Project 01

SDLC and STLC. Test Artifacts.

WELCOME

ТГУ & Школа 21

Привет!

Меня зовут Карина! ♥

Давайте узнаем, какую жизнь проживет наш будущий проект!

Удобная страховка

Застраховаться можно на любую сумму от 30 000 до 1 000 000 рублей

Размер платежей выбираете сами

Остановить страхование можно в любой момент

Структура нашей беседы:

01 SDLC

02 Team members

03 STLC

04 О тестировании

05 QA-специалист & тестировщик

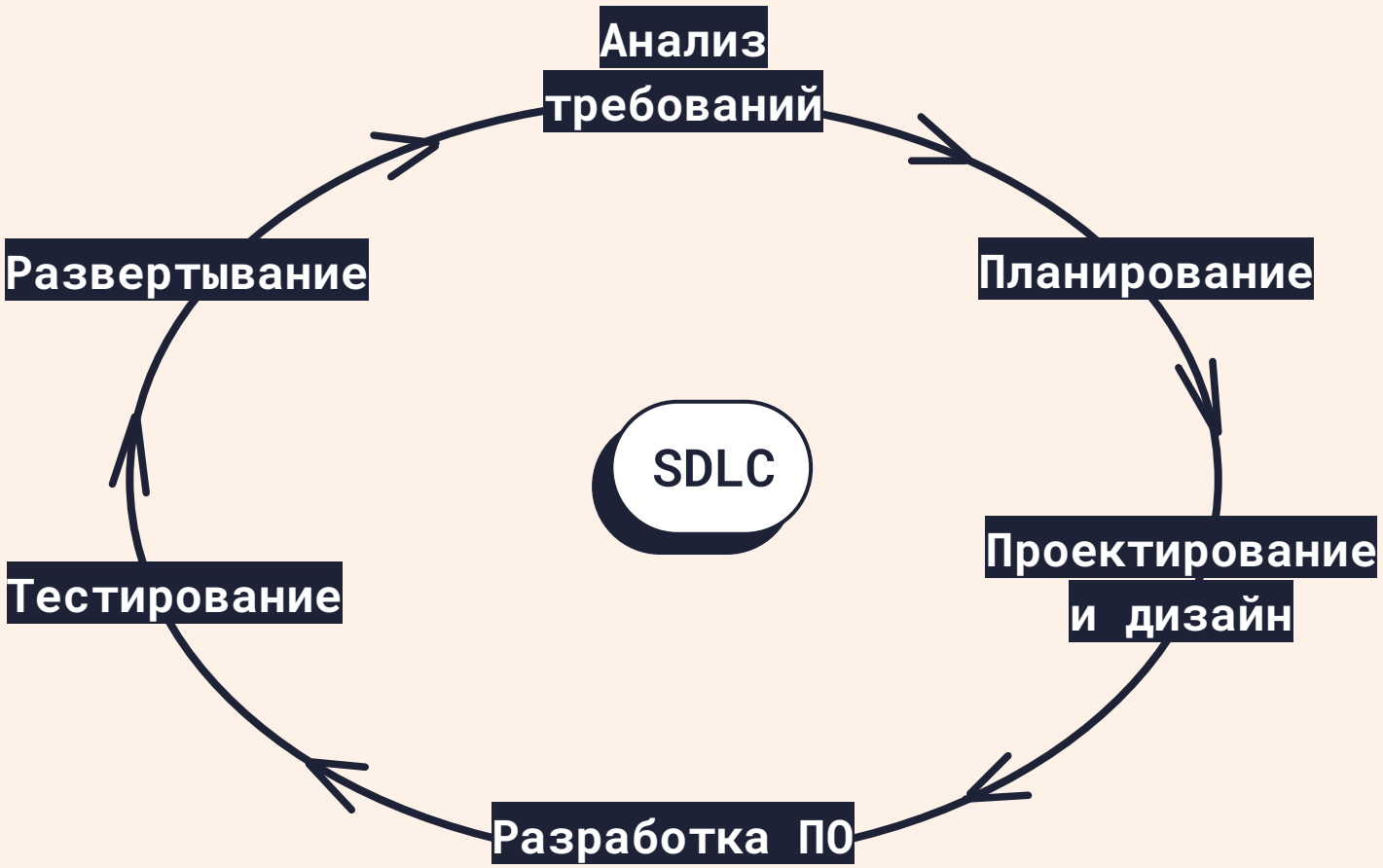
06 Методологии разработки ПО

07 Церемонии, появляющиеся в команде, работающей по Agile

01

Начнем с SDLC.

SDLC *(Software Development Life Cycle)* - жизненный цикл разработки будущего приложения.



