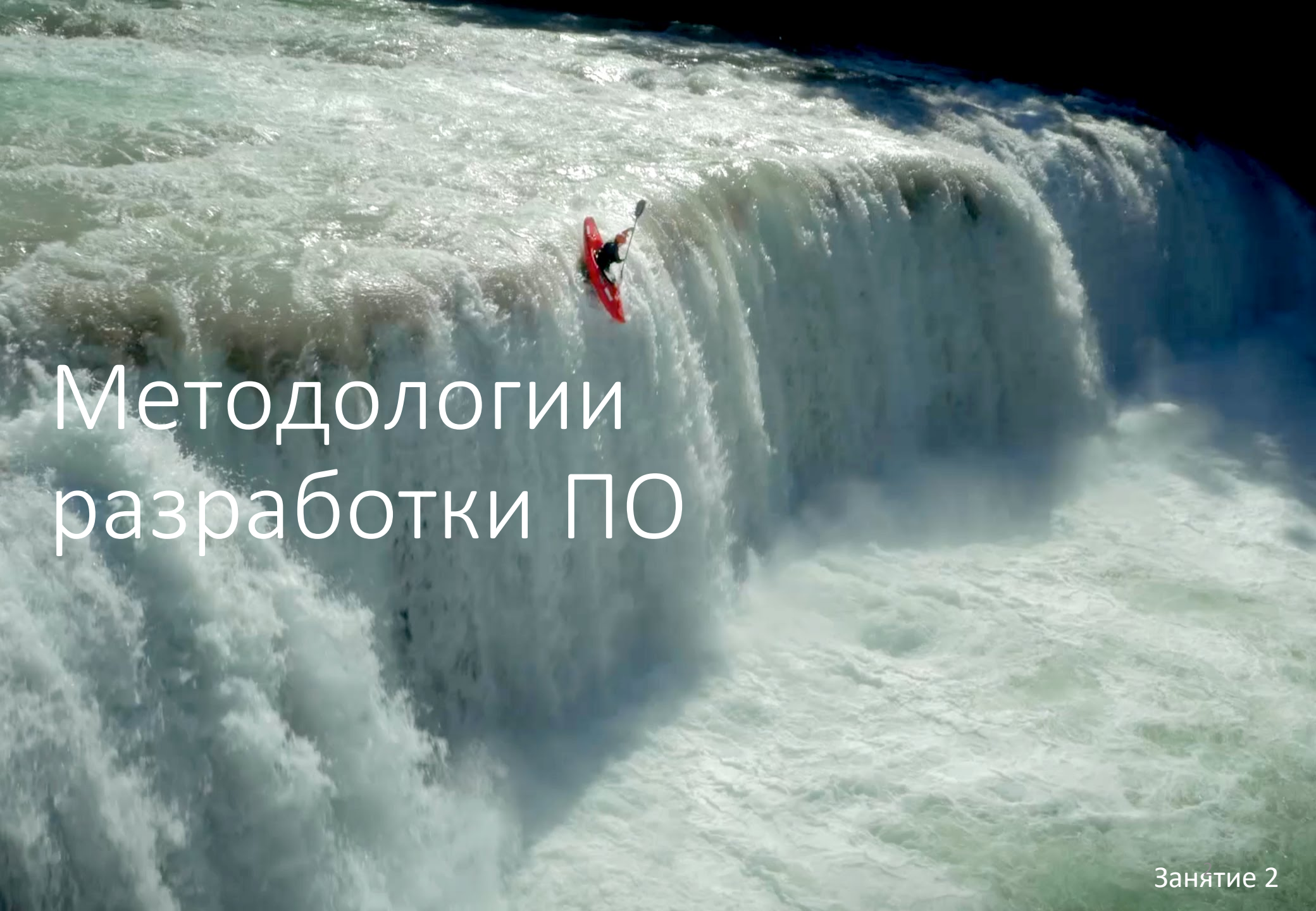


Методологии разработки ПО

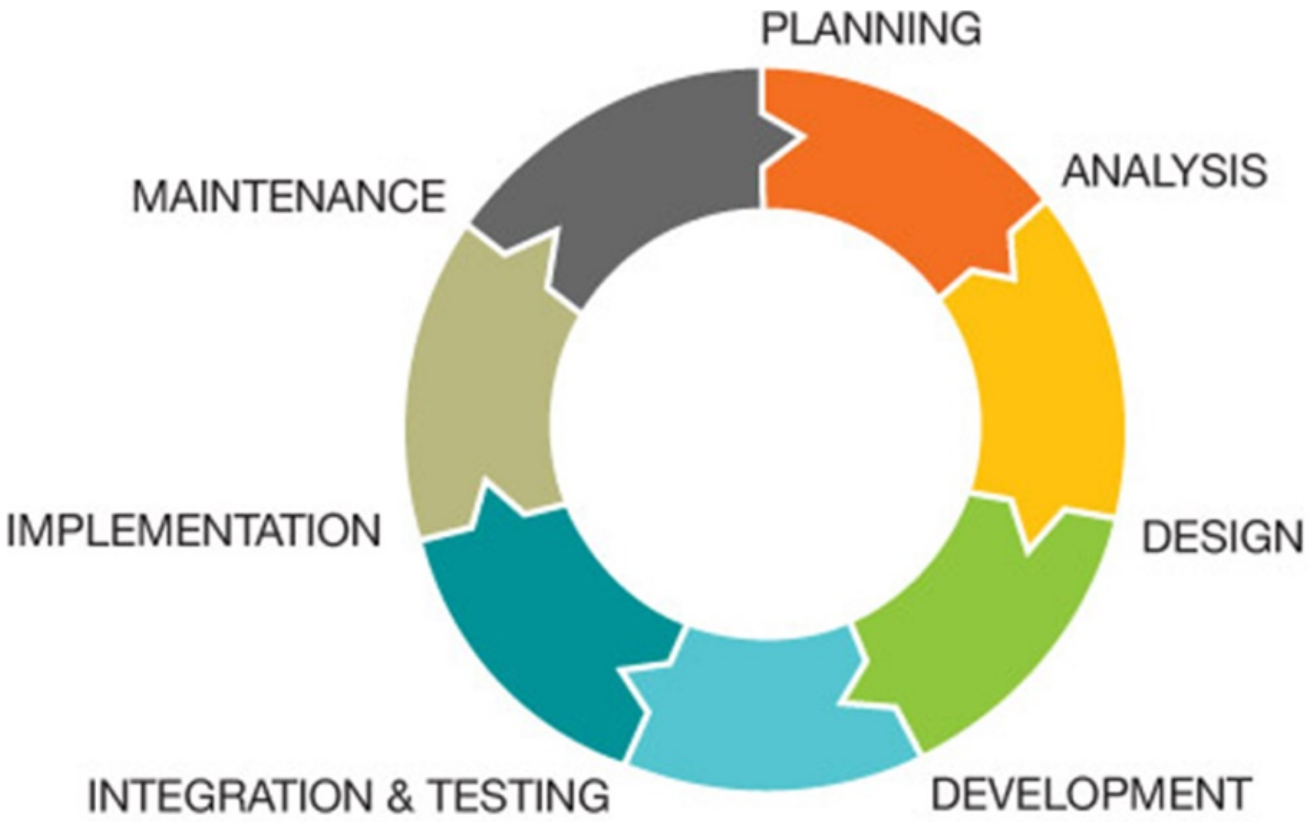
Занятие 2

План

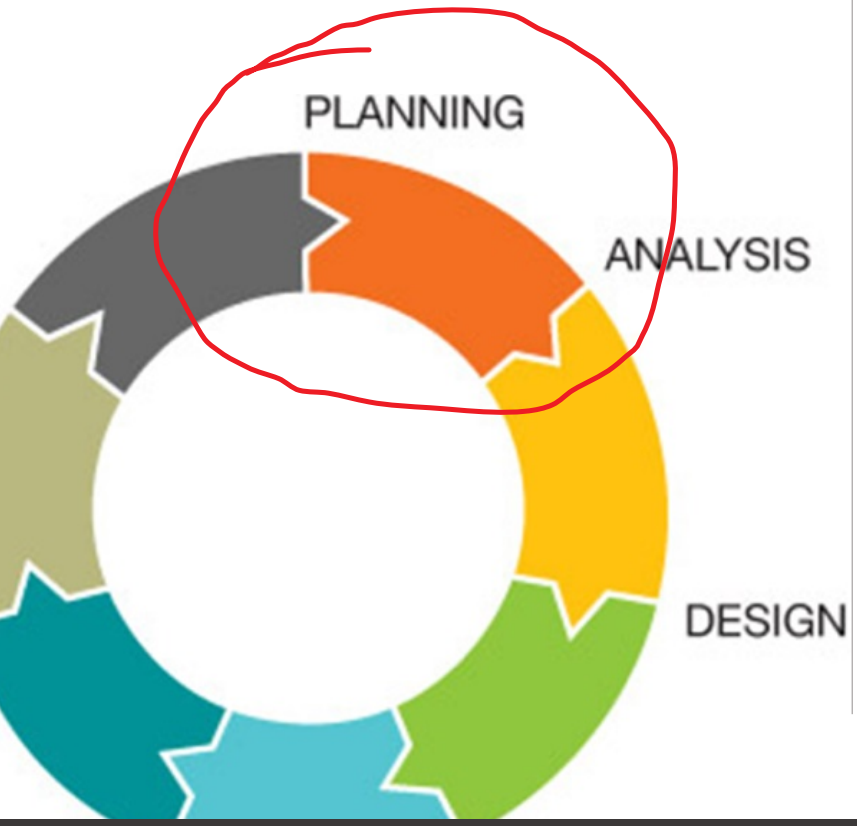
1. Жизненный цикл разработки системы (SDLC)
2. Методологии разработки
3. Роль БА в методологиях разработки
4. Попрактикуемся
5. Домой
6. Вопросы?

Жизненный цикл разработки системы (SDLC)

SDLC | Software development lifecycle



SDLC | Планирование



Планирование – наиболее критичный шаг в создании успешной системы. Во время этой фазы вы точно решаете, что хотите сделать и какие проблемы решить, при помощи:

- определения проблем, целей и ресурсов (таких, как персонал и издержки);
- изучения возможностей альтернативных решений путем встреч с клиентами, поставщиками, консультантами и сотрудниками;
- изучения, как сделать ваш продукт лучше, чем у конкурентов.

После анализа этих данных у вас будет три варианта: разработать новую систему, улучшить существующую или оставить систему как есть.

SDLC | Анализ

Анализ – Необходимо определить и задокументировать требования конечного пользователя системы - в чем его ожидания и как их осуществить. Кроме того, для проекта делается технико-экономическое обоснование, которое выясняет, является ли проект организационно, экономически, социально, технологически осуществимым. Очень важно поддерживать хороший уровень коммуникации с заказчиками, чтобы убедиться, что у вас есть ясное видение конечного продукта и его функций.

