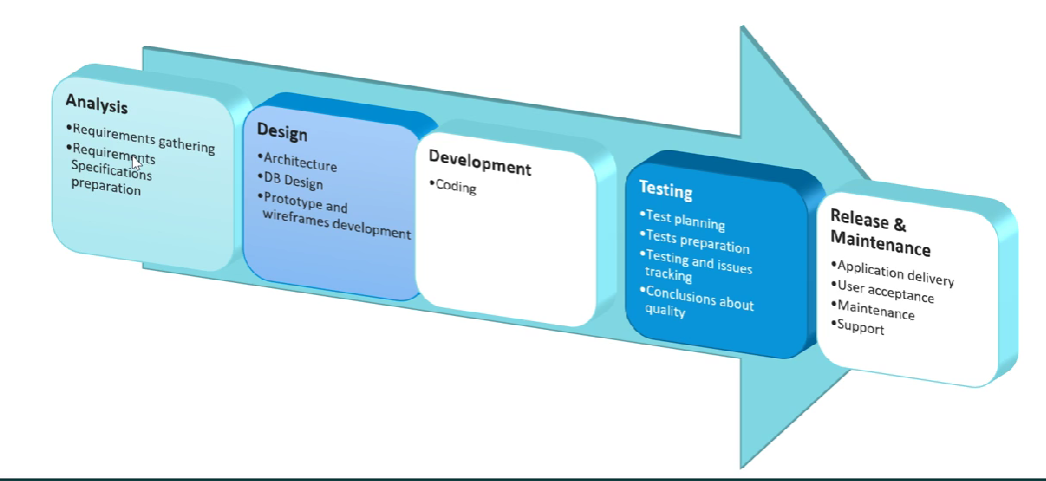
**Waterfall**

Одна из самых старых методологий, линейная технология

Минусы: тестирование только после написания кода (ошибка может жить на протяжении долгого времени, если она была сделана в начале написания кода) => эта методология растягивает время на разработку продукта, много документации

Плюсы: очень простой

## Ag[дж]ile

Не говорит, что делать – говорит, как делать (императивный подход)

Любой процесс == цель

Во главе стоит цель, если инструмент или процесс тормозит разработку – его надо выкинуть

Рабочий софт важнее исчерпывающей документации

Взаимодействие с клиентом важнее контракта (клиенто-ориентированность)

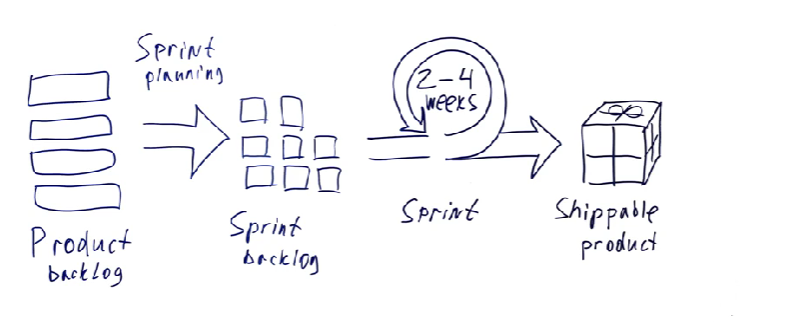
Реакция на изменения (подстраивание под меняющиеся условия)

Принципы:

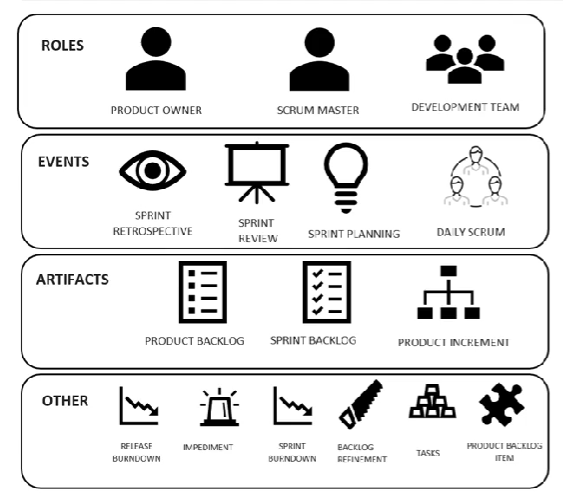
* Высший приоритет – удовлетворить заказчика с помощью раннего и качественного релиза софта
* Изменения в требованиях при необходимости
* Доставлять изменения часто (неделя – 4 недели)
* Пользователи и разработчики должны работать вместе и ежедневно
* Строить проект среди тех, кто заинтересован и кому это важно
* Самый эффективный способ общения – лично, лицом к лицу
* Рабочий софт – мерила прогресса
* Устойчивая разработка
* Уделять внимание хорошей практики и дизайну
* Простота – она нужна
* Команда должна работать эффективно
* Самоорганизация команд

## Scrum (построена на основе Agile)

Мы хотим видеть…(наши желания) -> (выбираем одну из задач)->(разбиваем на подзадачи)->спринт -> проходит итерация (может неделя, может 4…) -> получаем продукт



В отличии от waterfall, если мы хотим что-то добавить или изменить, например, в условиях меняющегося рынка, мы сможем сделать это за один спринт, который, условно, займёт две недели.