В целом, SDLC можно представить в виде **замкнутого цикла**, в котором каждый этап влияет на действия в последующих и дает перспективные указания на будущее.

Цикл проходит необходимое количество итераций, пока не будут удовлетворены все требования.

Переход между этапами не всегда выполняется строго последовательно. За особенности движения по этапам SDLC отвечают модели разработки ПО, которые более детально мы рассмотрим в **06** блоке.

Подробнее об этапах

Анализ требований

Отвечает на вопрос **«Какие проблемы требуют решений?»**.

Планирование

Отвечает на вопрос **«Что мы хотим сделать?»**.

Проектирование и дизайн

Отвечает на вопрос **«Как мы добьемся наших целей?»**

Разработка ПО

Регулирует **процесс создания продукта**.

Тестирование

Проверка программы на **соответствие** всем предъявляемым к ней **требованиям**.

Развертывание

Регулирует **использование финального продукта**.

SDLC.AGILE

В данном проекте выбрана **гибкая методология разработки ПО (AGILE)**.

Процесс работы *«по эджайлу»* делится на итерации – короткие циклы по две-три недели. Каждый цикл решает серию задач. По итогам каждой итерации команда анализирует результаты и меняет приоритеты для следующего цикла. В итоге за каждый цикл создается мини-продукт или отдельная часть, готовая к самостоятельному запуску.

02

Поговорим о Team members.

*о составе команды.

Наша команда

01 Владелец продукта

Представляет бизнес и говорит, чего требуется достичь

02 Проджект-менеджер

Отвечает за все процессы ведения и успешного завершения проекта

03 Продакт-менеджер

Отвечает за старт, ведение и сдачу проектных работ

04 Back-end разработчик

Разрабатывает серверную часть приложения и базу данных

05 Front-end разработчик

Разрабатывает клиентскую часть веб-приложения

06 Тестировщик

Тестирует продукт, чтобы убедиться, что он хорошо работает

Наша команда

07 Бизнес-аналитик

Оценивает, что работает, а что нет, и задает направление развития бизнеса

08 Дизайнер UX

Разрабатывает функциональную составляющую интерфейсов

09 Дизайнер UI

Разрабатывает графическую составляющую интерфейсов

03

А теперь время STLC.

STLC *(Software Testing Life Cycle)* - жизненный цикл тестирования будущего приложения.

STLC является частью SDLC. Это цикл в цикле. Он так же проходит необходимое количество итераций.

STLC запускается, как только требования определены или SRD (Документ с требованиями к программному обеспечению) передается группе тестирования.

В итоге дефекты классифицируются с точки зрения команды QA как **Приоритет**(порядок, в котором дефекты должны быть устранены), а с точки зрения разработки – как **Серьезность**(сложность кода для его устранения).

Подробнее об этапах

Анализ требований

Группа тестирования проводит **анализ высокого уровня**, касающийся тестируемого проекта.

Планирование тестирования

Команда тестирования **планирует стратегию и подход**.

Проектирование тест-кейсов

Разработка тест-кейсов на основе объема и критериев.

Настройка тестовой среды

Когда интегрированная среда **готова для проверки** продукта.

Выполнение теста

Проверка продукта в режиме реального времени и поиск ошибок.

Закрытие теста

После завершения тестирования матрица, отчеты, **результаты** документируются.

04

Что же такое тестирование?

каково его значение в процессе создания программного обеспечения?

Тестирование

- *процесс оценки качества* разрабатываемого ПО путем проверки соответствия программного продукта заявленным требованиям.

Тестирование играет важнейшую роль в обеспечении качества продукта.

05

А кто проводит тестирование?

кто такой QA-специалист, значение роли тестировщика, обязанности и основные задачи, которые он выполняет на проекте.

QA, QC и тестирование

QA-специалист

контролирует и обеспечивает качество работы продукта компании. Он отвечает и за отдельные этапы разработки софта. В частности, за выбор инструментов для разработки, предотвращение возможных проблем. Еще он участвует в процессе совершенствования продукта. QA охватывает все этапы разработки, включая описание проекта, собственно, тестирование, релиз и, зачастую, пост-релизный этап.

QC-специалист

отвечает за проверку конкретного продукта, что включает анализ кода продукта, дизайна, плюс тестирование. QC-инженер разрабатывает стратегию тестирование вполне определенного тестирования, взаимодействует с разработчиками и организует само тестирование.

Специалист по тестированию

занимается выполнением тестов. Тестированием называют проверку соответствия результатов работы программного продукта на соответствие заданным критериям. Тестировщики занимаются тестированием всего продукта в целом или же отдельных компонентов. Тестирование играет важнейшую роль в обеспечении качества продукта.