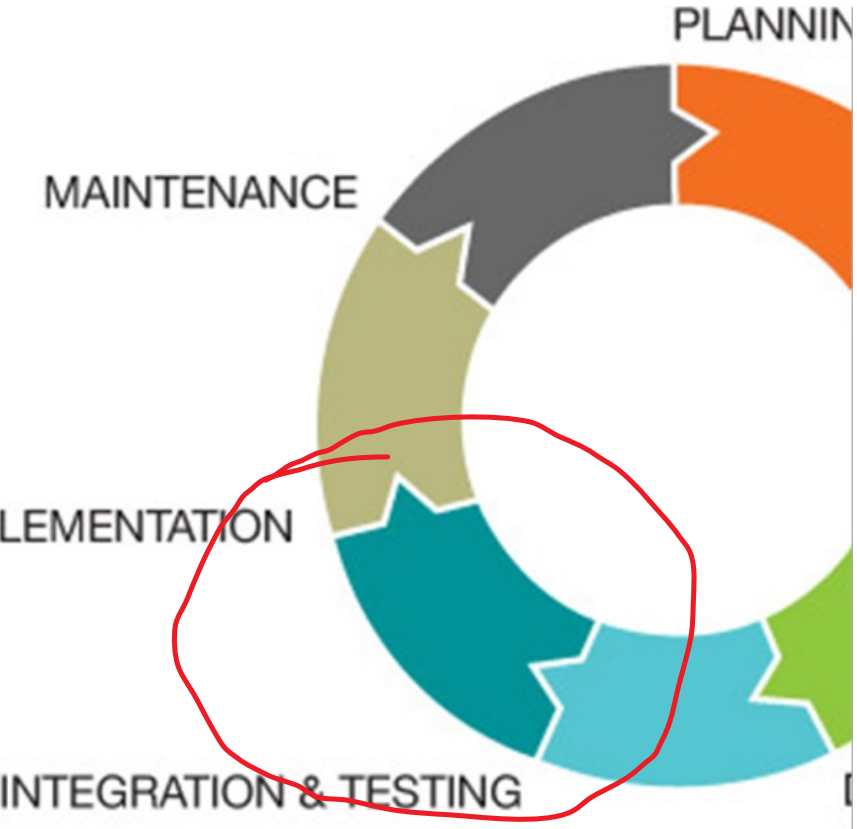
SDLC | Дизайн

Дизайн – Фаза дизайна наступает после того, как достигнуто хорошее понимание требований потребителя. Эта фаза определяет элементы системы, компоненты, уровень безопасности, модули, архитектуру, различные интерфейсы и типы данных, которыми оперирует система. Дизайн системы в общих чертах может быть сделан ручкой на листке бумаги - он определяет, как система будет выглядеть и как функционировать. Затем делается расширенный, детальный дизайн, с учетом всех функциональных и технических требований, как логически, так и физически.

SDLC | Разработка, внедрение и развертывание

Разработка, внедрение и развертывание – Эта фаза следует за полным пониманием системных требований и спецификаций. Это и есть собственно процесс разработки системы, когда дизайн системы уже полностью завершен и нагляден. В жизненном цикле разработки системы именно здесь пишется код, а если система включает хардверную часть, фаза внедрения будет включать в себя конфигурацию и настройку «железа» под определенные требования и функции. На этой стадии система готова к установке у заказчика, к запуску в боевом режиме. Возможно, конечным пользователям потребуется тренинг, чтобы они освоились с системой и знали, как ее использовать. Фаза внедрения может быть очень долгой - это зависит от сложности системы.

