Тема 2. Вибір методології управління проектом програмних систем. Формування системи управління проектом

1. Огляд сучасних методологій з управління проектами

Методологія управління проектами – це набір методів, процесів, принципів та практик, що визначають підхід до планування, виконання, моніторингу, контролю та завершення проектів. Правильний вибір методології є критично важливим для успіху проекту, оскільки вона визначає структуру роботи, розподіл ролей, комунікацію та процеси прийняття рішень.

1.1. PMBoK (Project Management Body of Knowledge)

PMBoK – це стандарт управління проектами, розроблений Інститутом управління проектами (Project Management Institute, PMI). Він представляє собою звід знань у галузі управління проектами, який узагальнює найкращі практики та підходи.

Основні характеристики PMBoK:

- **Процесний підхід** – управління проектом розглядається як сукупність взаємопов'язаних процесів.
- **Групи процесів** – PMBoK визначає п'ять груп процесів: ініціювання, планування, виконання, моніторинг та контроль, завершення.
- **Області знань** – стандарт виділяє десять областей знань з управління проектами:
    1. Управління інтеграцією проекту
    2. Управління змістом проекту
    3. Управління термінами проекту
    4. Управління вартістю проекту
    5. Управління якістю проекту
    6. Управління ресурсами проекту
    7. Управління комунікаціями проекту
    8. Управління ризиками проекту
    9. Управління закупівлями проекту
    10. Управління зацікавленими сторонами проекту

Переваги PMBoK:

- Універсальність – може застосовуватись до проектів у різних галузях
- Всеосяжність – охоплює всі аспекти управління проектами

- Визнаний міжнародний стандарт
- Детальний опис процесів, інструментів та методів

Недоліки PMBoK:

- Значна бюрократизація процесів
- Надмірна формалізація для невеликих проектів
- Недостатня гнучкість при змінах вимог

1.2. P2M (Project & Program Management for Enterprise Innovation)

P2M – це японський стандарт управління проектами, розроблений Японською асоціацією розвитку інжинірингу (ENAA). На відміну від інших методологій, P2M орієнтований на створення цінності та інновацій через проекти та програми.

Основні характеристики P2M:

- **Ціннісний підхід** – фокус на створенні цінності через проекти
- **Інтеграція проектів у програми** – узгодження проектів зі стратегічними цілями організації
- **Модель "Вежа P2M"** – концептуальна структура методології, що включає:
    - Вступний сегмент (основи управління проектами)
    - Сегмент управління програмами
    - Сегмент управління проектами (включає 11 областей знань)

Переваги P2M:

- Орієнтація на цінність та інновації
- Інтеграція зі стратегією організації
- Увага до загальної картини та контексту проектів
- Розвиток інтелектуального капіталу організації

Недоліки P2M:

- Менша поширеність у світі порівняно з PMBoK та PRINCE2
- Складність впровадження в організаціях з традиційним управлінням
- Вимагає високого рівня зрілості управління проектами

1.3. PRINCE2 (PRojects IN Controlled Environments)

PRINCE2 – це процесно-орієнтована методологія управління проектами, розроблена урядом Великої Британії як стандарт для управління IT-проектами, але згодом адаптована для проектів у різних галузях.

Основні характеристики PRINCE2:

- **Структурований підхід** – чітка структура проекту з визначеними ролями, відповідальністю та продуктами
- **Фокус на бізнес-обґрунтуванні** – проект повинен мати чітке бізнес-обґрунтування на всіх етапах
- **Управління за етапами** – проект розбивається на керовані етапи
- **Сім принципів PRINCE2:**
    1. Постійна оцінка доцільності проекту
    2. Навчання на досвіді
    3. Визначення ролей та відповідальності
    4. Управління за етапами
    5. Управління за винятками
    6. Фокус на продуктах
    7. Адаптація до середовища проекту

Переваги PRINCE2:

- Чітка організаційна структура з визначеними ролями
- Орієнтація на продукт та бізнес-обґрунтування
- Структурований контроль та моніторинг проекту
- Легка адаптованість до різних типів проектів

Недоліки PRINCE2:

- Значний обсяг документації
- Може бути надмірно бюрократичною для малих проектів
- Не включає специфічні галузеві практики
- Менш поширена в деяких регіонах світу

2. WaterFall та Agile підходи до управління IT проектами

2.1. WaterFall (Каскадна модель)

Каскадна модель – це лінійний послідовний підхід до управління проектами, при якому проект рухається через послідовні фази, і кожна наступна фаза починається тільки після завершення попередньої.

Основні фази Waterfall:

1. **Аналіз вимог** – визначення та документування вимог замовника
2. **Проектування** – розробка архітектури та детального дизайну системи
3. **Реалізація (Кодування)** – написання програмного коду

4. **Тестування** – перевірка відповідності розробленої системи вимогам
5. **Впровадження** – розгортання системи в робочому середовищі
6. **Підтримка** – виправлення помилок та внесення змін

Переваги Waterfall:

- Простота розуміння та управління
- Чітке визначення етапів та результатів
- Добре документований процес
- Ефективність для проектів з стабільними вимогами
- Легке визначення термінів та бюджету

Недоліки Waterfall:

- Низька гнучкість – складно вносити зміни після початку проекту
- Пізнє виявлення проблем
- Результати проекту видні лише в кінці
- Не підходить для проектів з нечіткими або мінливими вимогами
- Високий ризик невідповідності кінцевого продукту очікуванням замовника

Особливості застосування Waterfall:

- Підходить для проектів з чіткими, стабільними та добре зрозумілими вимогами
- Ефективний при строгих регуляторних вимогах, де необхідна детальна документація
- Доцільний для проектів з фіксованим бюджетом та термінами
- Підходить для проектів з низьким ризиком змін вимог
- Добре працює в команда з невисоким рівнем комунікації з замовником

2.2. Agile-підходи

Agile (гнучкі) методології – це група підходів до розробки програмного забезпечення, орієнтованих на використання інкрементної, ітеративної розробки, динамічне формування вимог та забезпечення їх реалізації внаслідок постійної взаємодії всередині самоорганізованих робочих груп та з замовником.

