



СПбГЭТУ «ЛЭТИ»
ПЕРВЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ

Е. Л. Турнецкая

Agile. Индустриальные стандарты разработки ПО

Конспект лекций

СПбГЭТУ «ЛЭТИ», 2023 г.

ПРАКТИКИ ИНДУСТРИАЛЬНОЙ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Методология разработки ПО кроме модели включает набор методов по управлению организацией самого процесса разработки [1]. Таким образом, методология конкретизирует процесс разработки ПО в рамках выбранной модели.

Принято различать «тяжеловесные» и «лёгкие» методологии разработки ПО. К первой группе относят методологии, построенные, например, на водопадной модели, где все этапы проекта определены. Такие методологии применяют при работе над большими или средними проектами. При организации разработки проектов с быстро меняющимися требованиями эффективнее применять набор практик, которые получили название «динамические», «облегченные» или «agile»-методологии

1. Принципы Agile

Основополагающий документ Agile Manifesto содержит четыре основные идеи и двенадцать принципов (рис.1).



Рисунок 1 – Основные принципы Agile

Основные идеи Agile Manifesto [2]:

1. Люди и взаимодействие важнее процессов и инструментов.
2. Работающий продукт важнее исчерпывающей документации.
3. Сотрудничество с заказчиком важнее согласования условий контракта.
4. Готовность к изменениям важнее следования первоначальному плану.

Выделим основные принципы Agile Manifesto:

А. Работающий продукт следует выпускать как можно чаще, с периодичностью от пары недель до пары месяцев.

Б. Изменения в требованиях вносят до завершающего этапа разработки.

В. Постоянное внимание к техническому совершенству и качеству проектирования повышает гибкость проекта.

Г. При разработке придерживаются принципа минимализма. Выбирают самое простое решение для реализации.

Д. Постоянный анализ выполненных задач и обсуждения по улучшению процессов разработки внутри команды.

При сравнении подходов к разработке используют критерий получения результата для заказчика и команды разработки.

При организации разработки на основе каскадной модели подводят итоги после внедрения программного продукта (рис.2). Если выбраны гибкие практики, то результат можно оценить после каждой итерации (рис.2).

ВОДОПАДНАЯ МОДЕЛЬ

Большая выгода в конце проекта



AGILE-ПРАКТИКИ



Рисунок 2 – Сравнение водопадной модели и agile-практик

2. Фреймворк SCRUM

Рейтинг agile-практик, применяемых при управлении разработкой программного продукта, показывает, что лидером выступает гибкий фреймворк SCRUM (рис.3) [6].

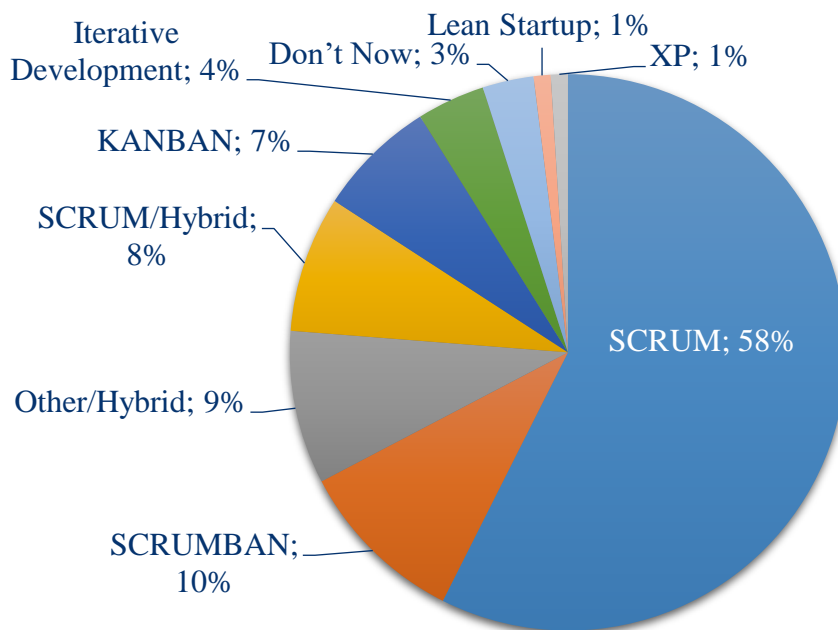


Рисунок 3 – Рейтинг agile-практик

При организации процесса разработки по SCRUM выделяют роли участников, артефакты и процессы (табл.1).

Таблица 1 – Роли, артефакты и процессы в SCRUM

Роли	Артефакты	Процессы
Команда Scrum-мастер Владелец продукта Стейкхолдеры	Журнал (бэклог) продукта Журнал (бэклог) спринта Инкремент продукта	Спринт Планирование спринта Scrum-митинг Обзор спринта Ретроспектива

В первую очередь, необходима команда универсальных разработчиков (Delivery Team), которые могут взаимозаменять друг друга по профессиональным компетенциям на любом этапе разработки [1, 3, 6].

Как правило, в состав команды входит от семи до десяти высокопрофессиональных специалистов, которые взаимодействуют со SCRUM-мастером и владельцем продукта (Product Owner).

SCRUM-мастер организует и обеспечивает бизнес-процессы, занимается решением бытовых проблем, проводит общие собрания (митинги), мотивирует сотрудников и следит за тем, чтоб на каждом этапе соблюдался SCRUM-подход.

Владелец продукта – лицо, ответственное за разработку. Это или заказчик, или официальный представитель, который осуществляет связь с заинтересованными лицами. Он ставит цель по разработке ПС перед разработчиками и следит за этапами разработки.

К заинтересованным лицам (стейкхолдерам, заказчикам) принято относить тех, кто инвестирует в результат проекта. Заинтересованные лица рассматривают ключевые решения, а также предоставляют обратную связь после каждого инкремента.

