливе мислення усуває відходи і концентрується на тому, що має найбільше значення. Евристична за своєю природою, система Scrum побудована на постійному навчанні та адаптації до мінливих обставин. Вона визнає, що команда буде розвиватися протягом усього проекту і що вона не знатиме всього спочатку. Завдяки зміні пріоритетів, вбудованій в процес, і коротким циклам випуску, Scrum дозволяє командам автоматично адаптуватися до мінливих умов і вимог користувачів, щоб команда могла безперервно вчитися і розвиватися [21].

Структура Scrum передбачає наявність набору цінностей, принципів та практик, яких дотримується Scrum, щоб надавати продукти та послуги. Основними принципами, на яких будується фундамент методології Scrum є: прозорість, адаптація та перевірка. Кожен учасник проекту та замовник мають розуміти яким чином побудований робочий процес, які є основні цілі та очікувані результати та яких зусиль потрібно докласти щоб отримати ці результати. Ось що мається на увазі під прозорістю. Постійна перевірка допомагає вчасно виявити потенційні проблеми та забезпечити сталість процесу й узгодженість дій.

У ситуації коли виникають певні ризики або інші обставини, що впливають на якість продукту необхідно швидко прийняти необхідні рішення та вжити заходів щодо мінімізації впливу цих ризиків та зменшити відхилення від результату. Тобто, команда має бути готовою адаптуватися до цих змін. [14] Дотримуючись цих принципів команди можуть працювати більш ефективно та результативно, надаючи високоякісні продукти, які відповідають потребам клієнтів.

Скрам-команда складається з кількох ключових ролей, кожна з яких має певні обов'язки та завдання. Команди зазвичай налічують від п'яти до десяти осіб, проте за потреби можуть масштабуватися до більших розмірів. Скрам-команди характеризуються своєю крос-функціональністю, тобто кожен член команди володіє усіма необхідними навичками для створення цінності і досягнення цілі продукту. Вони є структурованими та уповноваженими організовувати та керувати власною роботою [20]. В Scrum передбачені наступні ролі: власник продукту, scrum-майстер, розробник.

Власник продукту (Product owner) – людина, яка розуміє продукт та відповідає за максимізацію його цінностей. Він зосереджений на вимогах бізнесу, забезпеченні відповідності команди розробників цим вимогам та визначенні пріоритетів в роботі команди [20]. Основними завданнями власника продукту є: дослідження ринку для розуміння потреб споживачів, встановлення пріоритетів в програмі продукту, оцінка прогресу та інформування зацікавлених сторін, перевірка роботи команди на відповідність вимогам [26].

Scrum-майстер – фахівець зі Scrum, який виконує роль фасилітатора та працює над впровадженням гнучких процесів і створення ефективного середовища для роботи команди [26]. До його обов'язків входить: ведення зустрічей, планування необхідних ресурсів та усування будь-яких перешкод на шляху команди.

Розробники – люди в Scrum-команді, які беруть участь в розробці продукту та досягненні цілей. Вони володіють навичками, необхідними для реалізації вимог продукту та є самоорганізованими в своїй роботі. Незалежно від проекту, розробники завжди несуть відповідальність за створення плану своєї роботи, забезпечення якості шляхом дотримання цілей продукту та адаптацію свого плану до мети спринту [26].

Scrum-процес розроблений таким чином, щоб бути ітеративним і гнучким, дозволяючи командам адаптуватися до мінливих вимог і надавати цінність замовнику вчасно і ефективно (рис. 2.1).

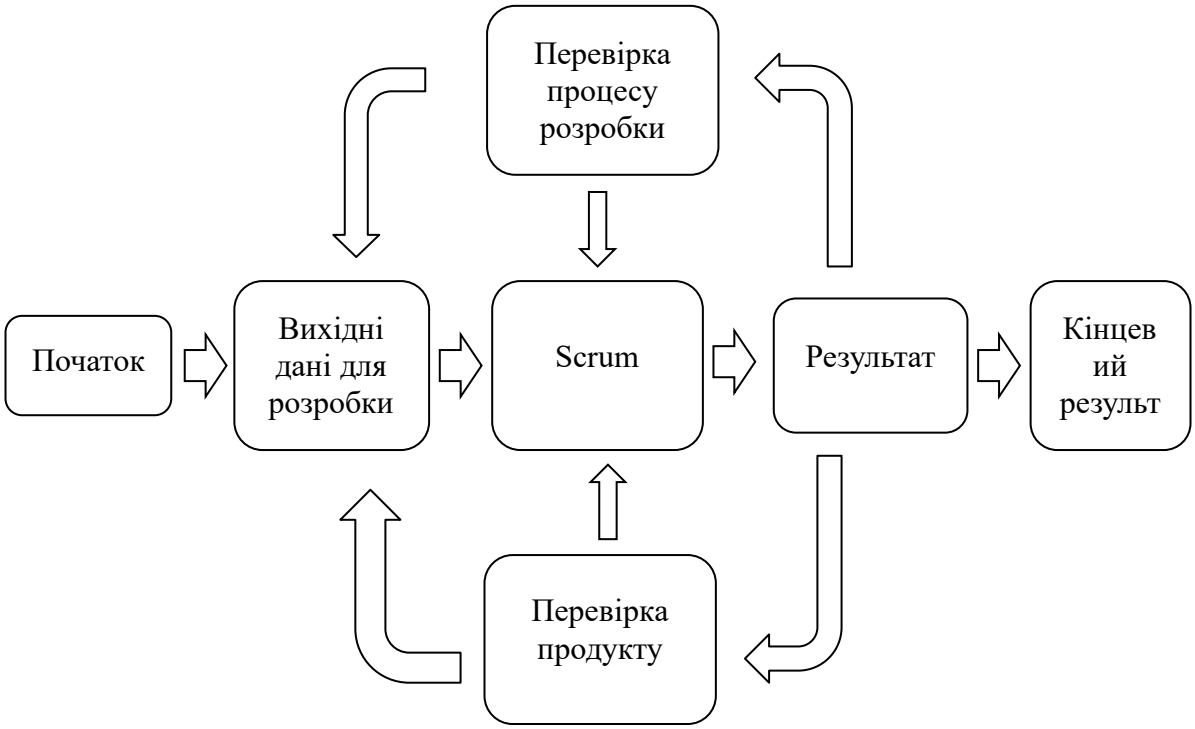


Рисунок 2.1 – Модель процесів методології Scrum [25]

Процес розробки проекту починається зі створення беклогу продукту (product backlog), який є пріоритетним переліком вимог або функцій, якими має володіти майбутній продукт та які необхідно виконати для досягнення цілей проекту. Також на даному етапі необхідно визначити пріоритетність цих вимог на основі їх важливості для проекту. Це необхідно для врахування всіх домовленостей та задоволення клієнтських очікувань [23].

Наступним етапом є планування спринту. В контексті методології Scrum, спринт (Sprint) – це обмежена в часі ітерація, під час якої команда розробників працює над створенням потенційно готового до випуску продукту. Спринти зазвичай тривають від двох до чотирьох тижнів і їхня тривалість визначається командою на основі її досвіду та складності роботи [24]. На початку кожного спринту команда зустрічається на нараді з планування спринту (Sprint Planning)

щоб обговорити завдання, які потрібно виконати протягом спринту і створити беклог спринту (рис. 2.2).

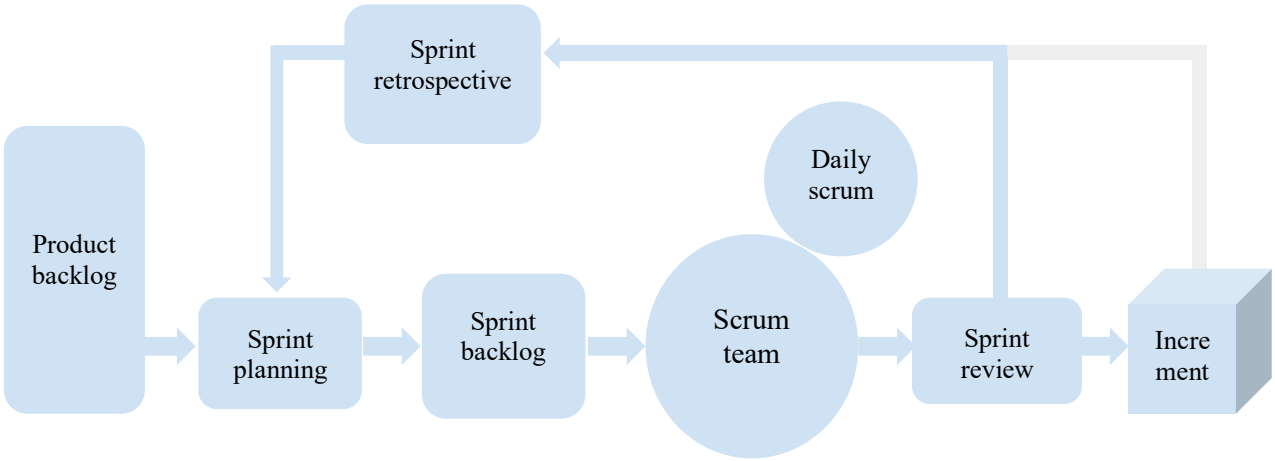


Рисунок 2.2 - Процес реалізації проекту за методологією Scrum [22]

Беклог спринту (Sprint backlog) – перелік задач, обраних командою розробників для впровадження в поточному циклі спринту [21]. Задачі розподіляються між членами команди і мають бути виконані до кінця циклу. Головною умовою є те, що після початку роботи над задачами, нові задачі вже не можуть додаватися в спринт. Команда працює над цими завданнями протягом усього спринту, проводячи щоденні скрам-зустрічі для моніторингу процесу та виявлення будь-яких проблем, які потребують вирішення.

