**Beet Seed**

| **№** | **Назва методології** | **Сильні сторони** | **Слабкі сторони** | **Для якої галузі є доцільною** |
| --- | --- | --- | --- | --- |
| **1** | **Agile** | 1. Програмне забезпечення має мінімум дефектів завдяки ітеративним зусиллям у тестуванні та доопрацюванні. 2. Зрозумілість між членами команди під час розробки завдяки частій та прозорій розробці. 3. Зміни у вимогах до проекту легко вносяться та опрацьовуються з незначним впливом на графік. 4. Загальне покращення якості результату. | 1. Іноді команда може втратити фокус через надмірну кількість запитів на зміни. 2. Документація в Agile відходить на другий план, що може стати проблемою на більш пізніх етапах розробки. 3. Agile фокусується на обговореннях та зворотному зв'язку, що може забирати багато часу у команди. 4. Через неструктурований підхід, Agile потребує досвідчених розробників, які можуть працювати незалежно. | Методологія Agile ідеально підходить для проектів з вимогами, що швидко змінюються. Якщо ви створюєте програмне забезпечення для нової ніші, вам варто використовувати Agile. Він найкраще підходить для реалізації додаткових ідей у міру того, як ви дізнаєтеся більше про потреби ринку. Звичайно, це передбачає, що ваша команда розробників має високий ступінь незалежності та комфортно працює у швидкозмінному, неструктурованому середовищі. |
| **2** | **Waterfall** | 1. Лінійність моделі водоспаду робить її легкою для розуміння, особливо для нових розробників. 2. Всі специфікації та результати прописані до початку розробки. 3. За допомогою моделі водоспаду немає місця для неправильної передачі інформації, оскільки вона чітко визначена на кожному етапі. | 1. Вона не передбачає зворотного зв'язку з клієнтами на ранніх етапах, що збільшує ризик відхилення проекту від запланованого. 2. Тестування виконується лише наприкінці розробки. Деякі проблеми складніше виправити на пізніших етапах. 3. Жорсткість водоспадної моделі не залишає місця для змін, що робить її непридатною для складних проектів. 4. Команда може витрачати занадто багато часу на документацію замість того, щоб створювати рішення, які вирішують проблеми користувачів. | Використовуйте Waterfall лише тоді, коли у вас є проект з чітко визначеним обсягом робіт. Він не підходить для розробки з великою кількістю невідомих. Waterfall ідеально підходить для проектів з передбачуваним результатом і коли у вас є команда недосвідчених розробників. |
| **3** | **Lean** | 1. Скорочує втрати в проекті, такі як надлишкові коди, непотрібна документація та завдання, що повторюються. 2. Загальна вартість розробки знижується завдяки принципам Lean розробки. 3. Час виходу програмного забезпечення на ринок скорочується, оскільки ощадлива розробка сприяє підвищенню ефективності. 4. Підвищується мотивація членів команди, оскільки вони отримують більше повноважень для прийняття рішень. | 1. Для успішної Lean розробки вам знадобиться команда висококваліфікованих розробників, яку непросто зібрати разом. 2. Менш кваліфіковані розробники можуть бути перевантажені обов'язками і втратити фокус на проекті. 3. Необхідна детальна документація, що створює величезне навантаження на бізнес-аналітика. | Lean розробка ідеально підходить для невеликих проектів з обмеженим бюджетом. Його принципи мінімізації втрат і підвищення ефективності означають, що ви будете використовувати невелику команду для досягнення приголомшливих результатів. Lean розробка стає менш практичною для великих проектів, оскільки вам знадобиться більша команда для виконання завдань. |
| **4** | **Scrum** | 1. Короткі ітерації дозволяють швидко вирішувати проблеми. 2. Скрам дуже швидко реагує на зміни, оскільки процес передбачає регулярний зворотній зв'язок. 3. Скрам економічний та ефективний. 4. Регулярні зустрічі гарантують, що члени команди завжди знаходяться на одній сторінці. 5. Внесок кожного члена команди помічають і цінують завдяки Scrum-зустрічам. | 1. Для того, щоб скрам працював, усі члени команди мають бути однаково кваліфікованими та відданими своїй справі. 2. Щоденні зустрічі Scrum можуть бути виснажливими для членів команди. 3. Може збільшити час виходу на ринок, якщо немає суворого контролю над дедлайном. 4. Не підходить для великих проектів. | Скрам - це ідеальна методологія, якщо у вас є проект з нечіткими вимогами, але він повинен адаптуватися до частих змін. Наприклад, вам потрібно швидко створити MVP і протестувати його серед користувачів. Пам'ятайте, що Scrum ефективний лише тоді, коли у вас є повністю віддана та досвідчена команда. |

**Beet Sprout**

**Чому з’явився Agile-маніфест?**

Хоча різні принципи гнучкого підходу існують ще з 1970-х років, сам маніфест - і повне визначення філософії гнучкого підходу - був створений на зорі нового тисячоліття.