Основні принципи Agile (Agile Manifesto):

1. Люди та взаємодія важливіші за процеси та інструменти
2. Працюючий продукт важливіший за вичерпну документацію
3. Співпраця з замовником важливіша за узгодження умов контракту
4. Готовність до змін важливіша за дотримання плану

2.2.1. Scrum

Scrum – це фреймворк для розробки та підтримки складних продуктів, що базується на емпіричному підході, який стверджує, що знання приходить з досвідом, а рішення слід приймати на основі того, що відомо.

Ключові елементи Scrum:

- **Спринт** – часовий відрізок (зазвичай 1-4 тижні), протягом якого створюється готовий до використання інкремент продукту
- **Ролі:**
    - Власник продукту (Product Owner) – відповідає за бачення продукту та пріоритизацію завдань
    - Scrum-майстер (Scrum Master) – забезпечує дотримання процесів Scrum та усуває перешкоди
    - Команда розробників (Development Team) – виконує роботу з створення інкременту
- **Артефакти:**
    - Беклог продукту (Product Backlog) – список всіх функцій та вимог
    - Беклог спринту (Sprint Backlog) – список завдань на поточний спринт
    - Інкремент (Increment) – готовий до використання результат спринту
- **Церемонії:**
    - Планування спринту (Sprint Planning)
    - Щоденний Scrum (Daily Scrum) – короткі щоденні зустрічі команди
    - Огляд спринту (Sprint Review) – демонстрація результатів
    - Ретроспектива спринту (Sprint Retrospective) – аналіз процесу роботи

Переваги Scrum:

- Раннє і регулярне отримання цінності
- Висока адаптивність до змін
- Краща видимість прогресу
- Висока залученість замовника
- Підвищена мотивація команди

Недоліки Scrum:

- Складно оцінити терміни та бюджет всього проекту
- Вимагає досвідченої самоорганізованої команди

- Може бути складним для великих проектів
- Вимагає постійної залученості всіх учасників

2.2.2. Kanban

Kanban – це метод управління розробкою програмного забезпечення, що фокусується на візуалізації робочого процесу, обмеженні незавершеної роботи та вимірюванні часу циклу.

Ключові елементи Kanban:

- **Kanban-дошка** – візуальне представлення роботи в різних стадіях
- **Ліміти незавершеної роботи (WIP limits)** – обмеження кількості завдань на кожному етапі
- **Потік (Flow)** – управління плавним рухом роботи через систему
- **Політики** – явні правила роботи в системі
- **Зворотний зв'язок** – регулярні зустрічі для аналізу та поліпшення процесу

Переваги Kanban:

- Гнучкість – немає фіксованих спринтів
- Фокус на завершенні роботи
- Зменшення часу циклу
- Прозорість процесу
- Плавне впровадження без радикальних змін

Недоліки Kanban:

- Відсутність часових рамок може призвести до затягування роботи
- Менше формалізації може ускладнювати планування
- Вимагає дисципліни від команди
- Не визначає ролі та відповідальності

2.2.3. Lean

Lean (ощадливий підхід) – це методологія, що фокусується на усуненні всіх видів втрат у процесі розробки і створенні максимальної цінності для замовника з мінімальними ресурсами.

Ключові принципи Lean:

1. **Визначення цінності** – розуміння, що є цінним для замовника
2. **Відображення потоку цінності** – аналіз всіх кроків процесу
3. **Створення потоку** – забезпечення безперервного руху роботи

4. **Впровадження системи витягування (Pull)** – робота починається тільки при наявності запиту
5. **Постійне вдосконалення** – безперервний пошук шляхів покращення

Переваги Lean:

- Фокус на цінності для замовника
- Усунення непродуктивних дій
- Прискорення доставки цінності
- Підвищення якості
- Оптимізація використання ресурсів

Недоліки Lean:

- Вимагає змін в організаційній культурі
- Складно впровадити в традиційних організаціях
- Потребує постійної підтримки та дисципліни
- Менша структурованість порівняно з Scrum

2.2.4. Six Sigma

Six Sigma – це методологія, орієнтована на поліпшення якості процесів шляхом виявлення та усунення причин дефектів (помилок) і мінімізації варіативності в процесах.