А теперь про подход к гибкой разработке и об agile-ролях!

06

Методологии разработки ПО

их отличия, плюсы и минусы.

Основные методологии разработки ПО

Традиционные

- «Waterfall Model
- «V-Model»
- «Incremental Model»
- «Iterative Model»
- «Spiral Model»

Гибкие

- Scrum
- Kanban

Рассмотрим традиционные модели разработки ПО

«Waterfall Model» (каскадная модель или «водопад»)

Подразумевает последовательное прохождение стадий, каждая из которых должна завершиться полностью до начала следующей. Благодаря её жесткости, разработка проходит быстро, стоимость и срок заранее определены. Но нет возможности сделать шаг назад, тестирование начинается только после того, как разработка завершена.

Плюсы и минусы:

+ Разработку легко контролировать
+ Стоимость проекта определяется заранее
- Тестирование начинается на последних этапах разработки
- Заказчик видит готовый продукт только в конце
- Разработчики пишут большой объем технической документации

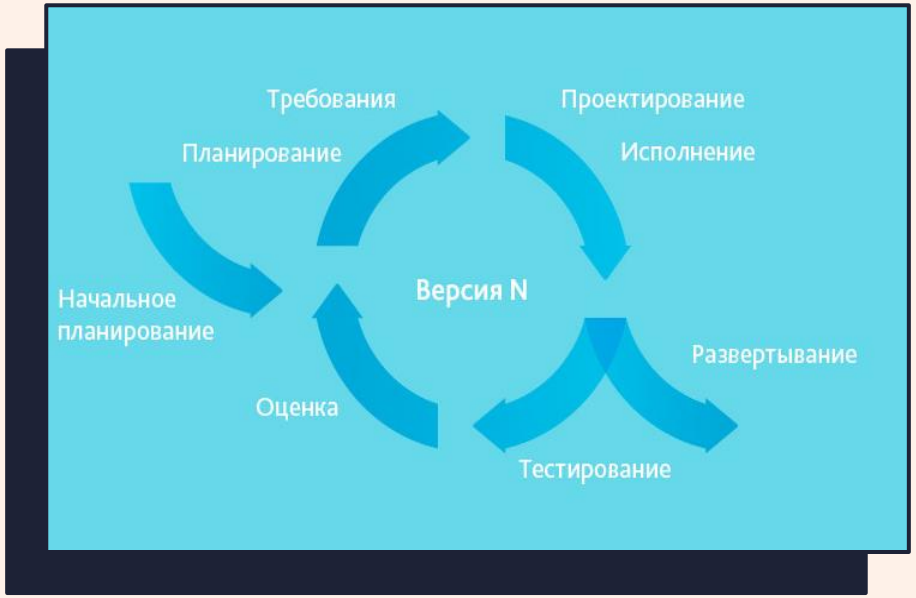
«V-Model»

Унаследовала структуру «шаг за шагом» от каскадной модели. Особенностью модели можно считать то, что она направлена на тщательную проверку и тестирование продукта, находящегося уже на первоначальных стадиях проектирования. Стадия тестирования проводится одновременно с соответствующей стадией разработки.

Плюсы и минусы:

+ Количество ошибок в архитектуре ПО сводится к минимуму
- Если при разработке архитектуры была допущена ошибка, то вернуться и исправить ее будет очень дорого

«Incremental Model» (инкрементная модель)