На початку 2001 року група з 17 розробників провела дві зустрічі - першу в Орегоні, другу в Сноуберд, штат Юта, - щоб обговорити проблеми та рішення в розробці програмного забезпечення, і саме так з'явився маніфест.

Простіше кажучи, маніфест був написаний як відповідь на велике розчарування традиційними процесами розробки 1990-х років.

Вибух персональних комп'ютерів означав, що розробка продуктів і програмного забезпечення зазнала значних змін, і розробники на зустрічах - та й у всій індустрії - відчували, що статус-кво більше не працює.

Розрив між потребами бізнесу та рішеннями, що розроблялися, становив у середньому три роки, а стандартні процеси на той момент були громіздкими, незадовільними, перевантаженими документацією та наглядом.

17 розробників, які зустрілися в Орегоні та Юті, назвали себе "Agile Alliance" і запропонували новий спосіб роботи, заснований на наборі цінностей і принципів, які "відновлять довіру до слова "методологія"".

Маніфест був розроблений, щоб розширити можливості розробників, пришвидшити процеси та сприяти заохоченню робочих практик, які більше зосереджені на користувачеві.

Цінності та принципи дозволяють командам бути адаптивними, швидко та ефективно реагувати на зміни, а також перебувати у стані постійного пошуку нових рішень.

**Які проблеми він мав вирішити і чи це вдалося?**

У лютому 2021 року виповнилося 20 років з дня прийняття Agile Маніфесту. З моменту його прийняття світ кардинально змінився - як з точки зору суспільства, так і з точки зору того, як ми працюємо. Незважаючи на це, Маніфест Agile все ще актуальний. Хоча концепція вже не нова, вона все ще керує багатьма нашими практиками та стратегіями. Але чи досягла вона того бачення, на яке сподівалися її творці? І так, і ні.

Дехто вважає, що ми далекі від досягнення бачення, викладеного в оригінальному Маніфесті Agile.

Особливо це стосується бізнесу корпоративного рівня, який, за словами Брайана Доусона, євангеліста DevOps з CloudBees, схожий на "танкер", який повільно і поступово розвертається.

На жаль, багато компаній все ще ставляться до "гнучкості" як до модного слова. Вони впроваджують Agile підходи в офісі і очікують, що всі підхоплять їх, але потім мало що роблять для впровадження та підтримки Agile практик.

Вони намагаються впровадити Agile в існуючі процеси замість того, щоб будувати нові процеси на основі принципів гнучкості. Вони розглядають Agile як ліки для вирішення проблем компанії, коли вони просто накладають пластир і сподіваються на краще.

Підприємства корпоративного рівня також розглядатимуть Agile як суто пов'язану з розробкою продукту. Вони впроваджуватимуть Agile лише у своїх відділах і сподіватимуться, що це призведе до підвищення якості продукції та збільшення обсягів виробництва. Але, як ті, хто знайомий з Agile, вже знають, що принципи гнучкого підходу повинні бути прийняті на організаційному рівні, щоб отримати вигоду з його справжньої цінності.

Існує тенденція вписувати Agile у жорсткі процеси. Дехто бачить Agile як простий процес, який можна одразу впровадити, і починає називати бізнес гнучким.

Це призводить до того, що процеси створюються заради процесів, які лише роз'єднують бізнес та його працівників. Розглядаючи Agile як жорсткий процес, ви втрачаєте майже всі переваги роботи з Agile.

Але, в той же час, існує тенденція занадто далеко відійти від жорсткості. Дехто вважає, що Agile - це вільна методологія, яка дозволяє командам безцільно плисти за течією. Хоча іноді це може давати чудові результати, обманюючи компанії, що їхня версія Agile працює, це просто призводить до відсутності дисципліни в командах. Agile-коуч може здатися непотрібною витратою, але його наявність необхідна для того, щоб ваша команда продовжувала наполегливо працювати, не забуваючи при цьому про Agile-практики.

На щастя, стартапи набагато краще сприймають цінності Agile-маніфесту. Оскільки вони ще не мають усталених процесів, вкорінених у культурі компанії, вони можуть будувати свої практики на основі принципів гнучкості, щоб вони були на першому плані під час всього, що вони роблять.

Це чудова новина для прихильників Agile підходів, адже сьогоднішні стартапи стануть завтрашніми підприємствами.

Тож, хоча ми ще не зовсім там, ми, безумовно, на шляху до досягнення бачення Agile Маніфесту.