Ключові елементи Six Sigma:

- **DMAIC** – цикл поліпшення процесів:
    - Define (Визначити) – визначення проблеми та цілей
    - Measure (Виміряти) – збір даних про поточний процес
    - Analyze (Аналізувати) – аналіз причин проблем
    - Improve (Покращити) – впровадження поліпшень
    - Control (Контролювати) – контроль нового процесу
- **Статистичний контроль процесів**
- **Орієнтація на дані та вимірювання**
- **Ієрархія ролей** (Чемпіони, Майстри чорного поясу, Чорні пояси, Зелені пояси)

Переваги Six Sigma:

- Зниження дефектів та підвищення якості
- Орієнтація на кількісні показники
- Систематичний підхід до вирішення проблем

- Зниження варіативності процесів
- Поліпшення прогнозованості результатів

Недоліки Six Sigma:

- Висока складність впровадження
- Вимагає значних інвестицій у навчання
- Орієнтація на процеси може відволікати від продукту
- Надмірний фокус на вимірюваннях і статистиці
- Менш гнучкий порівняно з іншими Agile-методологіями

3. Основні принципи, переваги, недоліки та особливості застосування підходів

3.1. Порівняння Waterfall та Agile підходів

Критерій	Waterfall	Agile
Підхід до розробки	Лінійний, послідовний	Ітеративний, інкрементний
Вимоги	Детально визначені на початку	Розвиваються протягом проекту
Гнучкість	Низька, складно вносити зміни	Висока, зміни є очікуваними
Участь замовника	В основному на початку та в кінці	Постійна протягом проекту
Документація	Всеосяжна, детальна	Мінімально необхідна
Тестування	Окрема фаза в кінці розробки	Інтегроване в кожну ітерацію
Ризики	Виявляються пізно	Виявляються рано і постійно
Вартість змін	Висока, особливо в пізніх фазах	Відносно низька
Передбачуваність	Висока для бюджету і термінів	Нижча, але вища для якості
Розмір команди	Може бути великою	Зазвичай невелика (5-9 осіб)
Підходить для проектів	З стабільними вимогами, чіткими термінами	З мінливими вимогами, інноваційних

3.2. Критерії вибору методології

Вибір методології управління проектом залежить від багатьох факторів:

1. **Характеристики проекту:**
    - Розмір і складність
    - Визначеність вимог
    - Стабільність середовища

    - Інноваційність
2. **Організаційні фактори:**
    - Корпоративна культура
    - Рівень зрілості управління проектами
    - Доступні ресурси
    - Регуляторні вимоги
3. **Характеристики команди:**
    - Досвід і кваліфікація
    - Розмір команди
    - Розташування (локальна чи розподілена)
    - Рівень самоорганізації
4. **Характеристики замовника:**
    - Рівень залученості
    - Готовність до співпраці
    - Досвід участі в проектах
    - Чіткість бачення продукту

3.3. Рекомендації щодо вибору методології

Waterfall рекомендується, коли:

- Вимоги чітко визначені та малоймовірно, що вони зміняться
- Проект має суворі регуляторні вимоги
- Потрібна детальна документація
- Замовник не може бути постійно залучений
- Команда розподілена і/або має обмежений досвід

Agile (Scrum) рекомендується, коли:

- Вимоги нечіткі або ймовірні зміни
- Важливо рано і часто отримувати зворотний зв'язок
- Замовник готовий до активної участі
- Команда досвідчена і готова до самоорганізації
- Інноваційність та швидкість виводу на ринок є критичними

Kanban рекомендується, коли:

- Проект пов'язаний з обслуговуванням або підтримкою
- Робочі завдання надходять нерегулярно
- Важливо оптимізувати потік роботи
- Необхідна висока гнучкість щодо пріоритетів

Lean рекомендується, коли:

- Фокус на максимізації цінності для замовника
- Є проблеми з ефективністю процесів
- Необхідно усунути втрати в робочому процесі
- Організація готова до культурних змін

Six Sigma рекомендується, коли:

- Критично важлива висока якість
- Є проблеми з варіативністю процесів
- Доступні дані для аналізу
- Організація готова інвестувати в навчання

4. Принципи формування системи управління проектом

Система управління проектом – це комплекс взаємопов'язаних організаційних, методичних, програмних, технічних та інформаційних засобів, спрямованих на підтримку та підвищення ефективності процесів планування та управління проектом.

4.1. Ключові принципи формування системи управління проектом

1. **Принцип цілеспрямованості**
    - Система повинна бути орієнтована на досягнення цілей проекту
    - Всі елементи системи повинні узгоджуватися з цілями
2. **Принцип системності**
    - Управління проектом має розглядатися як система взаємопов'язаних елементів
    - Зміни в одному елементі впливають на всю систему
3. **Принцип комплексності**
    - Система повинна охоплювати всі аспекти управління проектом
    - Необхідно враховувати взаємозв'язок між різними областями знань
4. **Принцип адаптивності**
    - Система повинна бути гнучкою і адаптуватися до змін

- Необхідно передбачати можливість коригування процесів

5. **Принцип ієрархічності**
    - Система повинна мати чітко визначену ієрархію управління
    - Повноваження та відповідальність повинні бути чітко розподілені
6. **Принцип зворотного зв'язку**
    - Система повинна забезпечувати постійний моніторинг та оцінку прогресу
    - Результати оцінки використовуються для коригування дій
7. **Принцип економічності**
    - Витрати на створення і функціонування системи повинні бути оптимальними
    - Система не повинна створювати надмірне адміністративне навантаження
8. **Принцип нормативності**
    - Система повинна базуватися на узгоджених нормах, стандартах та процедурах
    - Нормативна база повинна бути документована та доступна