Щоденний скрам (Daily scrum) – щоденна коротка нарада, яка відбувається в один і той же час, метою якої є перевірка прогресу у досягненні цілей спринту та коригування плану роботи на поточний день. Щоденний скрам триває не більше 15 хвилин, за які кожен член команди має відповісти на три ключові запитання: «що я робив учора?», «що я планую робити сьогодні?», «чи є якісь перешкоди на моєму шляху?» [26].

Наприкінці спринту команда збирається на оглядову зустріч (Sprint review) щоб продемонструвати завершену роботу зацікавленим сторонам та отримати зворотній зв'язок. Команда також проводить ретроспективну зустріч (Sprint retrospective), щоб проаналізувати свою роботу під час спринту та визначити сфери покращення в наступному спринті. Ретроспектива є важливою частиною

Scrum, оскільки дозволяє команді постійно вдосконалювати свої процеси і з часом ставати більш ефективною. До основних питань, які мають бути обговорені протягом цієї зустрічі належать [26]:

- Які покращення команда може запровадити в свою роботу?
- Що під час реалізації останнього спринту пройшло добре?
- Що можна покращити в наступному спринті?

Результатом кожного спринту має бути розроблена самостійна частинка готового продукту – інкремент. Кожен наступний інкремент є доповненням до всіх попередніх інкрементів і проходить ретельну перевірку для того, що приносити користь і цінність клієнту. Протягом спринту може бути розроблено один або декілька інкрементів, які представляються клієнту та іншим стейкхолдерам на Sprint Review [20].

Далі ці етапи повторюються кожного наступного спринту, утворюючи цикл і постійно створюючи цінність для замовника та адаптуючись до мінливих вимог.

Тепер варто звернути увагу на ще один важливий компонент Scrum – інструменти, які використовуються для планування та організації роботи команди і покращення ефективності. До них належать: Scrum-дошка, діаграма вигоряння задач.

Scrum-дошка є візуальним представленням прогресу проекту і являє собою таблицю, зі п’ятьма (іноді більше) стовпцями: «Зробити», «Виконується», «Переглядається», «Завершено». На початку спринту, всі картки з завданнями поміщаються на дошку «Зробити». Розробник, беручи завдання в роботу, переміщає його на дошку «Виконується», а для решти команди це означає, що над завданням вже працюють. Після завершення завдання, картка переміщується в колонку «Переглядається», що означає, що завдання буде переглянуто керівником розробників на відповідність вимогам та наявність помилок. І лише після цього картка переміщується на дошку «Завершено» [27].

Діаграма вигоряння задач (Burndown chart) – показує швидкість роботи команди, обсяг зробленої роботи та загальної роботи, що залишилася. Діаграма вигоряння є корисним інструментом, оскільки дозволяє оцінити ймовірність завершення роботи командою за відведений час (рис. 2.3) [27]. Сіра лінія на рисунку відображає очікувану швидкість роботи команди, у випадку відсутності затримок. Червона лінія відображає фактичний прогрес команди та кількість роботи, яка залишилась. В ідеальному варіанті червона лінія має знаходитись під сірою – це означає, що команда на правильному шляху і встигне завершити всю роботу до закінчення спринту.

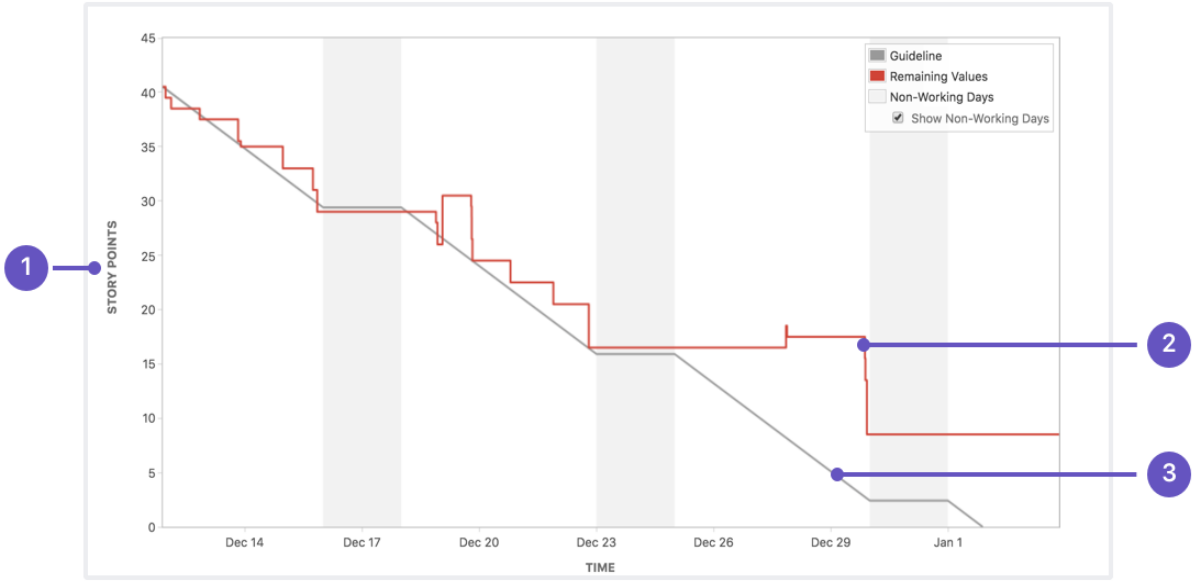


Рисунок 2.3 – Діаграма вигоряння задач [27]

2.3 Аналіз досліджень щодо результатів застосування методології скрам в IT-проектах

Наукове дослідження Університету прикладних наук Коблеца про переваги і фактори успіху гнучких методів розробки IT-продуктів показало, що Agile є найбільш поширеним методом розробки у сфері програмного забезпечення. При цьому частка Scrum становить аж 84%. До причин такого поширеного застосування гнучких методологій відносять якість, скорочення термінів та зниження ризиків. Результати дослідження свідчать про те, що застосування

гнучких підходів призвело до покращення результатів та підвищення ефективності проектів (рис. 2.4) [29].

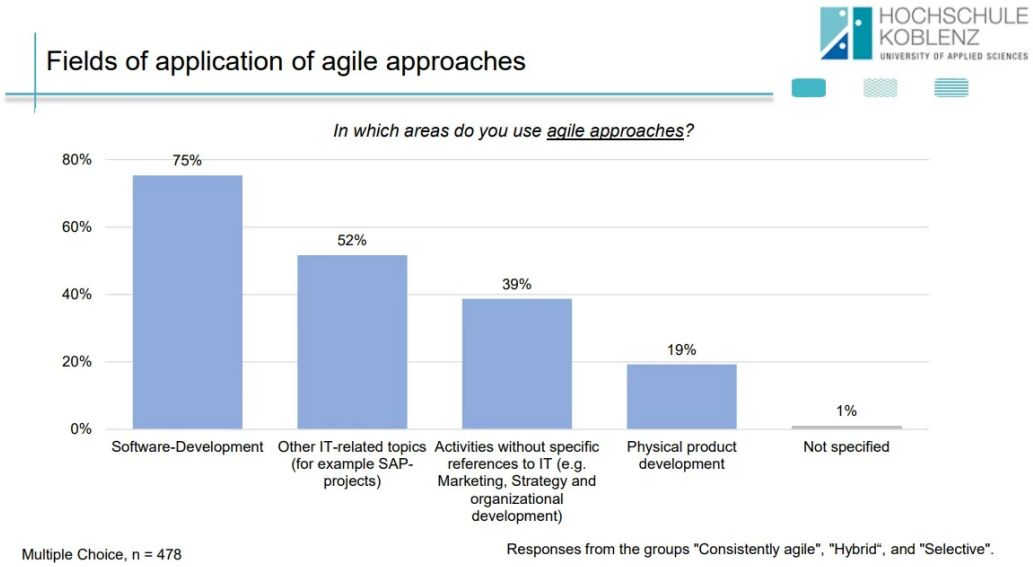


Рисунок 2.4 – Сфери застосування гнучких підходів [29]

Опитування State of Agile 2020 року визначило основні причини застосування гнучких методологій в IT-компаніях. До них належать: посилення співпраці, краща відповідність потребам бізнесу, покращення робочого середовища, збільшення прозорості процесів, швидше реагування на загрози, покращення користувацького досвіду, покращення обслуговування клієнта, підвищення рівня впровадження продукту [30].