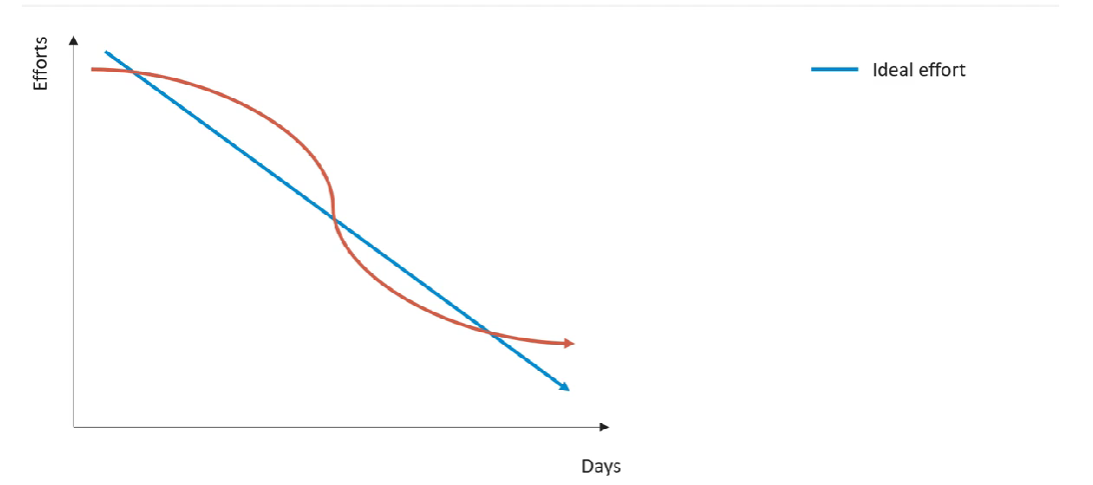
Роли:

* Владелец продукта – поставляет истории для разработчиков, принимает решения, устанавливает приоритеты, планирует (что брать на спринт), управляет бэклогом, общается с остальной командой.
* Скрам-мастер – отвечает за процесс разработки, как правило, один из разработчиков в команде, помогает с точки зрения процесса
* Команда разработки – как правило, небольшая команда (5-7 человек, обычно до 9), организованы самостоятельно

Мероприятия:

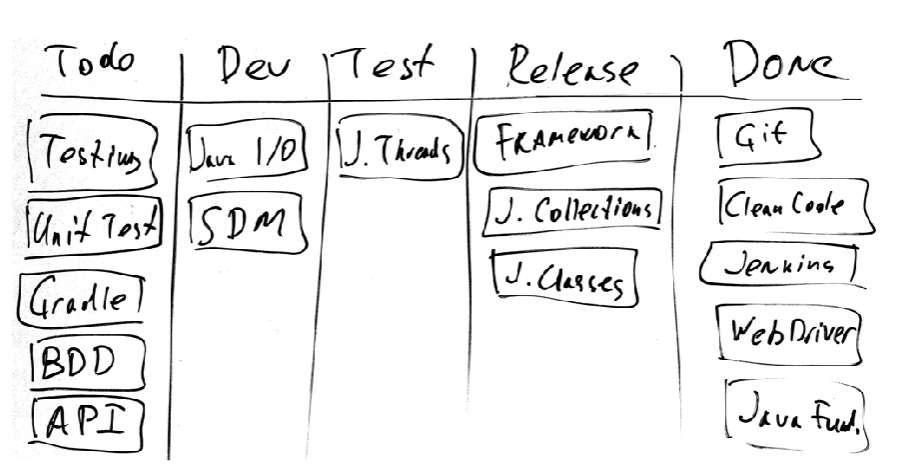
* Выбираем, что будет в спринте (команда определяет требования к story’s)
* Планирование
* Ежедневные небольшие собрания (рассказывают, что сделали и какие есть проблемы), занимает 5-20 минут
* Sprint review / Demo – получаем фидбек от клиента (product owner)
* Ретроспектива – что было классно в спринте, что можно улучшить

Burndown chart



## Kanban

Kanban-доска (board) – главный элемент методологии, на которой находятся задачи (story’s)



Не регламентирована работа, нет чётких правил, можно адаптироваться под условия.

Чем канбан полезен? Когда не надо строить долгосрочные планы, визуально видны задачи, легко очень начать.

Для кого он полезен:

* При поддержке продукта (support)
* Много изменений в спринте
* Присутствует не постоянная команда (переход команды из проекта в проект)

## XP(Extreme programming)

* Planning game (несколько человек планируют)
* Короткие релизы
* Метафора (сложные вещи объясняются простыми вещами и примерами)
* Простой дизайн
* Тестирование (юнит и автоматизированная)
* Рефакторинг (меняется структура, но не назначение кода)
* Парное программирование
* Коллективная разработка
* Непрерывная интеграция
* 40-часовая рабочая неделя (овертаймы могут быть, но не постоянно)
* Когда product owner находится рядом с developers – это хорошо
* Использовать стандарты программирования

## Test driven development

Вначале делаем тесты, а потом делаем код. Это один из принципов экстремального программирования.

Создаём тесты -> делаем так, чтобы тесты проходили -> делаем рефакторинг

Какая польза:

* Чистый и читаемый код
* Безопасный рефакторинг
* Постоянная проверка
* Снижение технических долгов (то, что откладывается в долгий ящик)
* Меньше дефектов

Из минусов:

* Тяжелее работать
* Больше ресурсов и времени

## BDD(Behaviour-driven development)

Создаёт тесты product owner, а не developer, это и отличает этот подход от TDD. Используется язык джеркен (преобразовывается с человеческого языка на понятный программисту)