4.2. Компоненти системи управління проектом

1. **Організаційна структура управління проектом**
    - Ролі та відповідальності учасників
    - Схема комунікацій
    - Процедури прийняття рішень
2. **Процеси управління проектом**
    - Процеси ініціювання
    - Процеси планування
    - Процеси виконання
    - Процеси моніторингу та контролю
    - Процеси завершення
3. **Методи та інструменти управління**
    - Методи планування (WBS, мережеві графіки, розрахунок критичного шляху)
    - Методи оцінки (параметрична оцінка, аналогова оцінка, експертна оцінка)
    - Методи аналізу ризиків
    - Методи контролю якості

4. **Інформаційна система**
    - Програмне забезпечення для управління проектами
    - Системи документообігу
    - Бази даних та знань
    - Засоби комунікації
5. **Система мотивації та розвитку команди**
    - Методи підбору персоналу
    - Системи навчання
    - Системи оцінки персоналу
    - Системи мотивації та винагороди

4.3. Етапи формування системи управління проектом

1. **Аналіз умов реалізації проекту**
    - Аналіз зовнішніх факторів
    - Аналіз внутрішніх факторів
    - Оцінка наявних ресурсів
2. **Визначення методологічного підходу**
    - Вибір базової методології
    - Адаптація до специфіки проекту
    - Визначення ключових процесів
3. **Розробка нормативної бази**
    - Створення положень та регламентів
    - Розробка шаблонів документів
    - Визначення метрик та індикаторів
4. **Формування організаційної структури**
    - Визначення ролей та відповідальності
    - Формування команди проекту
    - Визначення процедур координації та комунікації
5. **Вибір та впровадження інструментів**
    - Вибір програмного забезпечення
    - Налаштування інформаційних систем

    - Навчання персоналу
6. **Пілотне впровадження та коригування**
    - Тестування системи на пілотному проекті
    - Збір зворотного зв'язку
    - Коригування системи
7. **Повномасштабне впровадження**
    - Розширення використання на всі проекти
    - Інтеграція з іншими системами організації
    - Створення механізмів постійного вдосконалення

5. Конвергенція систем управління проектами

Конвергенція в контексті управління проектами означає процес зближення та інтеграції різних методологій, підходів та практик для створення оптимальної системи управління, що відповідає конкретним потребам організації та проекту.

5.1. Причини конвергенції систем управління проектами

1. **Обмеження класичних методологій**
    - Жодна з методологій не є універсальною
    - Кожна методологія має власні переваги та недоліки
2. **Розвиток підходів до управління**
    - Еволюція методологій під впливом практичного досвіду
    - Поява нових інструментів та технологій
3. **Глобалізація та міжнародні команди**
    - Необхідність адаптації до різних культур та практик
    - Потреба в універсальних підходах для розподілених команд
4. **Різноманітність проектів**
    - Різні типи проектів потребують різних підходів
    - В рамках одного портфеля можуть бути проекти різного типу
5. **Організаційна зрілість**
    - З розвитком організації змінюються потреби в управлінні
    - Зрілі організації здатні інтегрувати різні підходи

5.2. Форми конвергенції систем управління проектами

1. **Гібридні методології**

- Waterfall + Agile (WAgile)
- PRINCE2 Agile
- Дисциплінована Agile Доставка (DAD)
- Scaled Agile Framework (SAFe)

2. **Адаптація стандартних методологій**
   - Спрощення PMBoK для малих проектів
   - Адаптація Scrum для великих організацій
   - Модифікація PRINCE2 для гнучких проектів
3. **Ситуаційний підхід**
   - Вибір підходу залежно від типу проекту
   - Зміна методології на різних етапах проекту
   - Різні підходи для різних частин проекту
4. **Багаторівневі системи**
   - Agile на рівні команд
   - Класичні підходи на рівні проекту
   - Програмне управління на рівні портфеля

5.3. Принципи ефективної конвергенції

1. **Фокус на цілях, а не на процесах**
   - Методологія – це засіб, а не мета
   - Вибір найкращих практик для досягнення конкретних цілей
2. **Поступова еволюція**
   - Впровадження змін крок за кроком
   - Аналіз результатів та коригування підходу
3. **Залучення команди**
   - Участь команди у формуванні підходу
   - Врахування досвіду та рекомендацій учасників
4. **Постійне навчання**
   - Регулярна оцінка ефективності підходу
   - Впровадження покращень на основі отриманого досвіду
5. **Адаптація до контексту**

- Врахування особливостей організації
- Врахування специфіки проекту
- Врахування характеристик команди

5.4. Приклади успішної конвергенції

1. **Scrumban**
    - Поєднання часових обмежень Scrum із візуалізацією та потоком Kanban
    - Збалансоване поєднання структури та гнучкості
2. **Водоспадний розвиток із Agile-доставкою**
    - Класичне планування та дизайн
    - Agile-підхід до розробки та тестування
    - Структуроване впровадження
3. **Lean Six Sigma**
    - Інтеграція ощадливого виробництва з методологією підвищення якості
    - Фокус на цінності для клієнта та зниженні варіативності процесів
4. **PRINCE2 Agile**
    - Структура управління та контролю PRINCE2
    - Гнучкість та адаптивність Agile
    - Чіткі ролі та відповідальність із адаптивним виконанням

6. Вимоги до проектного менеджера

Проектний менеджер (керівник проекту) – це фахівець, відповідальний за досягнення цілей проекту шляхом планування, організації, координації та контролю ресурсів та діяльності протягом усього життєвого циклу проекту.