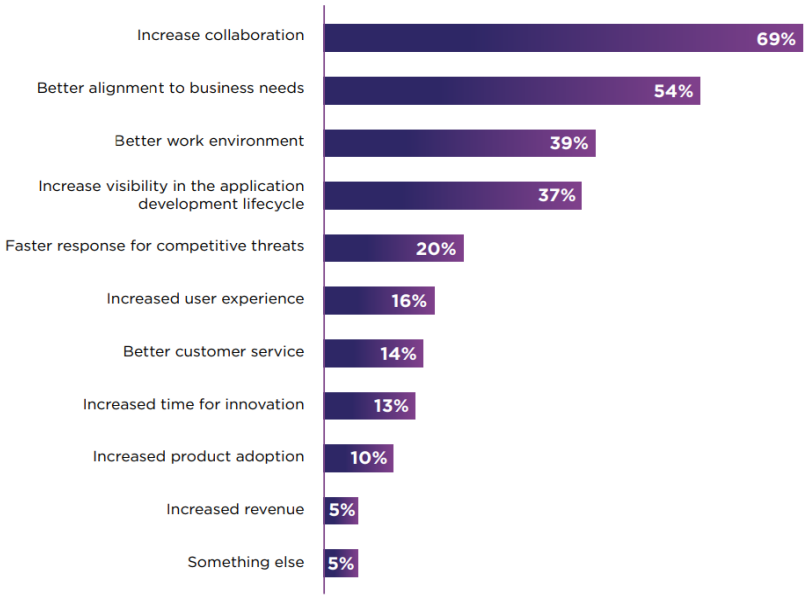


Рис. 2.3 – Результати опитування щодо причин застосування Agile в IT-компаніях [30]

Ці дослідження показують, що Scrum є найбільш популярним фреймворком для управління проектами в IT-компаніях. Вони також визначають перелік переваг застосування Scrum, що ще більше спонукає інші компанії застосовувати Scrum в роботі своїх команд. Однак варто звернути увагу і на більш детальні дослідження, які аналізують як Scrum змінюється на практиці і як ці зміни можуть впливати на результати проектів.

У дослідженні, щодо ефективності застосування Scrum в розробці програмного забезпечення проведеному дослідниками Гавайського університету в Маноа, взяло участь 11 різних компаній, які використовували Scrum в реалізації своїх проектів. Респонденти повідомили, що перша проблема, з якою стикнулися команди розробників при впровадження Scrum - процесів та цінностей – потреба в тренінгах, які б навчали команду працювати за Scrum. Команди не завжди розуміли суть всіх процесів та їх важливість, тому не цінували їх. По-друге було виявлено залежність між здатністю команди планувати спринт і досягненням якості продукту. Команди мають бути достатньо самостійними, про що говорить документація Scrum, однак дослідження показали, що відсутність контролю може призвести до зниження відданості команди та низької продуктивності [31].

По-третє було виявлено позитивний вплив ретроспективних зустрічей, які проводились з метою перевірки та вдосконалення. Однак, не всі з опитуваних компаній давали позитивний відгук про даний тип зустрічей – в деяких командах ці зустрічі були не ефективними і використовувалися лише як спосіб виплеснути скарги, а не покращити роботу. В дослідженні також говорилось, що найбільш складним аспектом Scrum було дотримання самоорганізованих автономних команд [31]. Загалом, дослідження показало, що дотримання всіх принципів та артефактів Scrum не гарантує успішного результату та отримання переваг, які він пропонує. Значно важливішим є розуміння цілей впровадження цих практик і використання основних цінностей.

Спираючись на ці та інші дослідження, можна зробити висновки, що застосування методології Scrum в IT-проектах має як переваги, так і недоліки. Отримати максимум користі від використання даного фреймворку можливо лише за умови попереднього навчання основним процесам та глибокому розумінні і застосуванні основних цінностей та принципів, які пропонує Scrum.

3. УДОСКОНАЛЕННЯ ПРОЦЕСУ УПРАВЛІННЯ ІТ-ПРОЕКТОМ ЗА МЕТОДОЛОГІЄЮ SCRUM

3.1Процес впровадження Scrum-фреймворку в проект

Дослідження ефективності методології Scrum проводилося на прикладі стартап-проекту з розробки додатку для подорожей. Дослідження зосереджено на проектній команді, яка використовувала Scrum в процесі реалізації проекту. Теоретичні дані надають базове уявлення про методологію, основні процеси та підходи, а дане дослідження присвячене практичному застосуванню методології та виявленню факторів успішності і основних проблем, з якими стикається команда проекту та способах їх вирішення. Дані збиралися шляхом спостереження та інтерв'ю.

Процес впровадження методології Scrum в проект включав наступні етапи:

1) Оцінка проекту
2) Формування команди
3) Створення продуктового беклогу
4) Планування спринту
5) Реалізація
6) Огляд та ретроспектива спринту

Оцінка проекту. Оцінка проекту є важливим етапом реалізації проекту, який включає в себе оцінку цілей, вимог, характеристик та очікувань від проекту щоб визначити доцільність його впровадження за допомогою методології Scrum.

В першу чергу було проаналізовано мету проекту та поставлені основні цілі проекту. Метою проекту є розробка додатку для подорожей, який надаватиме користувачам можливість планувати подорожі, слідкувати за витратами та документувати свій досвід. Основні цілі проекту включають:

- розробити веб-додаток з усім необхідним функціоналом до кінця серпня;
- покрити юніт-тестами щонайменше 80% функціоналу;
- запустити проект до кінця осені;
- отримати перших 100 користувачів до кінця 2023 року.

На основі цілей були сформовані вимоги до проекту, що включали в себе як бізнес-вимоги, так і технічні вимоги. Нижче наведено загальні вимоги, яким має відповідати продукт, щоб цілі проекту були досягнуті:

- наявність функції створення плану подорожі;
- наявність готових путівників для подорожей;
- можливість сортування подорожей за категоріями для зручного пошуку;
- наявність функції ведення обліку витрат.

До основних проектних обмежень належить бюджет та команда проекту. Відсутність інвестицій на початку проекту значно обмежує витрати на оплату сервісів, необхідних для реалізації функціоналу та працю розробників. Тому під час реалізації проекту необхідно враховувати дані обмеження.

Важливим пунктом на етапі оцінки проекту був аналіз ризиків та створення плану з управління цими ризиками (рис. 3.1).

ID	Risk	Impact description	Impact	Probability	Risk score	Strategy	Response	Responsible
1	Extension of the scope of work because some intermediate tasks were not included in the scope	Failure to meet the project schedule	3 - High	2 - Medium	6	mitigate	Creation of a detailed roadmap of the project with a breakdown into tasks	PM
2	Team members do not have the necessary technical qualifications to implement certain functionality	low-quality functionality, bugs in the code	2 - Medium	2 - Medium	4	mitigate	Team training	PM, TL
3	Risk of problems with software tools and platforms used by the development team	delaying the development process, wasting resources to solve these problems	2 - Medium	2 - Medium	4	reject	allocating time to solve problems	PM, TL
4	Poor communication or lack of coordination between team members, developers do not get in touch	low efficiency of the team, they do not understand their tasks	3 - High	2 - Medium	6	mitigate	Arranging periodic meetings with the team at a fixed time, maintaining constant communication	PM
5	Risk of reduced productivity due to lack of motivation or burnout	delay of deadlines	2 - Medium	2 - Medium	4	mitigate	Give developers interesting tasks, involve them in the decision-making process, 1:1 meetings	PM

Рис 3.1 - План з управління ризиками

План з управління ризиками допоміг виявити, оцінити та визначити пріоритетність ризиків, які можуть вплинути на досягнення цілей проекту. За допомогою аналізу було виявлено основні можливі ризики та розроблено стратегії пом'якшення і уникнення цих ризиків до того, як вони стануть реальними проблемами.

Оскільки в даному проекті є низький рівень втрати значної суми бюджету, відсутні суворі дедлайни реалізації проекту та низький рівень виникнення зовнішніх ризиків проект може бути реалізовано за допомогою методології Scrum.

Формування команди. В рамках методології Scrum, формування команди має вирішальне значення для успішної реалізації проекту. Команда повинна складатися з людей, які володіють ключовими навичками, досвідом та знаннями предметної області, а також гнучкістю та здатністю до самоорганізації.

Для реалізації даного проекту була зібрана команда у складі 5 осіб: два бекенд-розробники, один фронтенд розробник, дизайнер та Scrum-майстер, що поєднує у собі також роль власника продукту. Наступним кроком було визначено ключові ролі та обов’язки кожного члена команди. Дані занесено в таблицю 3.1

Ім’я	Роль в проекті	Обов’язки
Марія Д.	Scrum майстер	Особа, яка здійснює щоденне управління проектом і несе конкретну відповідальність за управління проектом в рамках затверджених обмежень за обсягом, якістю, часом часу та вартості, щоб забезпечити зазначені вимоги та цілі до проекту і досягти визначених результатів;
Дмитро Б.	Бекенд розробник	Розробник, який відповідає за забезпечення бекенд-частини проекту, підтримку інших

		розробників, а також затвердження та перевірку коду;
Данило П.	Бекенд розробник	Розробка бекенд-частини проекту згідно поставлених термінів та вимог;
Захар О.	Фронтент розробник	Розробка інтерфейсної частини додатку на основі дизайну, імплементація необхідного функціоналу;
Марія М.	Дизайнер	Розробка дизайну всіх частин додатку згідно поставлених вимог.