SDLC | Тестирование и интеграция

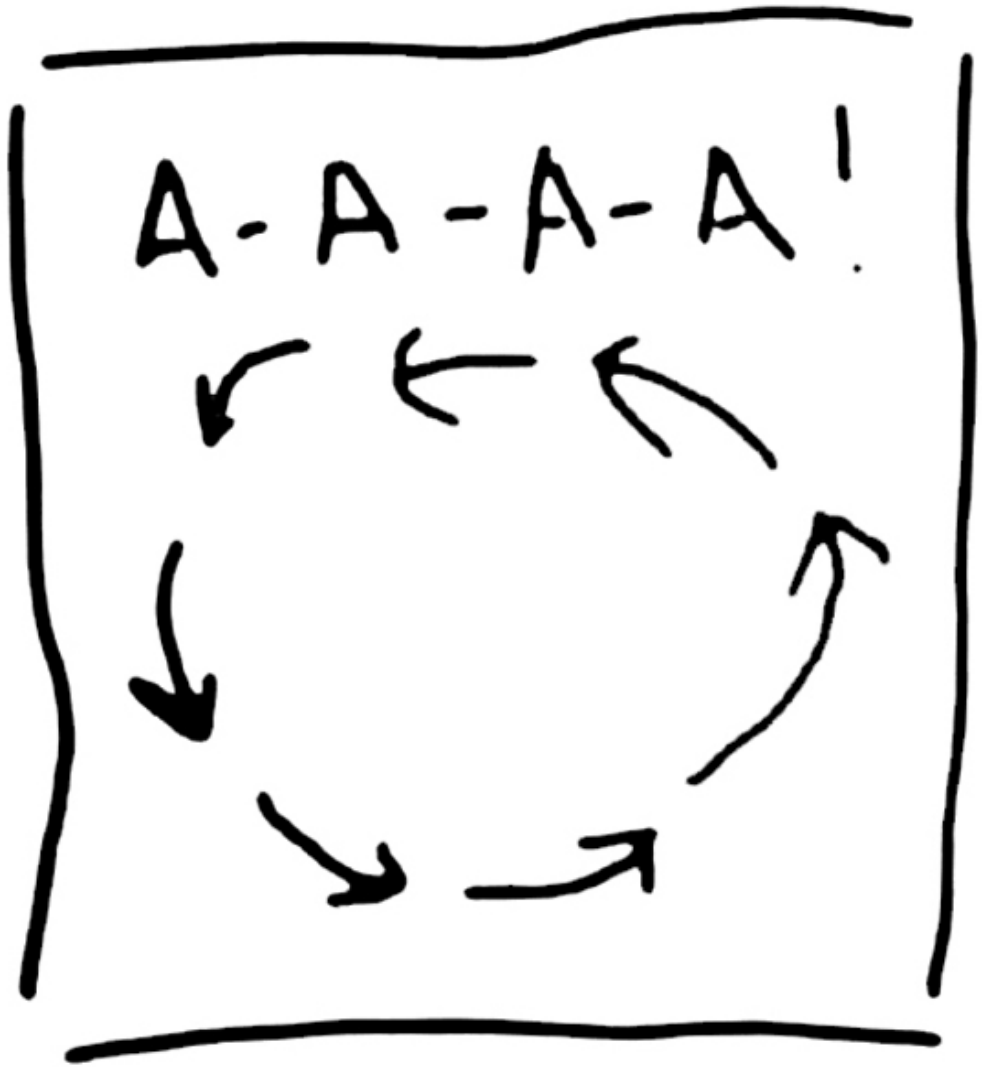


Тестирование и интеграция – Здесь происходит сборка различных компонентов и подсистем в одну целостную систему. Затем мы подаем системе различные входящие данные и анализируем выход, поведение и функционирование. Тестирование становится все важнее для удовлетворения потребителя, при этом оно не требует познаний ни в кодинге, ни в конфигурировании оборудования, ни в дизайне. Тестирование может выполняться настоящими пользователями или специальной командой сотрудников, также оно может быть систематическим и автоматизированным, с тем, чтобы удостовериться, что актуальные результаты работы системы совпадают с предусмотренными и желательными.

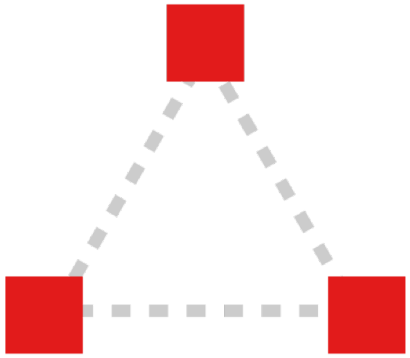
SDLC | Поддержка

Внедрение и поддержка – На этих фазах осуществляется выпуск ПО в промышленную эксплуатацию (в т.ч. и поэтапный) и дальнейшая периодическая техническая поддержка системы, чтобы убедиться, что система не устарела. Сюда входит замена старого оборудования и постоянная оценка производительности. Также здесь осуществляются апдейты определенных компонентов с целью удостовериться, что система отвечает нужным стандартам и новейшим технологиям.

Методологии разработки



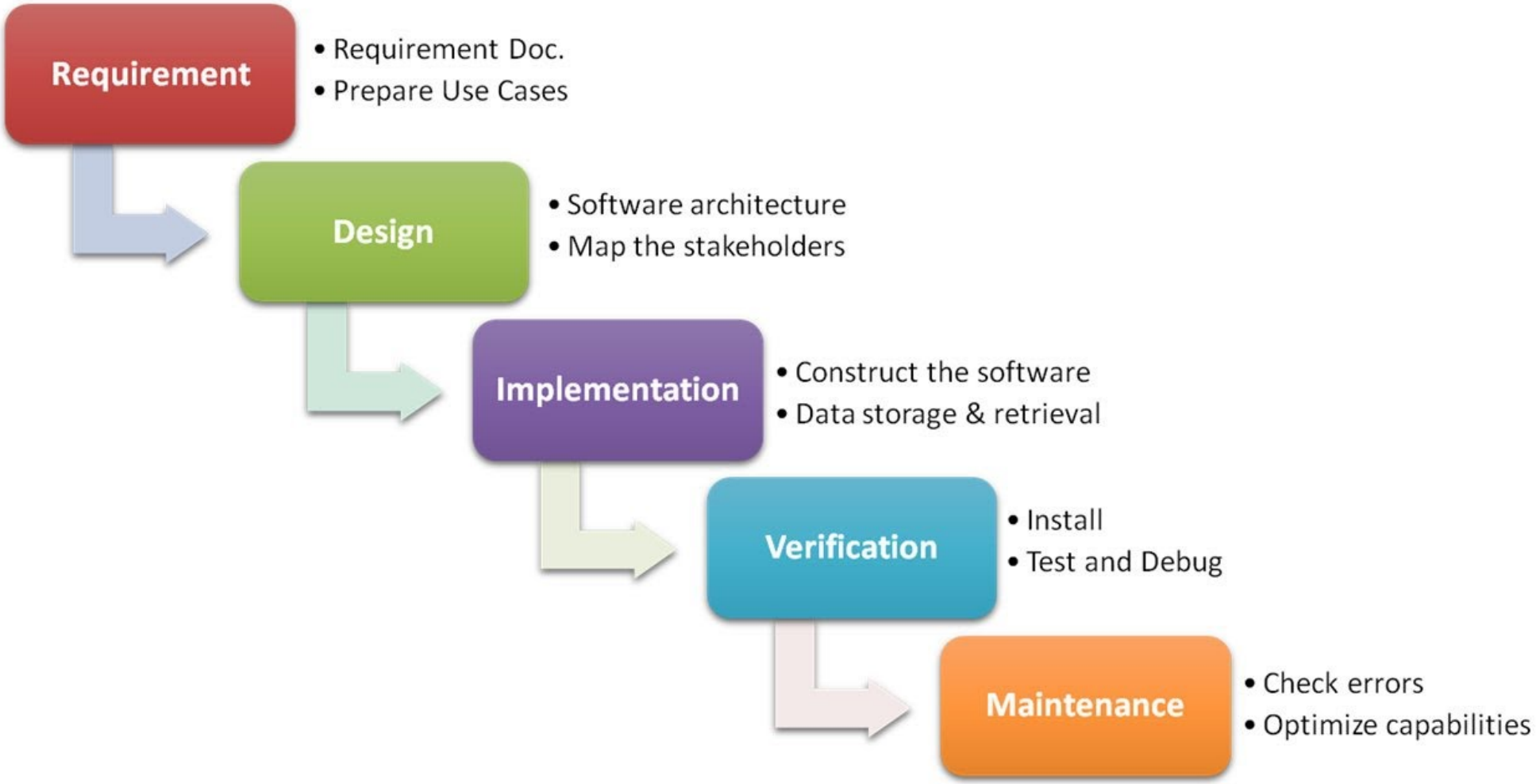
Методология – это система принципов, а также совокупность идей, понятий, методов, способов, средств и практик, определяющих стиль разработки программного обеспечения.