6.1. Професійні компетенції проектного менеджера

1. **Технічні компетенції**
    - Знання методологій управління проектами
    - Володіння інструментами планування та контролю
    - Розуміння предметної області
    - Знання технологій, що використовуються в проекті
2. **Управлінські компетенції**
    - Стратегічне мислення та планування
    - Організація та координація робіт

- Управління ресурсами
- Прийняття рішень
- Управління змінами
- Управління ризиками
- Контроль та моніторинг

3. **Лідерські компетенції**
    - Формування бачення та цілей
    - Мотивація та надихання команди
    - Вирішення конфліктів
    - Коучинг та розвиток команди
    - Делегування повноважень
4. **Комунікативні компетенції**
    - Ефективна комунікація з різними зацікавленими сторонами
    - Проведення презентацій та зустрічей
    - Навички ведення переговорів
    - Управління очікуваннями
    - Крос-культурна комунікація
5. **Бізнес-компетенції**
    - Розуміння бізнес-стратегії
    - Фінансова грамотність
    - Орієнтація на клієнта
    - Розуміння галузевих трендів
    - Навички аналізу та прийняття рішень

6.2. Особисті якості проектного менеджера

1. **Лідерство**
    - Здатність вести за собою
    - Впевненість у своїх діях
    - Відповідальність за результат
    - Здатність надихати та мотивувати
2. **Гнучкість та адаптивність**

- Швидка адаптація до змін
- Здатність працювати в умовах невизначеності
- Готовність змінювати підходи

3. **Стресостійкість**
    - Збереження працездатності в стресових ситуаціях
    - Управління емоціями
    - Витривалість при високому навантаженні
4. **Проактивність**
    - Ініціативність у вирішенні проблем
    - Орієнтація на результат
    - Бачення можливостей
5. **Критичне мислення**
    - Аналітичні здібності
    - Системне мислення
    - Оцінка ризиків та можливостей
6. **Етичність та чесність**
    - Дотримання професійної етики
    - Чесність у комунікації
    - Прозорість у діях

6.3. Специфічні вимоги до IT-проектного менеджера

1. **Технічні знання**
    - Розуміння процесів розробки програмного забезпечення
    - Знання методологій розробки (Agile, Waterfall)
    - Розуміння архітектури програмних систем
    - Знання технологій та інструментів розробки
2. **Управління командою розробників**
    - Розуміння специфіки роботи розробників
    - Здатність оцінювати технічні можливості та обмеження
    - Вміння мотивувати технічних фахівців
3. **Управління змінними вимогами**

- Гнучкість в управлінні обсягом робіт
- Пріоритизація функціональності
- Управління технічним боргом

4. **Технічна комунікація**
    - Здатність спілкуватися з технічними фахівцями
    - Вміння перекладати технічні концепції для нетехнічних зацікавлених сторін
    - Управління технічною документацією
5. **Розуміння якості програмного забезпечення**
    - Знання принципів забезпечення якості
    - Розуміння методів тестування
    - Управління технічними ризиками

6.4. Вимоги до проектного менеджера в різних методологіях

1. **Традиційні методології (Waterfall, PMBoK)**
    - Фокус на плануванні та контролі
    - Управління документацією
    - Строге слідування процесам
    - Управління змінами через формальні процедури
2. **Agile-методології**
    - Фасилітація та коучинг
    - Підтримка самоорганізації команди
    - Управління через лідерство, а не формальну владу
    - Адаптивність та гнучкість
    - Фокус на цінності для замовника
3. **Scrum (Scrum Master / Product Owner)**
    - Забезпечення дотримання процесів Scrum
    - Усунення перешкод для команди
    - Фасилітація церемоній
    - Управління беклогом продукту
    - Забезпечення розуміння вимог командою
4. **PRINCE2**

- Фокус на бізнес-обґрунтуванні
- Управління за винятками
- Чітке визначення ролей та відповідальності
- Фокус на продуктах проекту
- Адаптація методології до потреб проекту

7. Вимоги до команди проекту

Команда проекту – це група фахівців, які працюють разом для досягнення цілей проекту, вносячи свої знання, навички та досвід.

7.1. Загальні вимоги до команди проекту

1. **Професійні якості**
    - Відповідний рівень кваліфікації та досвіду
    - Технічні знання та навички
    - Розуміння предметної області
    - Знання методологій та інструментів
2. **Командні компетенції**
    - Здатність до співпраці
    - Ефективна комунікація
    - Вміння працювати в команді
    - Взаємодопомога та підтримка
3. **Особисті якості**
    - Відповідальність та дисципліна
    - Ініціативність та проактивність
    - Адаптивність до змін
    - Орієнтація на результат
    - Стресостійкість
4. **Організаційні вимоги**
    - Відповідність корпоративній культурі
    - Дотримання правил та процедур
    - Розуміння бізнес-цілей
    - Лояльність до організації

7.2. Специфічні вимоги до команди IT-проекту

1. **Технічні компетенції**
    - Знання мов програмування та технологій
    - Розуміння архітектури програмних систем
    - Досвід роботи з відповідними інструментами
    - Знання методологій розробки
2. **Якість коду та технічних рішень**
    - Дотримання стандартів кодування
    - Уважність до технічного боргу
    - Фокус на якості та тестуванні
    - Безпечність та надійність рішень
3. **Технічна комунікація**
    - Здатність пояснювати технічні рішення
    - Документування коду та рішень
    - Ефективна комунікація з іншими технічними фахівцями
4. **Навчання та розвиток**
    - Готовність вивчати нові технології
    - Обмін знаннями всередині команди
    - Слідкування за технологічними трендами