Таблиця 3.1 – Ключові ролі та їх обов’язки

Створення продуктового беклогу. Беклог продукту – перелік всіх завдань, які необхідно виконати Scrum-команді, щоб вважати проект реалізованим. Беклог продукту формується на основі вимог та історій користувачів. Історії користувачів є способом фіксації вимог у проекті, які мають певний формат і описують функціональність, якою має володіти продукт з точки зору користувача. Історії користувачів також необхідні для чіткого розуміння продукту та потреб аудиторії, які має задовольняти додаток [33]. Приклад створеної історії користувача зображено на рис 3.2.

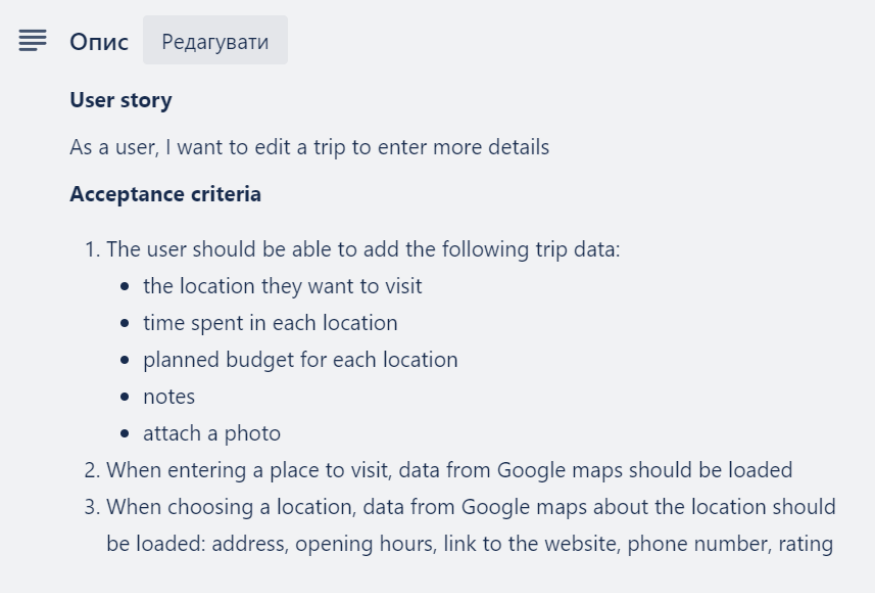


Рисунок 3.2 – Приклад історії користувача

На основі історій користувачів були створені задачі, які сформували беклог продукту (рис. 3.3)

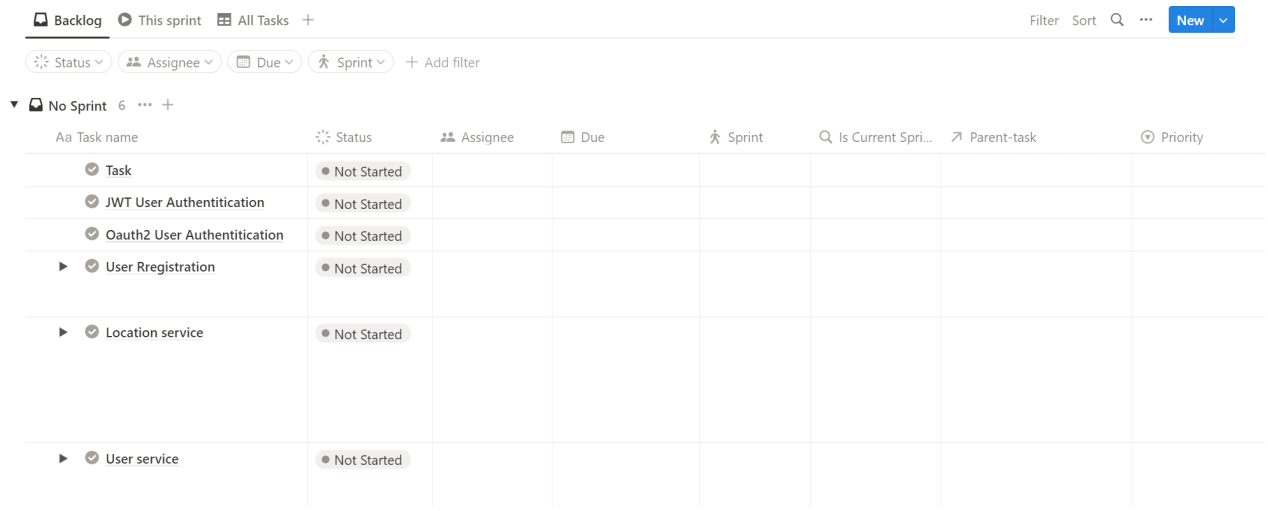
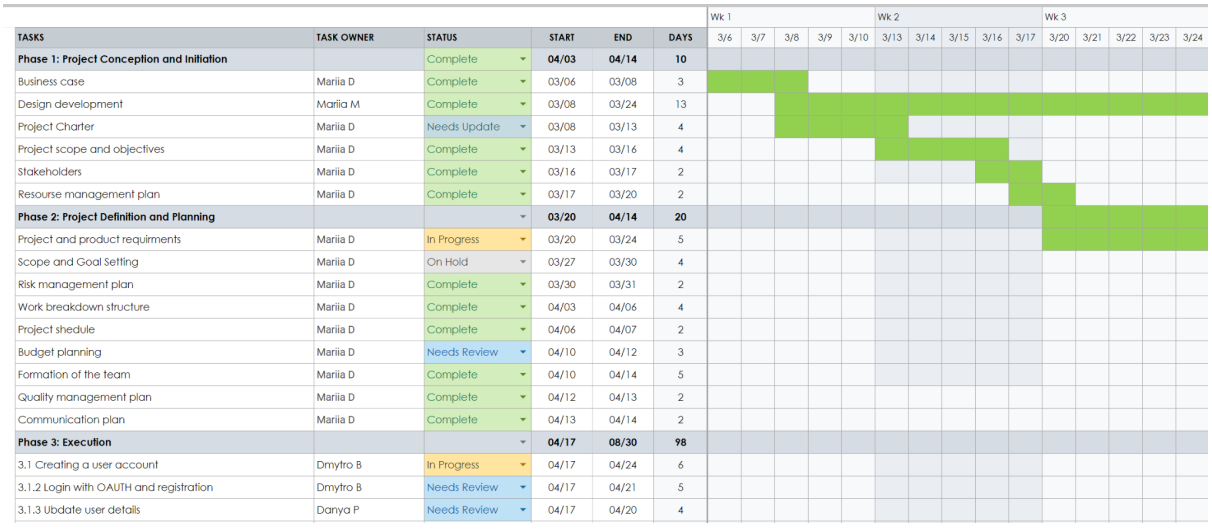


Рисунок 3.3 – Беклог продукту

Наступним кроком була побудована діаграма Ганта, в якій відображено ключові етапи реалізації проекту (рис. 3.4).



TASKS	TASK OWNER	STATUS	START	END	DAYS
Phase 1: Project Conception and Initiation		Complete	**04/03**	**04/14**	**10**
Business case	Mariia D	Complete	03/06	03/08	3
Design development	Mariia M	Complete	03/08	03/24	13
Project Charter	Mariia D	Needs Update	03/08	03/13	4
Project scope and objectives	Mariia D	Complete	03/13	03/16	4
Stakeholders	Mariia D	Complete	03/16	03/17	2
Resourse management plan	Mariia D	Complete	03/17	03/20	2
Phase 2: Project Definition and Planning			**03/20**	**04/14**	**20**
Project and product requirments	Mariia D	In Progress	03/20	03/24	5
Scope and Goal Setting	Mariia D	On Hold	03/27	03/30	4
Risk management plan	Mariia D	Complete	03/30	03/31	2
Work breakdown structure	Mariia D	Complete	04/03	04/06	4
Project shedule	Mariia D	Complete	04/06	04/07	2
Budget planning	Mariia D	Needs Review	04/10	04/12	3
Formation of the team	Mariia D	Complete	04/10	04/14	5
Quality management plan	Mariia D	Complete	04/12	04/13	2
Communication plan	Mariia D	Complete	04/13	04/14	2
Phase 3: Execution			**04/17**	**08/30**	**98**
3.1 Creating a user account	Dmytro B	In Progress	04/17	04/24	6
3.1.2 Login with OAUTH and registration	Dmytro B	Needs Review	04/17	04/21	5
3.1.3 Ubdate user details	Danya P	Needs Review	04/17	04/20	4

Рисунок 3.4 - Діаграма Ганта

Діаграма Ганта допомогла визначити ключові етапи розробки проекту та їх тривалість, що дозволяє оцінити необхідні зусилля для успішного завершення проекту.

Планування спринту. На даному етапі проект готовий до початку реалізації. Тому наступний крок - знайомство команди з проектом та методологією Scrum. В якості знайомства була проведена зустріч kick-off (зустріч перед початком проекту), на якій менеджер проекту представив ідею проекту, бізнес кейс, основні вимоги та задачі. На цій також команда ознайомилась з методологією Scrum, структурою роботи та основними мітингами.

Після чого була проведена перша зустріч з планування спринту (sprint planning), на якій було оцінено задачі з беклогу та сформовано беклог спринту - перелік задач на перший спринт, призначено кінцеві терміни та відповідальні особи.

