|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Назва методології | Сильні сторони | Слабкі сторони | Для якої галузі є доцільною |
| 1 | **Waterfall** | * Послідовність(Кожен етап проекту має визначені вимоги до переходу до наступного етапу. Це дозволяє підтримувати структуру та порядок у процесі розробки) * Простота керування(Завдяки своїй простоті, може бути добре структурований) * Документація(Створення документації на кожному етапі. Це дозволяє краще зрозуміти вимоги та результати кожного етапу і полегшує комунікацію) | * Лінійність(Не надає можливості налагоджувати зміни під час розробки проекту, якщо на певних етапах виявляться нові вимоги або помилки, то їх важко впровадити безпосередньо в поточному етапі) * Довгий цикл розробки(Весь процес розробки розділяється на послідовні етапи, що може призвести до довгого тривання проекту) * Висока вартість змін(Може призвести до необхідності повертатися до початку після завершення попередніх етапів, що може бути дуже дорогими і затратними по часу) | * Промислове обладнання(вимоги часто задаються відразу на початку проекту і залишаються стабільними протягом тривалого періоду) * Великі інфраструктурні проекти(дороги, мости, аеропорти)(У таких проектах вимоги зазвичай чітко визначені на початку, а сам процес будівництва потребує точної послідовності етапів) * Малий бюджет(може допомогти добре спланувати процес та підтримувати контроль над затратами) |
| 2 | **Scrum** | * Гнучкість(зосередження на ітеративному розвитку продукту у коротких часових рамках, дозволяє команді швидко реагувати на зміни вимог та уточнювати напрямок розвитку продукту під час кожного спринту) * Вдосконалення процесу (щотижневі ретроспективи, під час яких команда оцінює свою роботу та визначає, які зміни можуть покращити розвиток продукту і робочий процес) * Скорочення часу до випуску продукту(Ітеративний підхід дозволяє швидше отримувати функціональний продукт після кожного спринту. Це дозволяє знизити час до випуску на ринок і швидше отримати зворотний зв'язок від користувачів) | * Не для всіх проектів(Є ефективним для проектів з динамічними і невизначеними вимогами, але для стабільнимх та з жорсткими вимогами, може бути менш ефективним.) * Понаднормова робота(Зазначений спринт для розробки продукту може приводити до підсиленого темпу роботи та стресу в команді, особливо у завершальних днях спринту) * Scrum-master (Необхідний досвідчений Scrum-мастер, який здатний правильно налаштувати та керувати процесом. В противному випадку може призвести до некоректного застосування Scrum і недосягнення бажаних результатів) | * Розробка ПЗ(Вимоги до програмного продукту часто змінюються на протязі розробки. Scrum дозволяє залучати замовника на кожному етапі, реагувати на зміни та швидше розробляти функціональний продукт) * Маркетинг та реклама(Може бути застосований у маркетингових кампаніях, де необхідно швидко реагувати на зміни в ринкових умовах, ефективно співпрацювати зі замовниками та впроваджувати нові стратегії) * GameDev(Складна та творча сфера, де важлива швидкість реакції на зміни, тестування та оптимізація продукту. Scrum може бути корисним для управління розробкою ігор та забезпечення якісного релізу) |
| 3 | **Kanban** | * Візуалізація(Кожен етап відображається на дошці, що дозволяє команді зрозуміти, де знаходяться завдання та як швидко вони рухаються) * Контроль робочого навантаження(Дозволяє обмежити кількість завдань, що перебувають на кожному етапі робочого процесу. Це допомагає забезпечити рівномірний розподіл завдань і уникнути перевантаження команди.) * Зосередженість на завершенні завдань(Завдання переходять з етапу в етап із мінімальними затримками, що допомагає підтримувати фокус на завершенні роботи над одним завданням перед переходом до наступного.) | * Відсутність жорсткого графіку(Зосереджений на невеликих ітераціях завдань, він не надає жорсткого графіку з фіксованими термінами, як Scrum. Для деяких команд і проектів це може призвести до відсутності дедлайнів та меншої дисципліни.) * Відсутність детального планування(Зосередження на більш гнучкому підході до управління проектами, що може призвести до меншої деталізації планування, що для деяких команд або проектів це може створювати нестабільність та незабезпеченість у плануванні.) * Відсутність структури(Асобливо для початківців або команд з низьким рівнем самоорганізації, може бути важко управляти процесом через відсутність чіткої структури та правил.) | * Розробка ПЗ як і в Scrum * Технічна підтримка(може бути корисним для організації роботи служб підтримки, де важливо ефективно реагувати на запити та вирішувати проблеми клієнтів у встановлені терміни) * Бізнес(може бути використаний для оптимізації бізнес-процесів та операцій, таких як обробка замовлень, управління запасами, облік та інші операційні процеси.) |

**2lvl**

Основною причиною появи Agile-маніфесту стала необхідність зміни традиційного підходу до розробки програмного забезпечення. На той час існував значний розрив між методами розроблення ПЗ, орієнтованими на детальне планування проєкту, документування та розгортання на великих етапах (традиційні методи, як-от водоспад), та реальними вимогами сучасного динамічного бізнес-середовища, яке вимагає більшої гнучкості та швидкості розроблення продуктів. Маніфест Agile був спробою об'єднати різні підходи до розробки програмного забезпечення.

1. Перелік проблем, які він мав вирішити:

Повільна реакція на зміни (Традиційні методи розробки, такі як Waterfall, передбачали докладне планування і документацію на початку проекту, що ускладнювало зміни вимог та пристосування до змін у вимогах клієнта)

Недостатня співпраця замовника і команди розробників (Традиційні методи розробки можуть створювати бар'єри між замовником і командою розробників через важкий акцент на формальну документацію та організаційні процеси)

Недостатня увага до реальних потреб користувачів (В традиційних методах розробки, акцент часто ставиться на дотримання графіка та бюджету, а не на задоволення реальних потреб користувачів)

Велика кількість відмінень і змін у проекті (У традиційних методологіях великі проекти часто зазнають відмінень та змін, що може вплинути на вартість і терміни реалізації)

1. Чи вирішив проблеми?

Agile не є універсальним рішенням для всіх проблем, але практика такої розробки зробила значний внесок для покращення процесів розробки програмного забезпечення, допомогли збільшити швидкість реагування на зміни та зосередитися на створенні цінності для клієнтів.

**3lvl**

Для процесу розробки мобільного застосунку я обрав методологію Scrum. Ось кілька причин, що обґрунтовують мій вибір:

1. Ітеративний підхід (Scrum дозволяє розробляти продукт у короткі спринти, кожен з яких завершується прототипом або потенційно готовим функціоналом. Цей підхід дозволяє швидше випускати продукт на ринок та отримувати зворотний зв'язок від користувачів)
2. Гнучкість (Дозволяє командам швидко адаптуватися до змін вимог та пріоритетів. Це особливо важливо для стартапів, які працюють у змінному середовищі)
3. Планування (Допомагає створити прогнози стосовно термінів та результатів на кожен спринт, що дозволяє керувати проектом більш ефективно.)
4. Мотивація команди (Кожен спринт має конкретні цілі, і команда спрямовує свої зусилля на досягнення цих цілей, що може підвищити задоволення від роботи.)
5. Зниження ризиків (Ітеративний підхід допомагає виявляти ризики та проблеми раніше в процесі розробки)

Але варто не забувати, що незважаючи на ці переваги успіх будь якої методології залежить від правильної організації та комунікації)