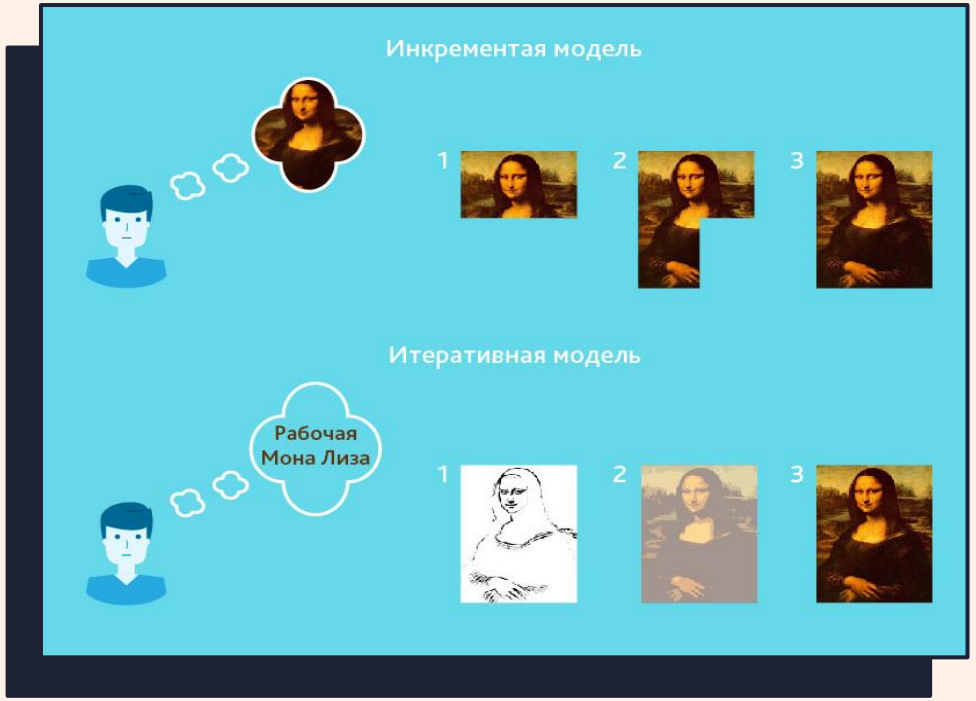


Инкрементные модели используются там, где отдельные запросы на изменение ясны, могут быть легко формализованы и реализованы.

Плюсы и минусы:

+ Рабочее приложение выходит на ранней стадии жизненного цикла продукта
+ Гибкость
+ Проще идентифицировать риски, справиться с ними
- Каждая фаза итерации неподвижна
- Могут возникнуть проблемы относительно архитектуры системы, так как не все требования собраны заранее для всего жизненного цикла ПО

«Iterative Model» (итеративная или итерационная модель)



Не требует для начала полной спецификации требований. Вместо этого, создание начинается с реализации части функционала, становящейся базой для определения дальнейших требований. Этот процесс повторяется. Версия может быть неидеальна, главное, чтобы она работала.

Плюсы и минусы:

+ Быстрый выпуск минимального продукта дает возможность быстро получать обратную связь как от заказчика, так и от пользователей
+ Постоянное тестирование пользователями позволяет быстро обнаружить и исправить ошибки
- Отсутствие фиксированного бюджета и сроков

«Spiral Model» (спиральная модель)

Спиральная модель предполагает 4 этапа для каждого витка:

1. планирование;
2. анализ рисков;
3. конструирование;
4. оценка результата и при удовлетворительном качестве переход к новому витку.

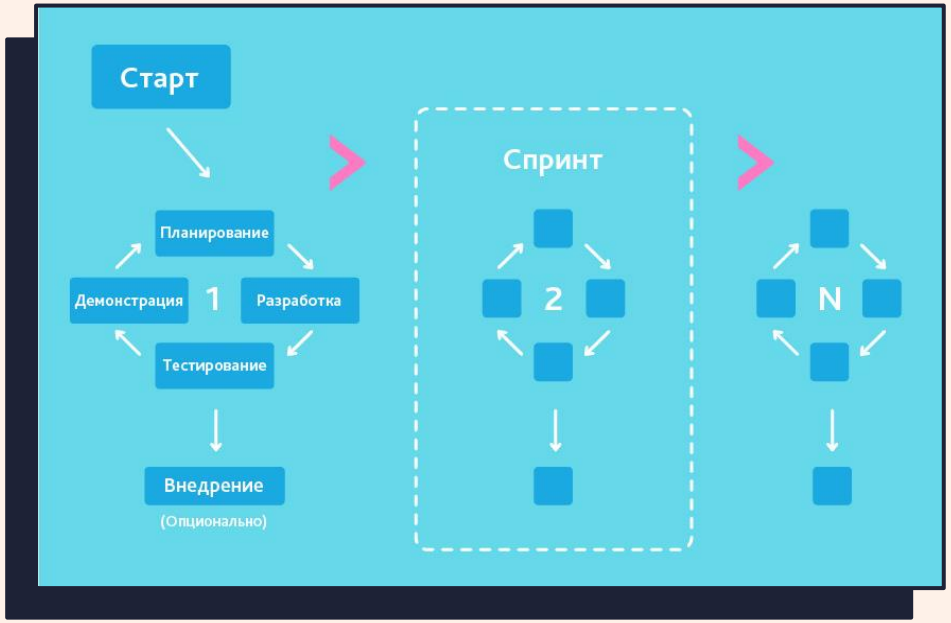
** Подходит только для сложных и дорогих проектов.*

Плюсы и минусы:

\+ Очень большое внимание уделяется проработке рисков
\- Есть риск застрять на начальном этапе
\- Разработка длится долго и стоит очень дорого