Для визначення кількості задач, яку команда може реалізувати за спринт використовувалася техніка story points. Story points – це техніка оцінки, яка використовується в гнучких методологіях для оцінки зусиль, необхідних команді для досягнення бажаного результату [34]. Сюжетні точки це умовні одиниці вимірювання, які враховують складність задачі, повторюваність та ризикованість. Для визначення сюжетних точок була використана послідовність чисел Фібоначчі: 0, 0.5, 1, 2, 3, 5, 8, 13, 20… . Більше число означає більшу кількість зусиль, необхідну на виконання задачі. Дана техніка допомагає не лише оцінювати кожну задачу окремо, а й одну відносно іншої.

Реалізація. Після формування беклогу спринту команда приступила до виконання задач. Для візуалізації процесів була використана Kanban дошка, яка допомагає покращити розуміння того, які задачі необхідно виконати, які задачі на даний момент перебувають в роботі та в який момент кожна задача буде завершеною.

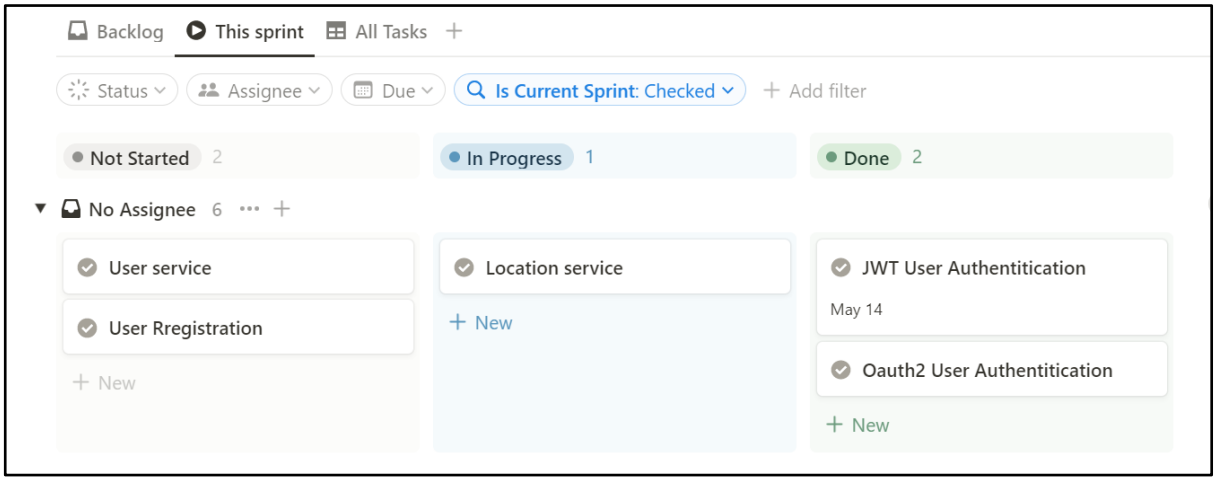


Рис. 5 - Kanban дошка проекту

Огляд та ретроспектива спринту. Наприкінці кожного спринту проводилась зустріч з огляду спринту (sprint retrospective), на якій команда аналізувала результати своєї роботи та демонструвала готові частини продукту.

3.2 Аналіз ефективності методології Scrum для управління IT-проектами на прикладі TripTide

Основною метою дослідження було проаналізувати ефективність Scrum-фреймворку для управління невеликою командою розробників IT-продукту. Дані дослідження були зібрані протягом двох тижнів роботи команди у спринтах. Слід зазначити, що проект не є завершеним, а перебуває на стадії розробки, тому ці дані не можуть свідчити про його успішність. Особливу увагу під час дослідження було приділено проблемам та викликам, з якими стикалася команда при застосуванні Scrum, а також аналізу ефективності її роботи за використанням основних принципів та процесів методології Scrum. Наприкінці спринту було проведено опитування команди щодо їхнього враження та зроблено відповідні висновки.

Впровадження Scrum на перших етапах викликало зацікавлення в кожного члена команди, оскільки більшість з них не була раніше знайома з даною методологією. Вони з цікавістю приймали участь в плануванні спринту, стендапах та ретроспективах. Однак в процесі реалізації проекту команда стикнулася з певними проблемами, які були пов'язані з нерозумінням правильності оцінки задач, розтягуванням зустрічей та відсутністю розуміння кінцевого продукту. Ці виклики стали причинами зниження ефективності команди і досягнення нею очікуваних результатів. Детальніше ці проблеми буде розглянуто нижче. Для аналізу успішності команди були використані наступні фактори: комунікація, мотивація, продуктивність, залученість команди.

Комунікація. Застосування Scrum-зустрічей та підтримання прозорості процесів мало позитивний вплив на роботу команди. Учасники команди не боялись обговорювати проблеми, які виникали у них на шляху та пропонували ідеї щодо покращення результатів роботи та продукту. Наявність великої кількості зустрічей сприяло згуртованості команди над однією метою і покращення зворотного зв'язку. На такий результат також вплинула наявність чіткої стратегії комунікації та визначені канали зв'язку – команда знала яким чином та в який час можна зв'язатися з кожним розробником і користувалася цим.

Мотивація. Використання Scrum мало як позитивний так і негативний вплив на мотивацію команди. З одного боку, прозорість, відкритість та постійна комунікація мотивувала команду проявляти себе та досягати результатів. З іншого боку, відсутність чіткого розуміння кінцевого продукту і постійне додавання нових задач в беклог мало не дуже позитивний ефект, оскільки фокусування лише на задачах дає неповне знання цінності, яку приносить кожний член команди, а поява нових задач викликає тривожність і страх не встигнути до дедлайну.

Продуктивність. Наявність структурованого плану та чітко визначеного списку задач безумовно сприяло покращенню продуктивності команди. Проте, потреба у високому рівні самоорганізованості та відсутності контролю вплинуло

на те, як розробники керували своїм часом і розподіляли навантаження. Оскільки задачі в беклозі спринту містили одну кінцеву дату – без урахування перевірки та код-ревю, розробники виконували задачі під кінець дедлайну, що зменшувало кількість часу на перевірку та затвердження коду керівником команди. Через це, зростала завантаженість керівника і знижувалась якість перевірки, що вплинуло на якість коду. До того ж, рівень самоорганізації команди можна охарактеризувати як доволі низький – вони потребували постійного нагляду менеджера і керівника команди.

Залученість команди. Рівень залученості команди був достатньо високий, враховуючи кількість часу, яку кожний розробник приділяв процесам, які вимагали його участі. Розробники активно обговорювали питання які виникали, пропонували свої ідеї для покращення процесів і були зацікавлені в спільному вирішенні проблем. Однак, оскільки кожен член команди був частково зайнятий в інших проектах – рівень їх залученості з часом падав, що впливало й на загальну продуктивність.

Переваги. Аналізуючи дані результати, можна зробити висновки, що переваги використання Scrum в даному проекті полягали в наступному:

1. Структурованість. Створення структурованого чіткого плану роботи значно полегшує роботу команди і вивільняє час, необхідний на організацію. Таким чином, команда більше фокусувалася на задачах, аніж на визначенні пріоритетів та дискусій з приводу того, що саме вона зараз має робити. За рахунок цього підвищилася продуктивність роботи команди, оскільки цілі та задачі були визначені і приорітизовані;
2. Прозорість процесів. Кожен член команди розумів чим займаються його колеги, які задачі зараз перебувають в роботі та на якому етапі знаходиться команда на шляху до виконання всіх запланованих завдань. Окрім цього вся документація по проекту була у відкритому доступі, де розробники могли знайти необхідну інформацію по проекту;

3. Постійне вдосконалення. Проведення ретроспективи сприяло постійному аналізу процесів та пошуку речей, які можуть бути покращеними. Команда самостійно аналізувала свою роботу, визначала фактори, які перешкоджали ефективній роботі та впроваджувала заходи щодо покращення;
4. Спільне навчання. Поєднання в команді спеціалістів різних рівнів кваліфікації та напрямків і відкрите спілкування в рамках вирішення задач сприяло передачі навиків та компетенцій і спільному вивченню нових технологій. За рахунок цього, команда удосконалювала свої навики та ростила сильних гравців всередині проекту;
5. Швидкі результати. Наявність ітерацій дає можливість отримувати невеличкий працюючий продукт вкінці кожного спринту, що впливає на швидкість розробки – команда мусить виконати всі задачі, аби представити готову частинку роботи, а також на мотивацію команди, оскільки викликає відчуття досягнення результатів.

Проблеми та виклики. Першою проблемою було нерозуміння кінцевого результату у кожного члена команди та відсутність передбачуваності. Методологія Scrum побудована таким чином, що вимоги до проекту постійно змінюються, як наслідок передбачити кінцевий результат проекту може бути складно. З одного боку це дозволяє постійно вносити нові зміни в існуючі процеси без потреби повної перебудови, а з іншого боку виникає проблема відсутності конкретики та ризик виконання непотрібних завдань. Проте ключовою є проблема нерозуміння кінцевого результату проекту, а оскільки саме керівництво не до кінця розуміє очікувані результати проекту команді складніше орієнтуватися в них. Команді не вистачило розуміння власне кінцевого продукту, який має бути розробленим, а концентрація лише на задачах завадила погляду з боку, що вплинуло на ефективність її роботи.