7.3. Вимоги до команди в різних методологіях

1. **Традиційні методології (Waterfall, PMBoK)**
    - Чітке слідування ролям та обов'язкам
    - Дотримання процесів та процедур
    - Фокус на документації та звітності
    - Виконання робіт згідно з планом
2. **Agile-методології**
    - Самоорганізація та самоуправління
    - Крос-функціональність (різні навички в одній команді)
    - Готовність до змін та адаптації
    - Прозорість та відкритість
    - Постійне вдосконалення

3. **Scrum-команда**
    - Крос-функціональність та самодостатність
    - Відданість спринту та його цілям
    - Активна участь у церемоніях Scrum
    - Колективна відповідальність за результат
4. **Kanban-команда**
    - Фокус на завершенні робіт
    - Дотримання лімітів незавершеної роботи
    - Постійне вдосконалення процесу
    - Колективна відповідальність за потік
5. **DevOps-команда**
    - Інтеграція розробки та експлуатації
    - Автоматизація процесів
    - Постійна інтеграція та доставка
    - Відповідальність за весь життєвий цикл продукту

7.4. Баланс компетенцій в команді проекту

1. **Технічні ролі та компетенції**
    - Архітектори та технічні лідери
    - Розробники (різні спеціалізації)
    - Тестувальники та QA-інженери
    - DevOps-інженери
    - UX/UI дизайнери
2. **Управлінські ролі та компетенції**
    - Керівник проекту / Scrum-майстер
    - Власник продукту / Бізнес-аналітик
    - Координатори та фасилітатори
3. **Допоміжні ролі та компетенції**
    - Документатори та технічні письменники
    - Тренери та наставники
    - Фахівці з забезпечення якості

4. **Рівні досвіду**
    - Баланс між досвідченими фахівцями та початківцями
    - Наставництво та передача знань
    - Розподіл завдань відповідно до рівня досвіду

8. Етапи життєвого циклу команди проекту

Життєвий цикл команди проекту описує етапи розвитку групи людей від початку формування до завершення проекту. Розуміння цих етапів допомагає керівнику проекту ефективно управляти командою.

8.1. Модель Такмана (Forming-Storming-Norming-Performing-Adjourning)

Брюс Такман запропонував одну з найбільш відомих моделей розвитку команди, яка включає п'ять етапів:

8.1.1. Формування (Forming)

Характеристики етапу:

- Знайомство членів команди один з одним
- Визначення цілей та завдань проекту
- Встановлення основних правил
- Висока залежність від лідера
- Обережність у взаємодії
- Низька продуктивність

Поведінка членів команди:

- Ввічливість і обережність у спілкуванні
- Прагнення зрозуміти свою роль
- Орієнтація на керівника
- Формальні відносини

Дії керівника проекту:

- Чітке визначення цілей і завдань
- Створення атмосфери довіри
- Сприяння знайомству членів команди
- Встановлення базових правил роботи
- Відповіді на питання та надання інформації

8.1.2. Штормінг (Storming)

Характеристики етапу:

- Виникнення конфліктів та суперечок
- Боротьба за владу та вплив
- Опір груповим завданням та вимогам
- Перегляд цілей та способів їх досягнення
- Емоційні реакції на завдання

Поведінка членів команди:

- Вираження незгоди та критики
- Формування підгруп та коаліцій
- Прояв індивідуальності
- Опір контролю

Дії керівника проекту:

- Управління конфліктами
- Заохочення відкритого спілкування
- Коучинг та підтримка
- Фокусування на цілях та результатах
- Підвищення згуртованості команди

8.1.3. Нормалізація (Norming)

Характеристики етапу:

- Розвиток згуртованості команди
- Встановлення норм та стандартів роботи
- Зростання довіри та поваги
- Прийняття ролей та обов'язків
- Зосередження на спільних цілях

Поведінка членів команди:

- Прийняття думок та ідей інших
- Конструктивна критика
- Відкрите спілкування
- Взаємодопомога та підтримка

Дії керівника проекту:

- Делегування відповідальності
- Фокус на розвитку навичок
- Заохочення ініціативи
- Встановлення ефективних процесів роботи
- Моніторинг дотримання норм

8.1.4. Продуктивна робота (Performing)

Характеристики етапу:

- Висока продуктивність та ефективність
- Взаємозалежність та гнучкість
- Стратегічна свідомість
- Автономність у роботі
- Фокус на досягненні результатів

Поведінка членів команди:

- Самостійне вирішення проблем
- Проактивність та ініціативність
- Взаємна підтримка та допомога
- Конструктивна критика
- Зосередженість на результаті

Дії керівника проекту:

- Фасилітація замість прямого управління
- Визнання досягнень
- Забезпечення ресурсами
- Усунення зовнішніх перешкод
- Підтримка мотивації

8.1.5. Розформування (Adjourning)

Характеристики етапу:

- Завершення завдань і проектів
- Підбиття підсумків та оцінка результатів
- Передача знань та документації
- Емоційне завершення роботи

- Перехід до нових проектів

Поведінка членів команди:

- Змішані емоції щодо завершення проекту
- Гордість за досягнуті результати
- Тривога щодо майбутнього
- Святкування успіхів

Дії керівника проекту:

- Формалізація завершення проекту
- Визнання внеску кожного члена команди
- Забезпечення переходу до інших проектів
- Збереження та поширення набутого досвіду
- Організація заходів з відзначення завершення проекту

8.2. Особливості управління командою на різних етапах життєвого циклу

8.2.1. Формування команди

Методи формування команди:

- Підбір за компетенціями та досвідом
- Врахування особистісних характеристик
- Балансування різних типів особистостей
- Визначення необхідних ролей та функцій

Інструменти формування команди:

- Проведення командоутворюючих заходів
- Визначення командних правил та норм
- Створення командної хартії або угоди
- Розвиток навичок комунікації

8.2.2. Розвиток команди

Методи розвитку команди:

- Навчання та тренінги
- Коучинг та наставництво
- Делегування повноважень
- Ротація ролей та завдань

Інструменти розвитку команди:

- Регулярні зустрічі та обговорення
- Зворотний зв'язок та оцінка
- Командні ретроспективи
- Спільне вирішення проблем

8.2.3. Підтримка ефективності команди

Методи підтримки ефективності:

- Мотивація та винагорода
- Управління конфліктами
- Підтримка балансу навантаження
- Забезпечення необхідними ресурсами

Інструменти підтримки ефективності:

- Системи моніторингу та контролю
- Регулярні звітності та огляди прогресу
- Системи управління знаннями
- Інструменти співпраці та комунікації

8.2.4. Завершення роботи команди

Методи завершення роботи:

- Формалізація передачі результатів
- Документування уроків та досвіду
- Оцінка індивідуальної та командної роботи
- Планування подальшого розвитку

Інструменти завершення роботи:

- Заключні звіти та презентації
- Церемонії закриття проекту
- Інтерв'ю з членами команди
- База знань проекту

9. Ролі та функції членів команди проекту

9.1. Традиційні ролі в команді проекту

9.1.1. Спонсор проекту

Функції:

- Забезпечення фінансування проекту
- Захист інтересів проекту на вищому рівні
- Прийняття ключових стратегічних рішень
- Затвердження основних результатів
- Усунення організаційних перешкод

Відповідальність:

- Забезпечення бізнес-обґрунтування проекту
- Призначення керівника проекту
- Забезпечення ресурсами
- Організаційна підтримка проекту

9.1.2. Керівник проекту

Функції:

- Планування та організація робіт
- Управління командою проекту
- Контроль виконання плану проекту
- Управління ризиками та змінами
- Комунікація з зацікавленими сторонами

Відповідальність:

- Досягнення цілей проекту
- Дотримання обмежень (терміни, бюджет, обсяг)
- Управління ресурсами проекту
- Звітність перед спонсором та зацікавленими сторонами

9.1.3. Члени команди проекту

Функції:

- Виконання завдань згідно з планом
- Забезпечення якості результатів
- Звітування про прогрес та проблеми
- Участь у плануванні та оцінці робіт

Відповідальність:

- Якісне виконання доручених завдань
- Дотримання термінів
- Участь у командній роботі
- Підтримка комунікації

9.1.4. Зацікавлені сторони (Stakeholders)

Функції:

- Визначення вимог та очікувань
- Погодження ключових рішень
- Надання зворотного зв'язку
- Підтримка проекту

Відповідальність:

- Чітке формулювання вимог
- Своєчасне надання інформації
- Участь у прийнятті рішень
- Прийняття результатів проекту

9.2. Ролі в Agile-команді

9.2.1. Власник продукту (Product Owner)

Функції:

- Визначення бачення продукту
- Управління беклогом продукту
- Пріоритизація функціональності
- Прийняття рішень щодо змісту продукту
- Взаємодія з замовником та зацікавленими сторонами

Відповідальність:

- Максимізація цінності продукту
- Чітке формулювання вимог
- Прийняття готової функціональності
- Забезпечення розуміння вимог командою

9.2.2. Scrum-майстер (Scrum Master)

Функції:

- Фасилітація Scrum-процесів
- Усунення перешкод для команди
- Коучинг команди та організації
- Забезпечення дотримання практик Scrum

Відповідальність:

- Ефективність Scrum-процесів
- Підтримка самоорганізації команди
- Сприяння постійному вдосконаленню
- Навчання Scrum-практикам

9.2.3. Команда розробників (Development Team)

Функції:

- Розробка інкременту продукту
- Оцінка робіт та планування спринту
- Самоорганізація для досягнення цілей спринту
- Постійне вдосконалення процесів та практик

Відповідальність:

- Якість інкременту продукту
- Дотримання технічних стандартів
- Досягнення цілей спринту
- Прозорість та відкритість щодо прогресу та проблем

9.3. Технічні ролі в IT-проектах

9.3.1. Архітектор / Технічний лідер

Функції:

- Розробка архітектури системи
- Прийняття ключових технічних рішень
- Забезпечення технічної цілісності системи
- Менторство та технічне керівництво

Відповідальність:

- Якість архітектури та технічних рішень
- Відповідність системи вимогам

- Технічні стандарти та практики
- Технічна документація

9.3.2. Розробники (Developers)

Функції:

- Написання програмного коду
- Участь у проектуванні системи
- Виконання code review
- Створення модульних тестів

