Корпоративно-информационные системы
Новиков Александр Константинович
631 группа
Эссе №2. Методика Agile
Тема. «Как я понимаю методику agile»

Эссе

Общее представление об Agile

Простыми словами методику Agile можно описать, как специальный подход к управлению проектами, позволяющий ускорить функционирование сотрудников, тем самым улучшив сервис для конечного клиента.

Давайте разберём методику Agile на примере компании, занимающейся перевозкой товаров. Для простоты пусть эта компания перевозит посылки только между физическими лицами. В качестве продукта компании в таком случае выступает сервис, предоставленный пользователям для осуществления отправки или получения товаров.

Над созданием, обеспечиванием функционирования и улучшением сервиса трудится некоторая команда сотрудников во главе с руководителем (он же team-lead). Владелец компании или руководящее лицо распределяет роли внутри команды.

Методика Agile чаще всего подразумевает, что ведётся сбор мнений и пожеланий клиентов относительно создаваемого продукта, в нашем случае, сервиса. Пусть в данном примере это выражается через систему отзывов на сайте компании, где любой человек может оставить пожелание, высказать свои замечания и недовольства или же похвалить ту или иную часть предоставляемого сервиса.

Также предположим, что в команде рассматриваемой компании есть один или несколько сотрудников, чья задача заключается в тестировании сервиса, поиске недостатков, придумывании улучшений и т.д.

Разумеется, в компании есть основная группа – группа разработчиков сервиса, состоящая из людей, хорошо знающих все технические тонкости и выполняющих поставленные руководителем или владельцем задачи. Также есть непосредственно исполнители – люди, осуществляющие доставку, но их мы рассматривать не будем, так как они выполняют функцию инструмента, который разрабатывается командой.

Упрощённая система функционирования данной компании выглядит следующим:

1. Сбор проблем и пожеланий. Клиент пишет отзыв с найденными недостатками или желаемыми улучшениями на сайте компании либо же человек, отвечающий за тестирование указывает что-либо в отчёте руководителю.
2. Фильтрация. Само собой, не всё, что удалось собрать необходимо выполнять. Самое сложное в этой фазе – сказать «нет», то есть, осознав ответственность, исключить из рассмотрения ту или иную проблему или потенциальное улучшение.
3. Постановка задач. Пожелания и недостатки всегда сформулированы очень неточно, поэтому необходимо обработать их и трансформировать в чёткую задачу для разработчиков.
4. Расстановка временных приоритетов. Необходимо решить, в какой последовательности начать выполнять появившиеся задачи. Для этого можно построить диаграммы «Делать вещи правильно», «Делать правильные вещи», «Делать быстро», расставить задачи и сгенерировать последовательность.
5. Последовательная подача задач команде разработчиков. В данную фазу также следует отнести тестирование, отладку и доработку, подразумевая, что как

результат разработчики выполнят все поставленные задачи корректно (разумеется, в реальности так может и не быть).

Система Agile подразумевает осуществление взаимодействия и доверительного общения внутри команды, когда все участвуют в улучшении сервиса.

Рассмотрим самые важные компоненты описанной выше системы более подробно.

Принятие решения «Нет» и расстановка приоритетов. Задача может быть легкой (на 1-2 человека часа), а может быть очень трудоёмкой (на месяцы человека часов). И первый вопрос, возникающий в голове руководителя (в общем случае человека, принимающие подобные решения): «Как найти отношение размера задачи и её полезности?». Ответ: «Никак», так как больше – не значит лучше. Для этого необходимо очень тщательно проанализировать проблему, послужившую причиной появления задачи, поэтапный ход решения задачи, затрачиваемые ресурсы и последствия получения или не получения результата. Не стоит забывать также про закон Хофштадтера: *«Любое дело всегда длится дольше, чем ожидается, даже если учесть закон Хофштадтера.»*.

Для более точного определения ценности задачи необходимо не только отличные знания в проблематике, но также и регулярное общение с командой, клиентами и прочими так или иначе заинтересованными лицами. Поэтому регулярно могут организовываться встречи всей команды с заинтересованными лицами для улучшения понимания проблематики и генерации решений поставленных задач.

Также всегда нужно держать баланс между краткосрочным и долгосрочным планированием, не уходя в ту или иную крайность, ведь это чревато сильной нерациональностью в распределении работы и функционировании в целом. Необходимо постоянно соблюдать баланс между реактивной и проактивной работой.

***

Разновидности Agile

Их существует три: Scrum, Kanban и XP. Вкратце рассмотрим каждую из них.

Scrum.

- Создание множества приоритетов (резерва продукта или сервиса)
- Начало спринта. Планирование. Определение основного подмножества пожеланий клиента (отставание в спринте) и решение, как будет реализовано это подмножество.
- Команде даётся определённое время (спринт) на реализацию плана.
- Ежедневные встречи с оценками прогресса.
- Готовность работы для клиента под конец спринта.
- Завершение спринта обзором и ретроспективой.
- Начало нового спринта.

Kanban

- Визуализация рабочего процесса (на стене или на сайте) в виде карты из состояний и этапов рабочего процесса.
- Выставление ограничений работы.
- Выставление ограничений по незавершённости (делали 5, смогли 3, значит в следующий раз начнём делать 3)
- Управление потоком, отслеживая пошаговый прогресс и изменения в процессе выполнения работ.
- Ясность процесса. Чёткое понимание для достижения рационального объективного консенсуса в отношении изменений.

- Улучшение совместной работы. Суть данного метода в постоянном взаимодействии и сотрудничестве команд, а также в понимании целостности.

XP (экстремальное программирование)

Простота. Коммуникация. Обратная связь. Уважение. Смелость.

Принципы:

1. Планирование работы над проектом
2. Малые релизы. Обновление состояния каждые две недели. (Biweekly meetings)
3. Общая методология. Общие названия проектов и задач. Общие условия общения.
4. Простой дизайн.
5. Тестирование.
6. Рефакторинг. Улучшение дизайна на каждом этапе, а не в конце финального.
7. Парное программирование. Пишут два человека на одном ПК.
8. Коллективная собственность. Код принадлежит команде.
9. Непрерывная интеграция. Синхронизация по стадии разработки.
10. 40-часовая неделя.
11. Клиент в доступности. Необходимо постоянно собирать данные о недостатках и необходимых улучшениях.
12. Единый стандарт кодинга.