Наступною проблемою, з якою стикнулася команда проекту і яка є поширеною серед інших проектів - нерозуміння ключових процесів через брак досвіду роботи з методологією. Scrum передбачає дотримання певних правил,

наявність частих зустрічей, прозорість та постійну взаємодію і тільки в цьому випадку досягається максимальна ефективність. Без досвіду роботи за методологією команда може не розуміти всі інструменти та важливість їх використання. У команди виникали питання щодо правильності оцінювання задач, визначення оптимальної кількості задач на спринт, та необхідності проведення великої кількості зустрічей. Для досягнення ефективності розуміння цінностей Scrum має бути однаковим у всієї команди.

Найскладнішою частиною роботи була оцінка задач за допомогою story points. У кожного члена команди формувалося власне уявлення щодо абстрактності оцінки і правильності підбору величини оцінки відносно складності задачі, тому виникали великі розбіжності в оцінці задач. З метою покращення розуміння оцінювання команда детально обговорювала кожну задачу та зусилля, необхідні для її виконання. Однак існували задачі, які були розпливчасті або невідомі для них і розробникам було складно оцінити кількість часу необхідного на виконання задач з дотриманням всіх критеріїв прийняття.

Методологія Scrum спирається на ітеративний процес реалізації проекту та наявність інкременту - готового продукту наприкінці кожної ітерації. Через це оцінити роботу та встановити реалістичні очікування може бути важко, а оособливо на початку проекту, коли вимоги є не повністю визначеними та можуть змінюватися. Ця проблема посилюється, коли у команди є брак досвіду роботи з певними технологіями, що використовуються в роботі, і вони не можуть об'єктивно оцінити складність задач та час, необхідний на впровадження того чи іншого функціоналу.

Робота за Scrum передбачає високий рівень гнучкості, адже в проект вносяться постійні зміни, які можуть критично впливати на те, що саме буде розроблятися та в який спосіб. Через це, команда має бути постійно готовою до будь-яких змін. Це також є проблемою, оскільки у більшості випадків люди

звикають до певного способу роботи та взаємодії і їм може бути важко переналаштуватися, тому вони будуть чинити опір цим змінам.

Наступною проблемою є потреба у високому рівні самоорганізації команди. Scrum-команди побудовані таким чином, що вони є самодостатніми і не потребують контролю з боку керівництва, а точніше, у Scrum навіть не передбачене таке поняття як керівництво. Ролью Scrum-майстра є організовувати команду і направляти її, функцію ж контролю команда залишає за собою, самостійно визначаючи час роботи, кількість задач, яку вони зможуть виконати за ітерацію та спосіб, в який вони ці задачі будуть виконувати. З одного боку це розвиває в команди почуття відповідальності за кінцевий результат, а з іншого - брак контролю та координації може призвести до неефективності та затримки в роботі. Розробники можуть відкладати завдання, плануючи їх виконання в кінцевий термін, що впливає на якість коду та неможливість своєчасного вирішення проблем у разі їх виникнення.

Одним з ключових принципів, на яких будується Scrum є прозорість, тобто кожен учасник команди розуміє всі етапи розробки, разом з іншими розв'язує проблеми та приймає участь в прийнятті рішень та внесенні змін в процес роботи. Однак іноді деякі ідеї запропоновані учасниками команди можуть суперечити загальному баченню проекту у інших учасників, через що виникає конфлікт інтересів, а відсутність чіткої ієрархії того, за ким залишається прийняття кінцевого рішення провокує виникнення суперечок та затримки в роботі.

Ще однією проблемою, з якою стикнулася команда - це велика кількість зустрічей. Більшості розробників цікавіше працювати з технологіями і їм може бути некомфортно або нудно від великої кількості зустрічей, тому їх було важко залучати в процес прийняття рішень, що теж негативно вплинуло на якість цих зустрічей.

Аналіз результатів. Команда є ключовим фактором успіху проекту в Scrum. Вона має бути повністю віддана проекту і вкладати всі ресурси і потенціал в його реалізацію. Ця вимога прописана в документації Scrum та згадується в багатьох дослідженнях [20]. Спостереження з командою проекту показали, що ця вимога не була виконана, оскільки розробники були лише частково зайняті в проекті і одночасно працювали на інших проектах. Без відданості проекту команда не могла дотримуватися всіх принципів Scrum і виконувати свої ролі.

Не менш важливим фактором успішності Scrum є гнучкість та відкритість до змін. Важливо щоб кожен член команди був готовий змінюватися і адаптуватися до постійно змінюваних вимог, процесів і обов'язків. Для більшості команди гнучкість була скоріше викликом аніж притаманною рисою. Це питання скоріше належить до трансформації на рівні особистості, адже сучасний світ потребує від суспільства високого рівня гнучкості та адаптації. Однак на рівні команди та проекту, кожен учасник має пройти попередню підготовку і навчання для того, щоб приносити цінність в перспективі.

Кожний учасник команди має свою роль і перелік обов'язків, тому проаналізуємо кожну з ключових ролей команди даного проекту: власник продукту, Scrum майстер, розробник.

Власник продукту повинен мати чітке бачення продукту та кінцевого результату, розуміти ключові характеристики та властивості, якими має володіти продукт, формувати вимоги для команди проекту та приймати рішення щодо реалізації продукту. Лише власник продукту володіє повною інформацією про продукт і має донести її до команди. Без належного керівництва з боку власника продукту команда відхиляється від очікувань і не досягає поставленого результату. Окрім того, роль власника продукту не повинна поєднуватися з іншою роллю в команді. Якщо проаналізувати роль власника продукту в досліджуваному проекті на основі відгуків членів команди, то варто зазначити, що йому бракувало чіткого розуміння продукту і його ключових характеристик.

Вимоги до продукту були сформовані не чітко та в команди виникали труднощі з пріоритезацією.

Scrum майстер є в першу чергу фасилітатором зі Scrum, який розуміє цінності, принципи та артефакти Scrum і доносить їх команді. Роль Scrum майстра в досліджуваному проекті була слабо вираженою, оскільки він володів недостатнім досвідом і знаннями та міг неправильно інтерпретувати цінності Scrum. Scrum майстер до цього не брав участі в реальних проектах, тому у нього виникало багато труднощів з організацією процесів та комунікацією з командою.

Оскільки Scrum-команди є самоорганізованими, вони не потребують менеджера, який буде керувати їхньою роботою. Однак, в реальному проекті цей принцип був порушений тому, що недосвідчена команда потребувала певного менеджменту та координації. Оскільки Scrum майстер повинен був поєднувати в собі роль менеджера проекту та частково власника продукту, деякі Scrum процеси були видозмінені або опущені, що також вплинуло на результати роботи.

Кожний член команди розробників повинен володіти певним рівнем усвідомленості та готовності до відкритості, активності та адаптації. Цінності Scrum передбачають високий рівень самоорганізації і віддачі, тому кожен повинен бути зацікавлений в досягненні результату. Окрім цього, вони мають бути готові до здобуття нових знань і навичок, окрім тих, якими вони вже володіють, адже постійна зміна вимог та масштабування проекту вимагають цього.

Якщо аналізувати команду розробників досліджуваного проекту, то вони були зацікавлені в досягненні спільного результату та створенні цікавого продукту. В більшості команди була присутня риса відкритості та проактивності, вони прагнули здобувати нові навички в межах проекту та удосконалювати свої вміння для подальшого застосування. Але не кожен був готовий адаптуватися до постійних змін і брати на себе повну відповідальність за результат, вони покладалися на інших. Тому отримані результати були нижче очікуваних.

Висновки. Застосування методології Scrum, як і будь-якої іншої методології управління проектами вимагає певного рівня знань і підготовки. Це стосується не лише фасилітатора Scrum, а й всієї команди, яка займається розробкою проекту. Більшість проблем, з якими стикалася досліджувана команда були пов’язані з браком досвіду та навичок роботи з методологією. Високий рівень мотивації та зацікавленості в проекті, самоорганізованість та готовність до змін – це ті якості, якими має володіти Scrum-команда, щоб досягти успіху в проекті. Успіх проекту також визначає ступінь участі зацікавлених сторін – в першу чергу клієнта та власника продукту, оскільки вони єдині хто має повне розуміння кінцевого результату.

Хоча це дослідження містить важливі висновки, воно має деякі обмеження. Перш за все, проект не було завершено, тому зібрані дані можуть свідчити лише про попередні результати і мають певний відсоток похибки, оскільки на момент дослідження, впровадження Scrum в проект перебувало на початкових етапах.

Другим обмеженням є розмір команди, яка складалася з п’яти осіб включно з дослідником. Вузька вибірка може впливати на суб’єктивність суджень опитуваних осіб, однак вона все ж дає деякі важливі факти. По-третє, обмежена кількість спринтів не надає повного уявлення про ефективність застосування методології, оскільки процес адаптації команди відбувається значно довше.

Незважаючи на дані обмеження, дослідження все ж має важливі висновки, які розкривають проблемні аспекти практичного застосування Scrum та впливатимуть на подальшу успішність проекту. В подальшому планується проведення спостереження та використання отриманого досвіду для успішної реалізації проекту.

3.3Розробка способів підвищення ефективності впровадження Scrum

Традиційно, проект вважався успішним якщо задовольняв три складові проектного трикутника: вартість, обсяг, та час. Вдале поєднання цих складових створювали цінність продукту і відповідну якість, яка б задовольняла клієнтські вимоги [35]. Однак, стало зрозумілим, що успіх проекту визначається не лише за даними критеріями. Існують інші фактори, які впливають на ефективність команди проекту та його успішність.