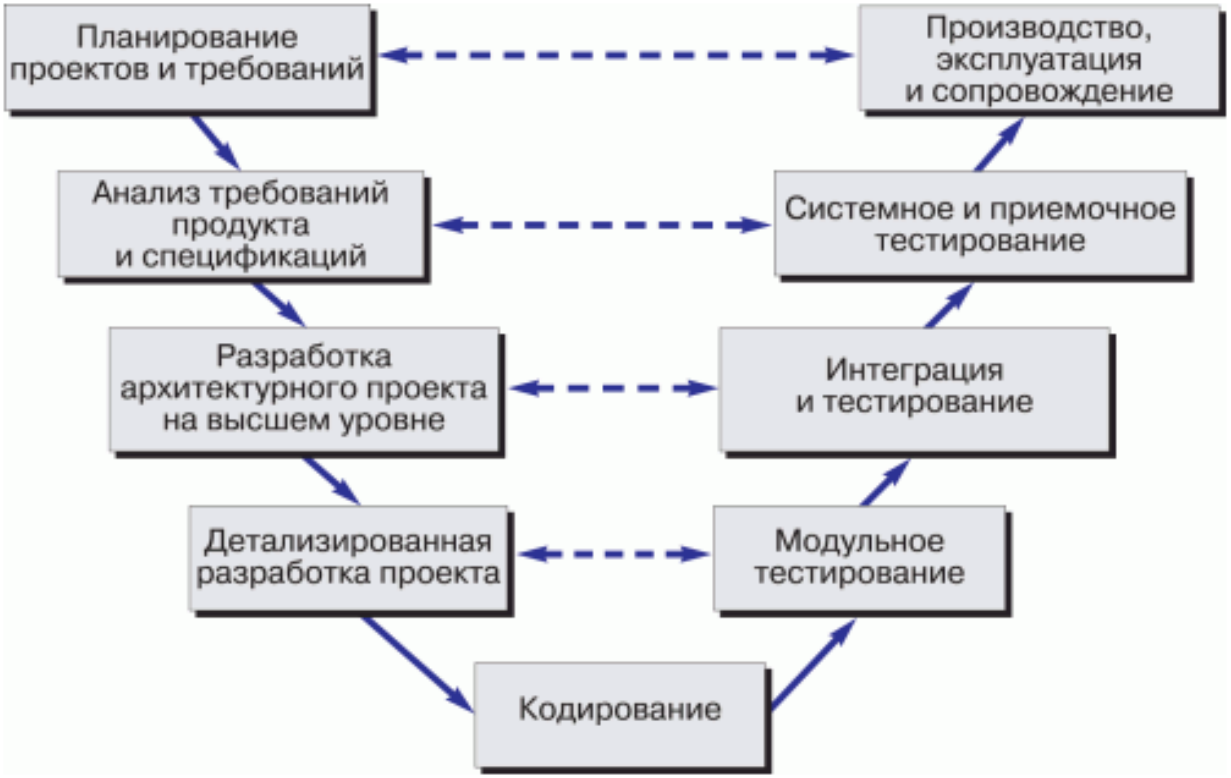
Методологии разработки | Линейные методологии

Предполагают выполнение всех фаз (этапов) разработки системы в строго последовательной очередности. Переход на каждый следующий этап осуществляется только по полному завершению работ по текущему этапу и наличию документации.

