## Урок 3. Домашнє завдання

Рівень І

| **Методологія** | **Плюси** | **Мінуси** | **Галузь застосування** |
| --- | --- | --- | --- |
| Waterfall | 1. Чітка послідовність (дає можливість розробити план проекту та поліпшує підбір команди) 2. Оцінка витрат та термінів до початку проекту (усі вимоги чітко проговорюються на початковому етапі і не змінюються під час усього процесу, це дає можливість точно оцінити майбутні витрати) 3. Документація кожного етапу (надає можливість надати замовнику звітність у будь-який момент, дозволяє створити базу для інших проектів) 4. Постійний контроль процесу та передбачуваність (цілі та задачі проекту чітко зрозумілі для розробників) | 1. Складно виправити помилки (тестування відбувається на останніх етапах розробки) 2. Висока вартість виправлення помилок (будь-яка помилка призведе до необхідності переробляти весь проект) 3. Постійна актуалізація документації (надлишкова документація) 4. Відсутність зворотного зв’язку з замовником (кінцевий результат може повністю не задовольнити замовника) 5. Відсутність можливості ознайомитися з системою заздалегідь ( користувач не має можливості звикати до продукту поступово) | Для реалізації проектів життєзабезпечення, де будь-яка помилка може призвести до фатальних наслідків (мед. техніка).  Завдяки суворим вимогам до виконання проекту Waterfall може бути використаний для розробки проектів у військовій чи авіаційній організації.  У державних проектах, де необхідними є сувора звітність та дотримання графіку. |
| V-Model | 1. Зменшення ризиків та покращення якості (завдяки тестуванню на ранніх етапах) 2. Зменшення можливих витрат (вартість усіх етапів проекту прогнозується і не змінюється) | 1. Недостатня гнучкість у процесі розробки (розробка починається тільки після переходу до іншого етапу, ніяких попередніх шаблонів не передбачено) 2. Складність виправлення фундаментальних помилок (відсутній алгоритм дій при виявленні помилок на етапі тестування) | У сферах, де роботу проекта неможливо зупинити. Де має відстеживатися кожна помилка. такий процес потребує багато часу та високого рівня професійності від виконавців (розробка авіаційного/військового/ ПЗ) |
| Kanban | 1. Гнучкість планування та безперервність процесу (після кожної ітерації команда видає готовий продукт з покращеною функціональністю. Це надає можливість вносити зміни в процес: передивитися пріоритети розробки, цілі, задачі) 2. Візуалізація процесу (завдяки дошці, можна побачити хто, які задачі та в якому обсязі вирішує, що спрощує пошук недоліків та багів. Дошка показує на якому з етапів знаходиться процес та стимулює співпрацю між членами команди) 3. Обмежена кількість виконуємих завдань (завдань рівно стільки, скільки може виконати команда, тим самим допомагає скоріше виявити недоліки у функціональності і вказує куди слід звернути уваги) | 1. Тільки для маленьких монопрофільних команд ( вимагає максимальної участі всіх членів команди, усі учасники процесу мають знати функціі один одного та мати високий кваліфікаційний рівень) 2. Неприйнятний для довгострокового планування (ця методологія передбачає конкретний наочний результат у зазначений термін) | Ця методологія відмінно себе показує в сферах неосновного виробництва: групи підтримки ПЗ, під час управління стартапами без чіткого плану, де потребується активна розробка, в HR проектах.  Також ефективно працює в IT галузі, в галузі будівництва, в сфері закупів. |

Рівень ІІ

Більшість проектів не мають 100% розуміння, яким має бути кінцевий результат, через це визначитись з точними завданнями просто неможливо.

До виникнення Agile, компанії зосереджувалися на надлишковому плануванні та документуванні своїх циклів розробки, а не на бажаннях та можливостях своїх замовників. Цінності, які рекламувались компаніями ніяким чином не допомагали команді, а особливо розробникам рухатися у напрямку до кращого шляху співпраці.

Agile-маніфест з’явився завдяки тому, що галузь програмування розвивалася дуже стрімко, ринок розробки ПЗ швидко змінювався, з’являлося все більше нових та гнучкіших методологій розробки ПЗ, які намагалися підлаштуватися під постійно змінювані вимоги замовників та очікування користувачів, але попри це не було єдиного шаблону/прототипу, під який могли б адаптуватися всі ці методи.

Маніфест з’явився, коли виникла необхідність об’єднати нові, більш гнучкі методології розробки під якимось єдиним гаслом, не звертаючи увагу на різницю поглядів, підходів або інструментів. Для цього були виділенні цінності та основи, які є загальними для всіх.

Цей маніфест створений, та залишається закликом до всіх учасників галузі розробки ПЗ. Основи маніфесту мали вирішити основні проблеми компаній, а саме звернути увагу на те, що: люди, та зручність їх праці та взаємодії важніше правильності процесів; зосередженість на створенні нової версії діючого ПЗ або продукту є важливіше зосередженості на створенні документаціі; розмовляти із своїм клієнтом задля розуміння того, що конкретно він хоче важніше, ніж мати розбіжності через формулювання задач; на шляху розробки треба бути відкритим до змін.

З моменту виникнення маніфесту пройшло більше, ніж 20 років. За цей час кількість розробників, які під час розробки користуються Agile-підходами зросла до 85%. За 2021 рік кількість команд, які використовують методологію Agile зросла з 37% до 86%. Це говорить про те, що застосування гнучкої методології дійсно є ефективним.

Виходячи з динамічності змін, які відбуваються у світі (Covid, війна), та враховуючи той факт, що впровадження та використання методології зростає з кожним роком (94% компаній застосовували методологію в своїй роботі, 65% компаній використовують методологію від одного року і більше), з моєї точки зору поява та подальше використання маніфесту допомогло вирішити основні проблеми, які були наведені в основах самого маніфесту.

Джерело статистики: <https://info.digital.ai/rs/981-LQX-968/images/SOA15.pdf>

Рівень ІІІ

Мобільний застосунок має бути розроблений за визначений термін (1-2 місяці), застосунок буде постійно оновлюватись (щоб відповідати вимогам ОС смартфонів), кількість користувачів буде зростати, пропорційно цьому буде зростати кількість завантажених світлин. З огляду на ці вимоги, я би запропонувала методологію Скрам або Канбан, або навіть одночасно ці два методи. Чому?

Тому що- це стартап, застосунок має визначений термін реалізації. Планування має бути виконане з урахуванням можливих змін. Необхідно максимально швидко розробити прототип і випустити його на ринок задля тестування користувачами, в процесі виправляючи баги або недоліки. Цей стартап не є масштабним продуктом, тому кількість людей в команді повинна бути обмежена.