Рассмотрим гибкие модели разработки ПО

Scrum

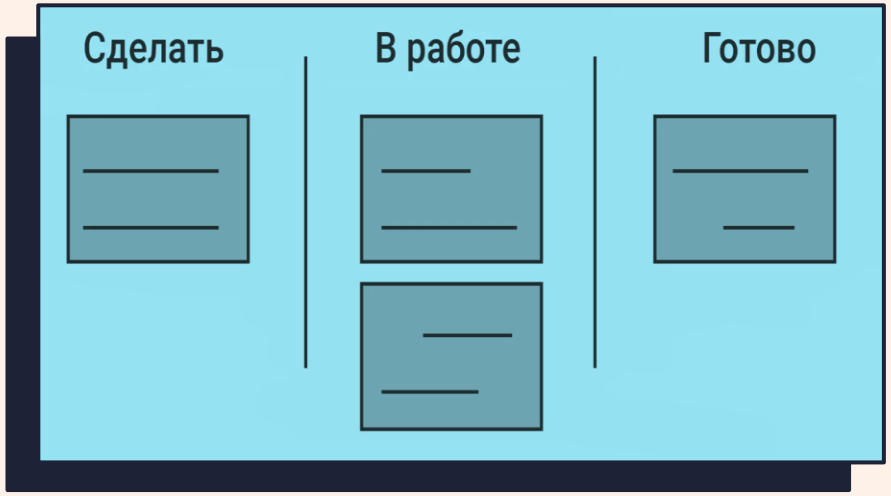


SCRUM – это фреймворк для управления проектами. Вся разработка делится на спринты. Есть backlog (список задач) для всего периода разработки и для каждого спринта отдельно. Каждая задача имеет свой story point (оценку сложности). После каждой итерации заказчик может наблюдать результат и понимать, удовлетворяет он его или нет.

Плюсы и минусы:

+ Заказчик может наблюдать результат в процессе разработки
+ Ежедневный контроль над процессом разработки
+ Возможность вносить коррективы во время разработки
+ Налаженные коммуникации со всеми членами команды
+ Малое количество документации
- Сложность оценки трудозатрат и стоимости требуемой на разработку
- Необходимость вовлечения каждого члена команды

Kanban



Команда ведёт работу с помощью виртуальной доски, которая разбита на этапы проекта. Каждый участник видит, какие задачи находятся в работе, какие – застряли на одном из этапов, а какие уже дошли до его столбца и требуют внимания.

Плюсы и минусы:

+ Простота использования
+ Наглядность
+ Высокая вовлеченность всей команды в процесс
+ Высокая гибкость в разработке
- Нестабильный список задач
- Сложность применения на долгосрочных проектах
- Отсутствие жестких дедлайнов

07

Какие церемонии появляются в команде, работающей по Agile?

их краткое описание и назначение.

Agile-церемонии

Stand-up

- короткий ежедневный митинг со всеми членами команды

Каждый отвечает на 3 вопроса:

1. Что делал?
2. Что будет делать?
3. И какие есть блокеры?

Sprint Planning

- событие, которое знаменует начало спринта

В ходе планирование определяется **объем** работы на спринт и **способы** выполнения этой работы.

Груминг

- собрание представителей Scrum-команды.

Во время собрания обсуждаются детали бэклога продукта и готовится очередное планирование спринта.

Agile-церемонии

Ревью

* проводится в конце спринта.

Команда рассказывает, что было сделано, и демонстрирует те части проекта, которые окончательно готовы..

Ретроспектива

* проводится в конце спринта.

Суть – выяснить, что было сделано хорошо, а что можно было улучшить. Помогают определить, **что работает, а что нет**.

Демо

- показ результатов, достигнутых командой, и их оценка стейкхолдерами.

Демо проводится в виде встречи, на которой присутствует команда и все заинтересованные стейкхолдеры.

Как правило, в agile-командах все участники проекта равноценны в иерархии и работают в одном пространстве.

☺

Звенья AGILE-команды

01 Product Owner

Описывает бэклог(перечень требований) продукта и управляет им

02 Команда разработки

Отвечает за разработку ПО. Должна быть кросс-функциональнальной

03 Scrum Master

Отвечает за грамотное применение той или иной гибкой методологии

04 Stakeholder

Влияет на принятие решений и работу команды разработчиков, но не напрямую

Ключевой ценностью Agile является «качество - это ответственность команды», в которой подчеркивается, что за качество программного обеспечения отвечает вся команда

Спасибо за внимание!

У Вас есть вопросы?

karina825karpenkova@yandex.ru

THANK YOU