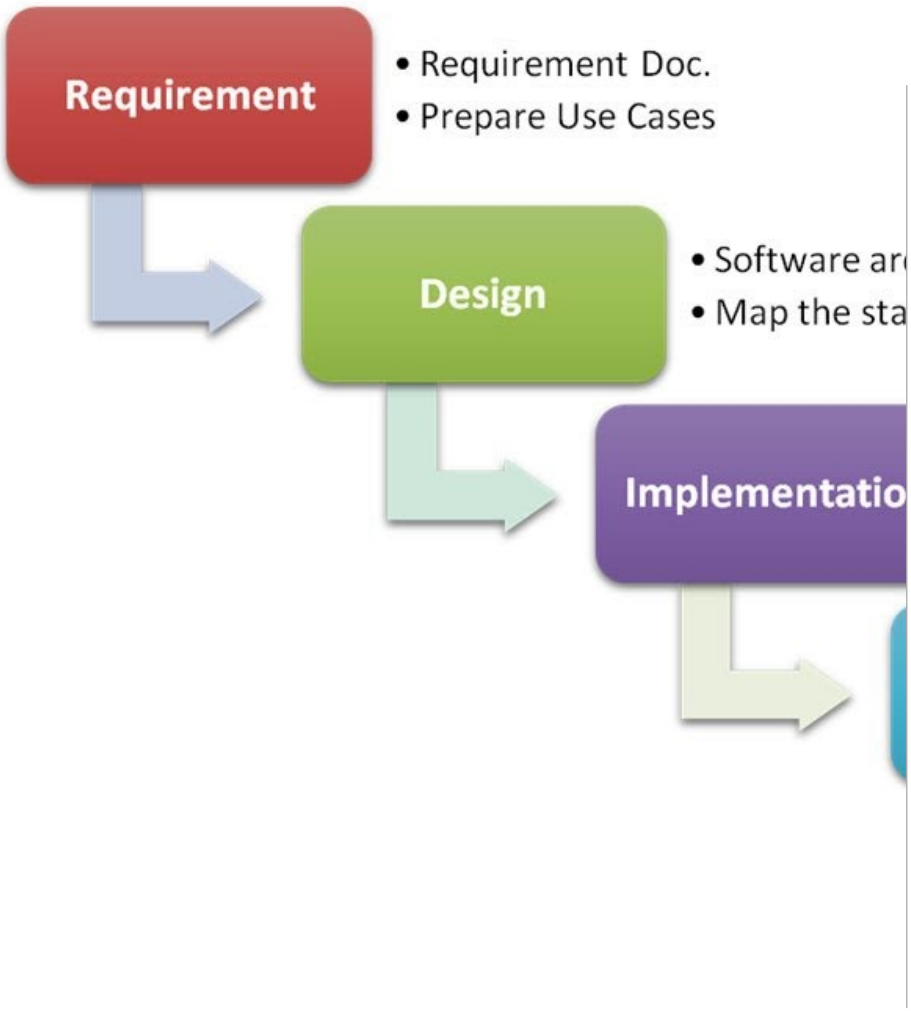
Методологии разработки | Waterfall



Методологии разработки | V-модель



Методологии разработки | + / - линейных методологий



Преимущества:

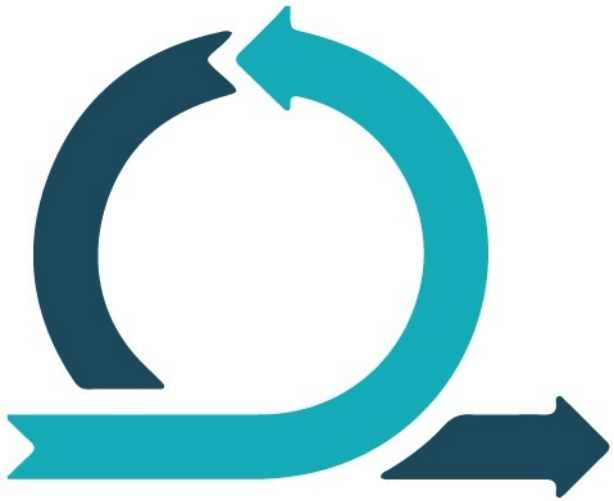
+ Полная и согласованная документация системы;
+ Легко определить сроки и ресурсы;
+ Высокая контролируемость процессов.

Недостатки:

- Низкая устойчивость к изменениям;
- Необходимость полного понимания требований на начальных этапах.

Попрактикуемся!

Методологии разработки | Гибкие методологии (Agile)



В гибких методологиях разработка программной системы сводится к серии коротких итераций (1 – 3 недели). Каждая итерация, по сути, является небольшим проектом, где каждая задача проходит большинство стадий жизненного цикла. Главная цель каждой итерации – получить некий инкремент, представляющий ценность для конечного пользователя. Однако, этого инкремента не всегда достаточно для выпуска релизной версии продукта. Все принципы гибких методологий сведены в Agile Manifesto.

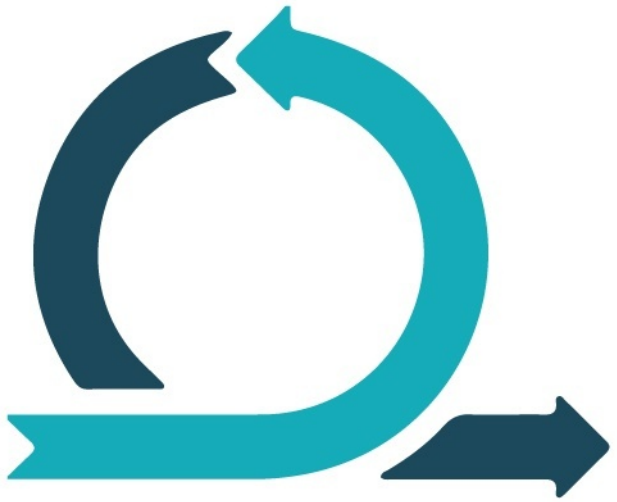
Методологии разработки | Agile Manifesto

1. Люди и взаимодействие важнее процессов;
2. Готовый продукт важнее документации по нему;
3. Сотрудничество с заказчиком важнее согласования условий контракта;
4. Готовность к изменениям важнее следования первоначальному плану.