На основі попередніх досліджень та власних спостережень було визначено, що до факторів, які впливають на ефективність методології Scrum в проекті належать: експертність в методології, автономність команди, участь зацікавлених сторін, постійне вдосконалення та комунікація (рис. 3.1).

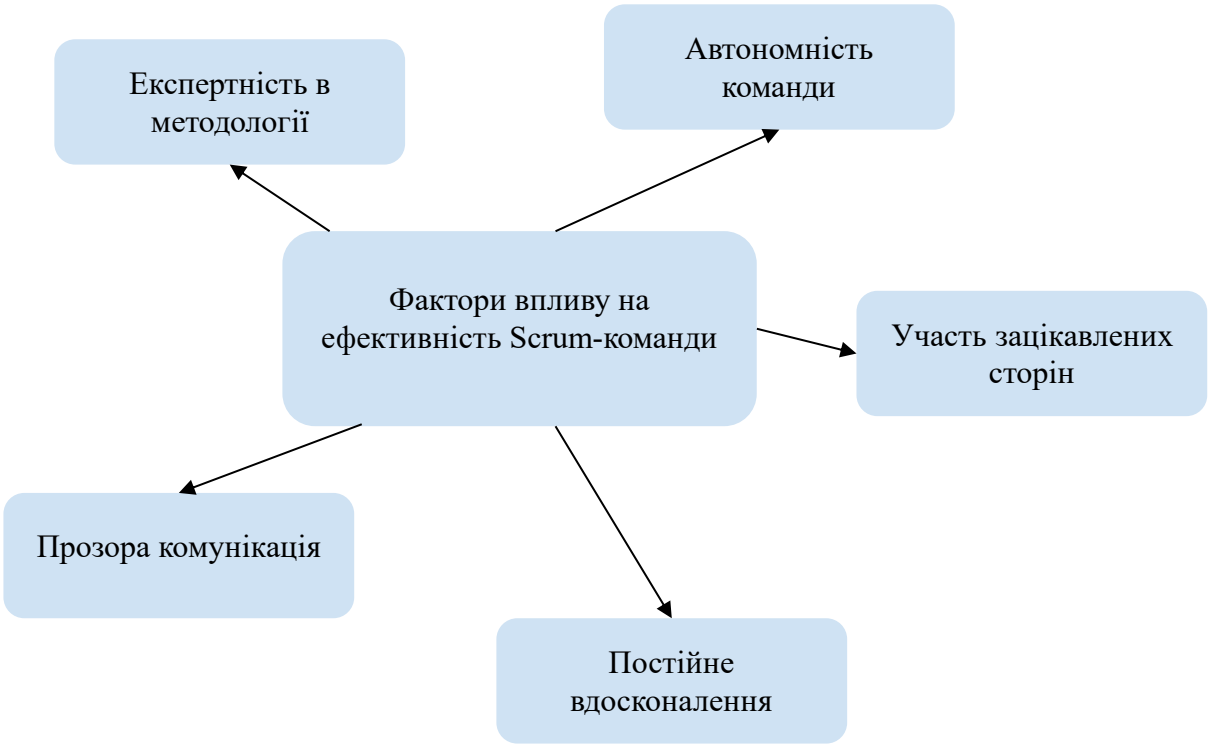


Рис. 3.1 – Фактори впливу на ефективність методології Scrum в проекті

Результати проведеного дослідження показали наскільки важливим є не лише дотримання всіх принципів та цінностей методології, а й підбір правильної команди, підготовка та навчання, налаштування внутрішнього середовища,

взаємодія зі стейкхолдерами, створення стратегії комунікації та постійне удосконалення процесів. На основі результатів аналізу було розроблено способи підвищення ефективності Scrum та покращення результатів від інтеграції даної методології в будь-який проект.

Навчання Scrum. В першу чергу, впровадженню методології в проект повинен передувати певний етап підготовки. Результати дослідження показали, що імплементація Scrum методології в проект без попереднього навчання команди може мати не дуже позитивні наслідки. Командам, які не були знайомі з методологією та її основними принципами і цінностями може бути важко зрозуміти важливість певних процесів та основних переваг організації роботи саме таким чином. Кожен член команди повинен вміти правильно оцінювати задачі, складати звіти та мати чітке розуміння того, навіщо проводиться daily scrum або ретроспектива. Тому в першу чергу було запропоновано розробити систему підготовки команди до впровадження Scrum. Система передбачає три етапи: ознайомлення, тренування, адаптація. Основна мета етапу ознайомлення: представити команді концепцію Scrum, його основні принципи, цінності та артефакти. Цей етап включатиме в себе наступні заходи:

1) Основи Scrum. Історія Scrum, ключові переваги та приклади застосування іншими організаціями покращать розуміння командою методології Scrum та мети його використання;
2) Детальний розгляд Scrum фреймворку, включаючи його події та артефакти. Команда детальніше ознайомиться з такими поняттями як спринт, щоденний скрам, ретроспектива, інкремент;
3) Обговорення цінностей Scrum сприятимуть глибшому розумінню того, що закладено в його основу і без чого неможливо досягти успіху.

На етапі навчання команда буде сфокусована на отриманні необхідних навичок для ефективного використання Scrum. По-перше, це проведення тренінгів, на яких фахівці зі Scrum зможуть відповісти на всі запитання команд, а також самостійне навчання командою на основі наданих матеріалів. Глибоке

занурення в процеси Scrum та покращення розуміння можливе за використання імітаційних спринтів, метою яких буде надати можливість команді попрактикуватися в Scrum без наслідків для проекту. Організація подібних заходів може займати додатковий час, але в довгостроковій перспективі зекономить кошти клієнту.

Важливо зазначити, що методи гнучкого оцінювання, які використовуються в методології Scrum базуються на суб'єктивній думці учасників оцінювання, що може призвести до упередженої оцінки. В першу чергу така проблема виникає в недосвітчених командах, які не мають попереднього досвіду використання Scrum та знань передметної області. Тому існує потреба в навчанні команди, а також наявності експерта з оцінки, який виступатиме в ролі коуча і доноситиме цінність та розуміння інструментів до кожного члена команди. Проте потрібно враховувати, що Scrum все ж таки передбачає наявність сильних та самоорганізованих командах, що є однією з його переваг, тому варто знаходити баланс та просуватися в напрямку до здобуття командою самостійності в подальшій роботі.

Окрім навчання та підготовки команди розробників до використання методології Scrum в проекті, досягнення успіху також передбачає високий рівень кваліфікації Scrum майстра. Я вважаю, що кожен фахівець зі Scrum повинен проходити певну сертифікацію для оцінки його рівня та визначення спроможності довести команду до результату.

Автономність команди. Автономність команди є передумовою концепції постійного вдосконалення, яка закладена в основу Scrum, а також важливим фактором, який впливає на ефективність Scrum-команди. Вона проявляється у високому рівні самоорганізованості та крос-функціональності. Чим вищий рівень автономії команди, тим більшу відповідальність вона бере на себе за досягнення результатів. Підвищення автономності команди можливе за впровадження наступних кроків:

- Визначення цілей та очікувань. Команда повинна розуміти мету та бажані результати своєї роботи, щоб будувати свою роботу відповідно до них;
- Сприяння відкритому спілкуванню та створення атмосфери відкритості і прозорості;
- Делегування повноважень щодо прийняття рішень членам команди для покращення у них навичок вирішення проблем, встановлення пріоритетів та прийняття правильних рішень;
- Надання можливостей для безперервного навчання і зростання за допомогою внутрішніх курсів та тренінгів.

Участь зацікавлених сторін. Власник продукту та клієнт – це ті сторони, які найкраще знають продукт та його бізнес-цінність і найбільш зацікавлені в його успішності. Вони встановлюють цілі та визначають пріорітети і можуть надати цінні ідеї для реалізації. Залучення їх до планування спринтів та проведення релізів покращує розуміння командою пріоритетів та допомагає їй зосередитися на створенні високоцінних функцій, які відповідають клієнтським очікуванням.

Крім того, стейкхолдери відіграють важливу роль у забезпеченні зворотного зв’язку. Їх участь в спринт-оглядах дає змогу перевіряти кожний інкремент на ранніх етапах, гарантуючи забезпечення відповідності вимогам і очікуванням. Залучення зацікавлених сторін також допомагає зменшити ризики, виявляючи потенційні проблеми і відхилення в процесі розробки та сприяє постійному вдосконаленню отриманих рішень з кожною ітерацією.

Встановлення контрольних точок. Спостереження за Scrum командою виявило схильність до відтягування термінів виконання задач до кінцевого дедлайну, що затримує перевірку коду старшим розробником і відповідно знижує її якість. Було запропоновано встановлювати так звані проміжні контрольні точки, метою яких буде перевірка готовності задач за деякий час до дедлайну, що допоможе підвищити продуктивність команди і уникнути випадків, коли розробники замовчують проблеми тим самим затягуючи терміни реалізації. Хоча

Scrum передбачає самостійність команд, деякий відсоток контролю під час спринту може покращити якість результату та завадити виникненню проблем під час релізу.