Відповідальність:

- Якість та надійність коду
- Відповідність технічним стандартам
- Інтеграція компонентів
- Виправлення дефектів

9.3.3. Тестувальники (QA Engineers)

Функції:

- Розробка тестових сценаріїв
- Виконання тестування
- Документування дефектів
- Верифікація виправлення дефектів

Відповідальність:

- Якість тестування
- Виявлення дефектів
- Відповідність продукту вимогам
- Регресійне тестування

9.3.4. DevOps-інженери

Функції:

- Налаштування середовищ розробки та тестування
- Автоматизація процесів збірки та розгортання
- Управління інфраструктурою
- Моніторинг та підтримка системи

Відповідальність:

- Стабільність середовищ
- Ефективність процесів CI/CD
- Безпека інфраструктури
- Швидкість та надійність розгортання

9.3.5. Бізнес-аналітики

Функції:

- Аналіз бізнес-процесів та вимог
- Розробка документації вимог
- Узгодження вимог з зацікавленими сторонами
- Валідація реалізації вимог

Відповідальність:

- Якість та повнота вимог
- Відповідність вимог бізнес-цілям
- Комунікація між бізнесом та технічною командою
- Документація вимог

9.3.6. UX/UI дизайнери

Функції:

- Розробка інтерфейсів користувача
- Проектування користувацького досвіду
- Створення прототипів та макетів
- Проведення юзабіліті-тестування

Відповідальність:

- Якість інтерфейсів
- Відповідність інтерфейсів потребам користувачів
- Дотримання стандартів дизайну
- Узгодженість інтерфейсів

9.4. Командні ролі за Белбіном

Модель командних ролей Белбіна визначає дев'ять ролей, які необхідні для ефективної роботи команди. Ці ролі не пов'язані з функціональними обов'язками, а відображають поведінкові патерни та природні схильності людей.

9.4.1. Ролі, орієнтовані на дію

Виконавець (Implementer)

- Характеристики: дисциплінований, надійний, консервативний, ефективний
- Внесок: перетворює ідеї на практичні дії, систематично та ефективно працює
- Допустимі слабкості: брак гнучкості, повільна реакція на нові можливості

Оформлювач (Shaper)

- Характеристики: динамічний, напористий, орієнтований на завдання
- Внесок: кидає виклик, тисне, знаходить обхідні шляхи
- Допустимі слабкості: схильність до провокацій, може ображати почуття інших

Завершувач (Completer Finisher)

- Характеристики: старанний, сумлінний, тривожний
- Внесок: шукає помилки та упущення, доводить справу до кінця
- Допустимі слабкості: схильність до зайвого занепокоєння, неохоче делегує

9.4.2. Ролі, орієнтовані на людей

Координатор (Co-ordinator)

- Характеристики: зрілий, впевнений, хороший голова
- Внесок: з'ясовує цілі, сприяє прийняттю рішень, делегує ефективно
- Допустимі слабкості: може сприйматися як маніпулятор, делегує особисту роботу

Командний гравець (Team Worker)

- Характеристики: співчутливий, дипломатичний, сприйнятливий
- Внесок: слухає, будує, знімає напругу, запобігає конфліктам
- Допустимі слабкості: нерішучий у критичних ситуаціях, уникає конфронтації

Дослідник ресурсів (Resource Investigator)

- Характеристики: екстравертний, ентузіаст, комунікабельний
- Внесок: вивчає можливості, розвиває контакти, налагоджує зв'язки
- Допустимі слабкості: надмірно оптимістичний, втрачає інтерес після початкового ентузіазму

9.4.3. Ролі, орієнтовані на інтелект

Генератор ідей (Plant)

- Характеристики: креативний, винахідливий, неортодоксальний
- Внесок: пропонує нові ідеї та підходи, вирішує складні проблеми

- Допустимі слабкості: ігнорує деталі, занадто зайнятий ефективною комунікацією

Аналітик-стратег (Monitor Evaluator)

- Характеристики: тверезий, стратегічний, проникливий
- Внесок: бачить всі варіанти, точно оцінює ситуацію
- Допустимі слабкості: брак драйву та здатності надихати інших, надто критичний

Спеціаліст (Specialist)

- Характеристики: цілеспрямований, ініціативний, відданий
- Внесок: забезпечує знання та навички в дефіцитній сфері
- Допустимі слабкості: зосереджений на вузькій технічній сфері, ігнорує "велику картину"

9.5. Формування збалансованої команди

Принципи формування збалансованої команди:

1. **Поєднання функціональних ролей**
    - Забезпечення всіх необхідних компетенцій
    - Відповідність ролей потребам проекту
    - Чітке визначення обов'язків та відповідальності
2. **Баланс командних ролей за Белбіном**
    - Представлення різних командних ролей
    - Компенсація слабких сторін однієї ролі сильними сторонами іншої
    - Розуміння природних схильностей членів команди
3. **Урахування особистісних характеристик**
    - Сумісність типів особистості
    - Доповнюваність стилів роботи
    - Узгодження цінностей та підходів
4. **Баланс досвіду та інновацій**
    - Поєднання досвідчених фахівців та нових талантів
    - Забезпечення передачі знань
    - Підтримка інноваційного мислення
5. **Різноманітність та інклюзивність**
    - Різноманітність досвіду та підходів
    - Різні перспективи та точки зору

- Інклюзивне середовище для всіх членів команди