Методологии разработки | Гибкие методологии (Agile)

- Удовлетворение клиента за счёт ранней и бесперебойной поставки ценного программного обеспечения;
- Приветствие изменений требований даже в конце разработки;
- Частая поставка рабочего программного обеспечения;
- Тесное, ежедневное общение заказчика с разработчиками на протяжении всего проекта;
- Проектом занимаются мотивированные личности, которые обеспечены нужными условиями работы, поддержкой и доверием;
- Рекомендуемый метод передачи информации — личный разговор;
- Работающее программное обеспечение — лучший измеритель прогресса;
- Спонсоры, разработчики и пользователи должны иметь возможность поддерживать постоянный темп на неопределённый срок;
- Постоянное внимание улучшению технического мастерства и удобному дизайну;
- Простота — искусство не делать лишней работы;
- Лучшие технические требования, дизайн и архитектура получаются у самоорганизованной команды;
- Постоянная адаптация к изменяющимся обстоятельствам. Команда должна систематически анализировать возможные способы улучшения эффективности и соответственно корректировать стиль своей работы.

Методологии разработки | + / - гибких методологий



Преимущества:

+ Высокая скорость разработки => быстрее продукт попадает пользователям;
+ Высокая устойчивость к изменениям;
+ Вовлеченность всех заинтересованных лиц в процесс;
+ Низкие затраты времени на процессные обязательства.

Недостатки:

- Скудность документации;
- Контролируемость процессов и риски.

Методологии разработки | В каких проектах применимы?

Линейные

- Необходимость подробной и детальной документации;
- На начальных этапах известно, что хотим получить в итоге.
- Необходимость точного прогнозирования.

Примеры проектов: инженерные проекты, ре-платформинг, совершенствование текущей системы.

Гибкие

- Ограниченные сроки и ресурсы;
- Нет полного понимания итогового продукта;
- Высокая вероятность изменений.

Примеры проектов: разр. новой ф-ти, совершенствование существующей системы и др.

Методологии разработки |SCRUM-объекты

Роли

- Product Owner;
- SCRUM-master;
- Team.

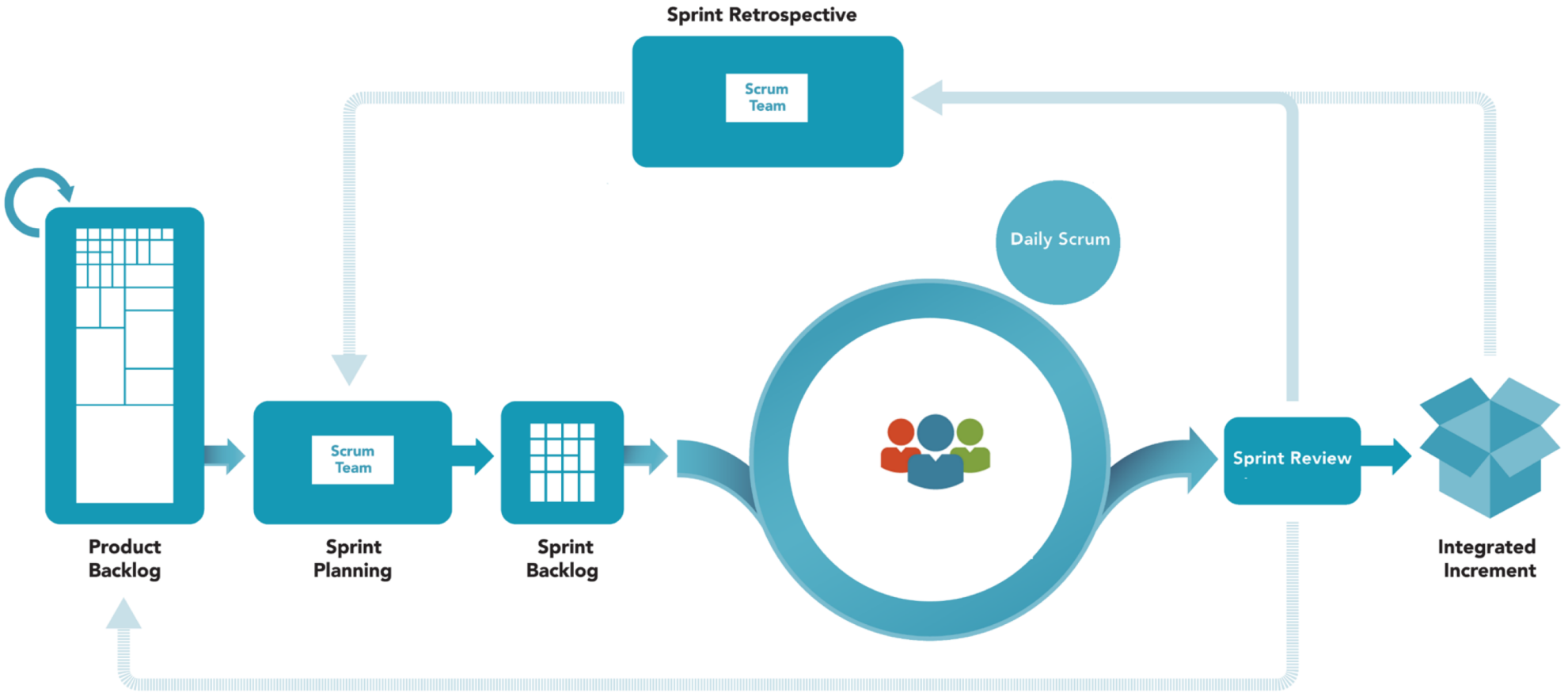
Артефакты

- Product backlog;
- Sprint backlog;
- Burndown chart.
- Increment.

Активности

- Planning;
- Daily stand-up;
- Retrospective;
- Sprint demo.