Проміжні дедлайни також допоможуть команді визначити пріоритети в роботі та зосередитися на найважливіших завданнях. Розбиття цілей на менші більш керовані частини з конкретними термінами дають змогу визначити що потрібно робити в першу чергу, запобігаючи непотрібним затримкам та забезпечуючи прогрес. Встановлення проміжних термінів також покращує розуміння очікувань від команди та заохочує їх дотримуватися визначених дедлайнів.

Пріоритизація беклогу. Пріоритезація беклогу – це визначення порядку в якому команда працюватиме над задачами з беклогу. Дослідження показало, наскільки важливим є даний аспект в досягненні цілей командою. За допомогою встановлення пріоритетів команда зосереджує свої зусилля на створенні найбільш цінних функцій на ранніх етапах. Працюючи над першочерговими завданнями команда також гарантує, що найбільш важливий функціонал буде реалізовано раніше, що забезпечить вигоду для клієнта та користувачів. Надання цінності на перших релізу не тільки підвищує задоволеність клієнта, а й сприяє покращенню мотивації команди за рахунок швидкого зворотного зв’язку. Визначення пріоритетів також допомагає виявити та усунути потенційні ризики на ранніх стадіях, перевірити припущення та вчасно внести необхідні корективи, зводячи до мінімуму необхідність переробок.

Пріоритезація беклогу повинна відбуватися в декілька етапів:

1) Визначення вимог користувачів і зацікавлених сторін. Це необхідно для розуміння їхніх очікувань від продукту і встановлення першочергових пріоритетів;
2) Встановлення критерій пріоритетності, які визначатимуть пріоритетність тієї чи іншої задачі відповідно до вимог;

3) Оцінювання задач та визначення пріоритетів на основі встановлених критерій;
4) Перегляд і переоцінка пріоритетів відповідно до змін клієнтських очікувань;

Перехресний аналіз. Будь-яка команда розробників зазвичай передбачає наявність одного керівника (Team Leader), до обов'язків якого входить перевірка коду кожного розробника. У випадку, коли команда складається із п'яти і більше осіб, перевірка роботи кожного з розробників може займати у керівника дуже багато часу, через що відбуваються затримки релізу та вихід за межі дедлайну. Рішенням даної проблеми може бути введення практики перехресного аналізу коду, тобто перевірка коду розробниками одне в одного перед здачею завдань керівнику. Таким чином розробники будуть вчитися на своїх помилках та помилках колег і покращуватимуть свої лідерські навички.

Загалом, дані рекомендації спрямовані на покращення процесу управління проектом та усунення основних проблем, які можуть виникати при застосуванні методології Scrum в поєднанні з іншими фреймворками.

ВИСНОВКИ

Метою цього дослідження було дослідити ефективність методології Scrum для реалізації IT-проекту невеликою командою. На основі спостереження за командою розробників та опитування було проведено аналіз впливу Scrum на досягнення проектних цілей. Результати дослідження підтвердили ключові переваги Scrum, зазначені в посібнику зі Scrum, а також звернули увагу на проблеми і виклики, з якими стикається команда проекту при впровадженні Scrum. Результати показали, що Scrum дійсно позитивно впливає на ефективність команди проекту, їхню мотивацію, згуртованість та здатність досягати результатів. Однак з іншого боку виникає велика кількість індивідуальних проблем і викликів які необхідно вирішити в процесі реалізації проекту.

Також було визначено, що простого використання практик та дотримання церемоній Scrum недостатньо для того, аби отримати всю користь від фреймворку. Необхідне глибоке розуміння цінностей та достатній рівень досвіду використання методології не лише Scrum майстром, а й кожним учасниками команди.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Тарасюк Г.М. (2009). Управління проектами: Навч. посібник. 3-є вид. К.: Каравела.
2. Ноздріна Л. В., Ящук В. І., Полотай О. І. (2010). Управління проектами: Підручник. Центр учбової літератури.
3. Краснокутська, Н. С. Осетрова Т. О. (2018). Еволюція розвитку та сучасні тренди в управлінні проектами. Тернопіль: Видавничо-поліграфічний центр Тернопільського національного економічного університету «Економічна думка».
4. Project management methodology: definition, types, examples. Retrieved from https://mymanagementguide.com/basics/project-methodology-definition/
5. Waterfall Model. Retrieved from https://www.projectmanager.com/guides/waterfall-methodology
6. 12 Project Management Methodologies. Retrieved from https://www.coursera.org/articles/project-management-methodologies-your-guide
7. What Is Waterfall Methodology? Here's How It Can Help Your Project Management Strategy. Retrieved from https://www.forbes.com/advisor/business/what-is-waterfall-methodology/
8. What is Kanban? Retrieved from https://www.digite.com/kanban/what-is-kanban/#:~:text=It%20all%20started%20in%20the,every%20stage%20of%20production%20optimally.
9. Scrumban. Retrieved from https://www.productplan.com/glossary/scrumban/
10. Scrumban: The best of two Agile methodologies. Retrieved from https://asana.com/ru/resources/scrumban
11. Critical Path Method. Retrieved from https://asana.com/ru/resources/critical-path-method

12. What is Lean. Retrieved from https://leanmethods.com/resources/articles/what-is-lean/
13. 9 Project Management Trends in 2023 – Where Are We Headed ? Retrieved from https://www.theprojectgroup.com/blog/en/project-management-trends/
14. Project Management Trends (2023): What does the Future Look Like Retrieved from https://www.proofhub.com/articles/project-management-trends/
15. Джефф Сазерленд. Scrum. (2016). Навчись робити вдвічі більше за менший час. Вид: Книжковий клуб «Клуб Сімейного Дозвілля».
16. Scrum for Education - Experiences from eduScrum and Blueprint Education. InfoQ. Retrieved from https://www.infoq.com/articles/scrum-education/
17. Being Well, Being Agile. Retrieved from https://resources.scrumalliance.org/Article/being-well-being-agile
18. R. Saradha. (2016). A Study on Adapting and using Scrum in Financial Sectors. IJERT. NCETCS. (Volume 4 – Issue 05). Retrieved from https://www.ijert.org/a-study-on-adapting-and-using-scrum-in-financial-sectors
19. Agile working creates opportunities for BBVA Compass in U.S. market. Retrieved from https://www.bbva.com/en/agile-working-creates-opportunities-bbva-compass-u-s-market/
20. Ken Schwaber, Jeff Sutherland. (2010). The Scrum Guide. The Definitive Guide to Scrum: The Rules of the Game / Ken Schwaber, Jeff Sutherland.
21. What is scrum and how to get started. Retrieved from https://www.atlassian.com/agile/scrum
22. What is Scrum? Retrieved from https://www.scrum.org/resources/what-scrum-module
23. Scrum Framework. Retrieved from https://www.mountaingoatsoftware.com/agile/scrum
24. What is scrum? Retrieved from https://www.scrumalliance.org/about-scrum
25. Демиденко М. А. (2020). «Сучасні методи управління проектами інформатизації»: навч. посіб. НТУ «Дніпровська політехніка».

26. Wrike Scrum Guide. Wrike. Retrieved from https://www.wrike.com/scrum-guide/
27. Learn how to use burndown charts in Jira Software. Retrieved from https://www.atlassian.com/agile/tutorials/burndown-charts
28. What Is a Scrum Board & How Do I Create One? Retrieved from https://www.projectmanager.com/blog/what-is-a-scrum-board
29. Status Quo (Scaled) Agile: 4th study on Benefits and Success factors of agile methods. Retrieved from https://www.process-and-project.net/studien/studienunterseiten/status-quo-scaled-agile-2020-en/
30. State of Agile Report. Retrieved from https://stateofagile.com/
31. S. Hassani-Alaoui , A. Cameron, T. Giannelia. (2020). "We Use Scrum, but …": Agile Modifications and Project Success. Proceedings of the 53rd Hawaii International Conference on System Sciences. HICSS Conference Office.
32. Igor Kononenko. (2014). THE METHODS OF SELECTION OF THE PROJECT MANAGEMENT METHODOLOGY. International Journal of Computing, 13(1).
33. User stories with examples and a template. Retrieved from https://www.atlassian.com/agile/project-management/user-stories
34. What are story points? Six easy steps to estimate work in Agile. Retrieved from https://asana.com/ru/resources/story-points
35. The project triangle. Retrieved from https://support.microsoft.com/en-us/office/the-project-triangle-8c892e06-d761-4d40-8e1f-17b33fdcf810
36. Christiaan Verwijs, Daniel Russo. (2022). A Theory of Scrum Team Effectiveness. ACM Trans. Softw. Eng. Methodol. 37, 4, Article 111.
37. Рекуненко І. І., Лук'янихін В. О., Майборода Т. М., Кобушко Я. В., Таранюк К. В. (2022) Методичні вказівки щодо підготовки, оформлення та захисту кваліфікаційної роботи бакалавра. Суми : Сумський державний університет.
38. The Guidebook for Project and Program Management for Enterprise Innovation (P2M) // Project Management Association of Japan, 2001.- 93 p.

39. В. О. Кузьміних, Р. А. Тараненко. (2019). Основи управління ІТ проектами: навч. посіб. для студ. спеціальності 122 «Комп’ютерні науки»/ КПІ ім. Ігоря Сікорського. Київ : КПІ ім. Ігоря Сікорського.
40. Микитюк П. П., Брич В. Я., Микитюк Ю. І., Труш І. М. (2021).Управління проектами: підручник. [для студ. вищ. навч. закл.]. Тернопіль.