**SDLC та STLC. Методології розробки ПЗ.**

**Перший рівень** — відпрацюй навички на базовому рівні.

Склади порівняльну таблицю найбільш поширених методологій:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Назва методології | Сильні сторони | Слабкі сторони | Для якої галузі є доцільною |
| 1 | Waterfall | **-** Деталізація кожного кроку роботи, що супроводжується документуванням.  **-** Вимоги максимально виразно і чітко викладені, не можуть суперечити одна одній або змінюватися в середині роботи.  **-** Легкість розуміння методики як такої навіть для не найдосвідченіших розробників.  **-** Легкість контролю і, за необхідності, передачі проєкту іншій команді, завдяки суворій системі звітності.  **-** Можливість заздалегідь знати, скільки часу і грошей буде витрачено на проєкт. | **-** Відсутня можливість для маневру, якщо в процесі розробки з'ясувалося, що продукт не відповідає вимогам ринку.  **-** Висока ймовірність виявлення критичних проблем уже на завершальному етапі розроблення, причому їхнє усунення на етапі готового продукту обходиться надмірно дорого.  **-** Відсутня можливість для маневру, якщо в процесі розробки з'ясувалося, що продукт не відповідає вимогам ринку. | **-** Може бути використана у створенні ПЗ для різних сфер діяльності: автомобілебудування, будівництва будівель і споруд, фінансового сектору, медицини та електроніки.  **-** Якби ми будували будинок, було б смішно сказати:  "Ми заллємо частину фундаменту, зведемо на цій ділянці стіну і подивимося, як це буде, а потім вирішимо, як закінчити решту будинку". В основному всі етапи будівельних робіт виконуються поступово один за одним.  Це приклад Waterfall методології. |
| 2 | V-model | Строгі етапи. Планування, тестування і верифікація самої системи проводиться на ранніх етапах. Пишемо раннє тестування. Є такий етап, як проміжне тестування. Покращений, порівняно з каскадною моделлю, тайм-менеджмент. | Не гнучкість моделі. Саме написання і створення програми відбувається на етапі написання коду, грубо кажучи, на середині процесу розробки. Немає можливості динамічного внесення будь-яких змін, тому що всі етапи йдуть один за одним. Недостатній аналіз ризиків. | Зазвичай використовується в проєктах, у яких існують часові та фінансові обмеження, і для таких завдань, які передбачають більш широке, порівняно з каскадною моделлю, тестове покриття. |
| 3 | Iterative model | Раннє створення працюючого ПЗ, оскільки ітерації зазвичай невеликі за часом, ми можемо доставити якусь частину функціональності нашого продукту за невеликий проміжок часу і дуже швидко. Такі системи зазвичай дуже гнучкі, тобто вони готові до модифікації вимог на будь-якому з етапів розробки. Завжди простіше провести тестування та аналіз ризиків саме для такої невеликої частини нашого життєвого циклу продукту. | Кожна фаза це самостійна якась одиниця, і окремі ітерації одна на одну не накладаються. Не завжди всі вимоги відомі до початку проєктування, це може ускладнити роботу і реалізацію загальної архітектури нашої системи. | Добре підходить для великих проєктів, коли нам відомі не всі вимоги, але відомі хоча б якісь ключові milestone, вимоги. І коли вимоги будуть змінюватися в процесі розробки. |

**Другий рівень** — детальніше заглибся в практику.

1. Виконай завдання попереднього рівня.

2. Напиши розгорнуті відповіді (0,5 - 1 сторінки тексту) на такі два питання:

* На твою думку, чому з’явився Agile-маніфест?
* Які проблеми він мав вирішити і чи це вдалося?

Раніше під час реалізації проєкту в життя, застосовувалася модель розробки Waterfall. У водоспадній моделі всі етапи розробки йшли один за одним. З початку, наприклад, був збір ідей, написання вимог, потім йшла розробка, потім йшло тестування, і ми не могли повернутися на попередній етап, якщо ми вже перебуваємо на якомусь із них. Ми повинні були обов'язково починати все спочатку, що призводило до збільшення вартості та термінів виконання нашого продукту. Тобто, при використанні технології Waterfall потрібно продумати все до дрібниць, що дуже ускладнює роботу спеціалістів. А іноді й робить її неможливою. Чим хороші гнучкі методи? Тим що ми можемо змінювати наші вимоги, ми можемо повертатися до попередніх етапів нашого розроблення ПЗ без взаємодій з іншими етапами. Тому 2001 року 17 розробників зібрали всі свої зауваження, як вони працюють із гнучкими методами в розробці ПЗ та написали Agile-маніфест. Метод Agile пропонує гнучкий підхід до управління проєктами. Це так званий ітеративний підхід в управлінні, коли ми робимо невеликі прирости. Ми йдемо по кроках, етапах. Гнучка методологія розробки передбачає поділ проєкту на короткі етапи - ітерації. У розробників з'являється ретроспектива наприкінці кожної з них. Якийсь відгук від клієнта. У такому разі, якщо замовника щось не влаштує, легко можна буде внести правки, не чекаючи кінцевого продукту.

Головні цінності Agile це:

- Люди і взаємодія важливіші за процеси та інструменти. Тобто, якщо в нашій команді є якісь принципи, структури, інструменти, умови, які можуть завадити роботі, то їх необхідно позбутися. Найголовніше, що люди мають самі обирати спосіб організації, набір якихось процесів, інструментів, і саме ці інструменти та процеси мають допомагати людям у роботі, а не заважати їй. Тому насамперед звертаємо увагу на людей і взаємодію.

- Наступна цінність це те, що працюючий продукт важливіший за вичерпну документацію.

Тобто, це не означає, що не повинно бути ніякої документації на проєкті, але ця документація не повинна бути надлишковою, і на неї не потрібно витрачати величезну кількість часу й ресурсів, головне - доставити працюючий продукт, функціональність безпосередньо замовнику.

- Співпраця із замовником важливіша за узгодження умов контракту. Це означає, що не повинно бути надлишкових прив'язаностей до наших контрактів. Якоїсь документації яка регулює нашу взаємодію із замовником. Ми повинні насамперед звертати увагу на те, що б нам було комфортно співпрацювати один з одним. Ми не хочемо псувати стосунки з нашим замовником і будь-які наші документи, угоди, контракти та інше повинні сприяти нашим позитивним взаєморозумінням з клієнтом, але ніяк не псувати їх.

- Готовність до змін важливіша за дотримання початкового плану. Ми завжди повинні мати на увазі, що ми з часом все рівно будемо вносити якісь зміни.

Важливо пам'ятати про те, що завжди в Agile є зміни і, в принципі, для цього і була створена ця система.

Крім цих цінностей, в Agile є 12 принципів, які уточнюють ці цінності.