Методологии разработки |SCRUM-процесс

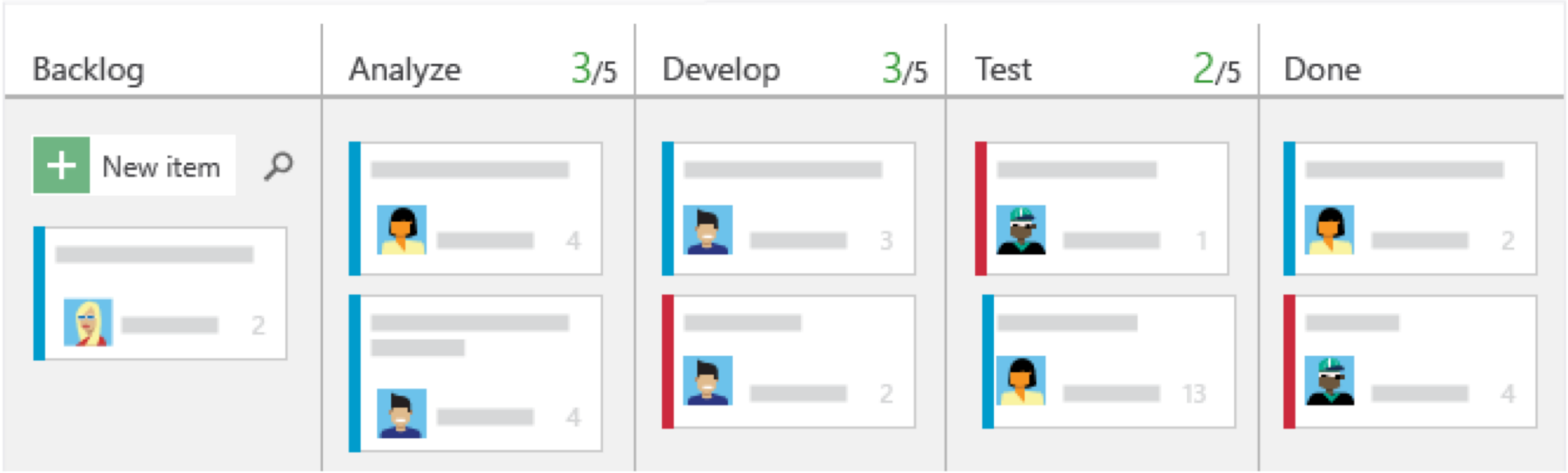


Попрактикуемся!

Методологии разработки | Kanban

Канбан – гибкий процесс разработки, предполагающий равномерное распределение нагрузки между всеми участниками. Каждый понимает, что и когда должно выполняться в определенный момент времени. В отличие от SCRUM, не обязывает участников брать ответственность закрыть задачу до конца спринта, проводить все процессные активности.

Методологии разработки | Kanban



В реальных проектах редко можно встретить организацию процессов разработки, строго соответствующих предписанным правилам и принципам. Все напрямую зависит от внутренних процессов компании, обстоятельств.

Роль БА, в методологиях разработки

Роль БА в мет-ях| БА в Waterfall

- Этап анализа - наиболее активная роль: сбор и анализ требований, создание всеобъемлющей документации и согласование с заказчиком, технической стороной.
- Этап дизайна - сотрудничество с архитекторами и разработчиками по вопросам соответствия требованиям заказчика предлагаемые ими решения.
- Этап имплементации – сотрудничество с командой разработки, разъяснение, уточнение требований. Приемка. Демо промежуточных результатов.
- Этап верификации и внедрения – проверка соответствия готовой ф-ти, консультации тестировщикам по части пунктов требований. Помощь в разработке тестовых сценариев. Сопровождение при приемке продукта заказчиков. Отработка возражений и запросов на изменение.
- Этап поддержки и сопровождения – обучение конечных пользователей функциональным особенностям, написание руководств по пользованию. Помощь в определении причин ошибок и их соответствия контракту.

Роль БА в мет-ях| БА в Waterfall

- Этап анализа - наиболее активная роль: сбор и анализ требований, создание всеобъемлющей документации и согласование с заказчиком, технической стороной.
- Этап дизайна - сотрудничество с архитекторами и разработчиками по вопросам соответствия требованиям заказчика предлагаемые ими решения.
- Этап имплементации – сотрудничество с командой разработки, разъяснение, уточнение требований. Приемка. Демо промежуточных результатов.
- Этап верификации и внедрения – проверка соответствия готовой ф-ти, консультации тестировщикам по части пунктов требований. Помощь в разработке тестовых сценариев. Сопровождение при приемке продукта заказчиков. Отработка возражений и запросов на изменение.
- Этап поддержки и сопровождения – обучение конечных пользователей функциональным особенностям, написание руководств по пользованию. Помощь в определении причин ошибок и их соответствия контракту.

Роль БА в мет-ях| БА в SCRUM

Product Owner

BA

Team

SCRUM-master

Роль БА в мет-ях| БА в SCRUM

- PBL – Детализирует задачи, описывает юзер-стори, критерии приемки. Помогает PO определить приоритеты задач, в т.ч. И с технической стороны.
- Planning, Grooming – Консультирует команду по границам задач, сути. Вносит уточнения в требования. Помогает оценить задачи.
- Sprint – Консультирует команду по задачам. Выполняет приемку функциональности, вносит уточнения.
- Daily stand-up – Может привлекаться в качестве консультанта, однако своими планами обычно не делится ☺
- Demo – Может демонстрировать результаты спринта (чаще всего), отрабатывать возражения

Домой

- Читать: BABOK Chapter 1 до конца ☺
- К. Вигерс «Разработка требований к ПО» глава 1, 2, 4.
- На досуг: Scrum Guide.