SCRUM-методика делит все рабочие процессы на одинаковые спринты – определенные периоды (неделя, месяц, квартал и так далее) в зависимости от состава команды и от проекта. Перед началом каждого периода (спринта) формируются задачи, а когда период заканчивается, обсуждают результаты и артефакты, после чего можно начинать следующий спринт (рис.4).

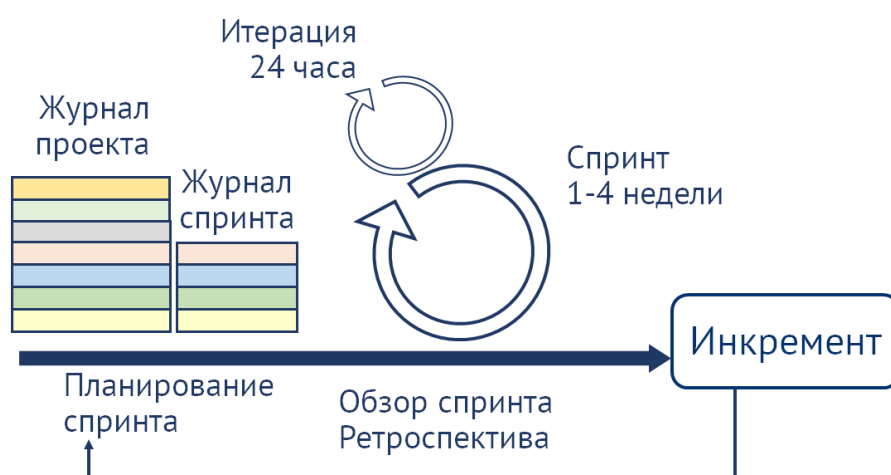


Рисунок 4 – Спринт в SCRUM

Планирование спринта осуществляют на общей встрече участников SCRUM-команды. По его результатам формируют список задач, которые необходимо решить в течение одного спринта, и перечень работ, которые должны быть для этого проделаны.

SCRUM-митинги – 15-минутные ежедневные совещания разработчиков, на которых обсуждают возникающие в процессе разработки вопросы, а также уточняются планы на ближайшие 24 часа. Эти совещания ведет SCRUM-мастер.

Обзор спринта – встреча, которая проводится по окончании каждого спринта для обсуждения его результатов внутри коллектива разработчиков.

Ретроспектива спринта – обсуждение улучшения или расширения функциональных и нефункциональных возможностей программной системы. Ретроспектива происходит после обзора спринта перед планированием следующего спринта.

SCRUM-разработка включает в себя три важных документа, называемых артефактами:

1. Журнал продукта (бэклог продукта, Product Backlog) – список требований к продукту или задач, которые реализуют в процессе разработки. Такой список формируют на основе анализа требований заказчика, собранных в виде пользовательских историй (user story).

2. Журнал спринта (бэклог спринта, Sprint Backlog) – часть требований журнала продукта, выбранная для выполнения при планировании данного спринта, а также выработанный план работ.

3. Инкремент – это сумма всех выполненных требований журнала продукта, реализованных во время текущего спринта и всех предыдущих спринтов. Это функционал, который демонстрируется стейкхолдерам по итогам спринта. Поэтому при завершении спринта производится тестирование новых функций или улучшений продукта.

Принято считать, что каждый инкремент должен быть пригоден к эксплуатации, хотя решение о запуске данного инкремента в эксплуатацию принимает владелец продукта, и оно не обязательно должно быть положительным после каждого спринта.

Цель спринта описывают аббревиатурой SMART: Specific (конкретный), Measurable (измеримый), Achievable (достижимый), Relevant (значимый), Time bound (ограниченный во времени). Каждая буква аббревиатуры SMART означает критерий эффективности поставленных целей.

SCRUM подходит для компаний с небольшой командой разработчиков, например, стартапов, внутри которых отсутствует жесткая административная иерархия, с наличием постоянного контакта между клиентом и командой для оперативного внесения правок в требования или модификации ПО, в условиях ограниченного бюджета и строгих временных рамок по завершению работы над проектом.

3. Набор практик KANBAN

Набор практик KANBAN основан на балансе: работу по разработке программной системы распределяют среди сотрудников примерно одинаково. Проект разделяют на стадии разработки. Основным критерием эффективности реализации проекта является скорость или время прохождения задачи через все стадии.

Чтобы использовать KANBAN, команда должна знать принципы, по которым работает [4, 6]:

1. Возможность изменять ход производства/разработки. При возникновении задачи, имеющей более высокий приоритет, чем находящаяся в разработке, внимание сотрудников переключается на ее выполнение.

2. Ограничение по количеству одновременно выполняемых задач на каждом этапе разработки. Таким образом, регулируется нагрузка на сотрудников.

3. Наглядное отображение процесса разработки ПС средствами сторимаппинга.

4. Постоянное улучшение программного продукта и процесса его разработки.

5. Время цикла. Время выполнения каждой задачи обязательно фиксируют и оценивают. Уменьшение времени цикла работы над выполнением конкретной задачи относят к приоритетным вопросам.

Для контроля и регулирования времени выполнения задач разработчики проводят небольшие совещания – каденции (петли обратной связи).

Каденция – это регулярное совещание, которые задают особый ритм процессу разработки. В зависимости от задач, которые требуют решения их проводят раз в день, в неделю или раз в месяц.

Для визуализации процесса работы над проектом применяют канбан-доски. На каждую задачу назначают исполнителя, срок выполнения и необходимые артефакты. При решении задача перемещается по доске от точки принятия обязательств «Сделать» (ToDo) до точки отдачи обязательств «Сделано» (Done). (рис.5).

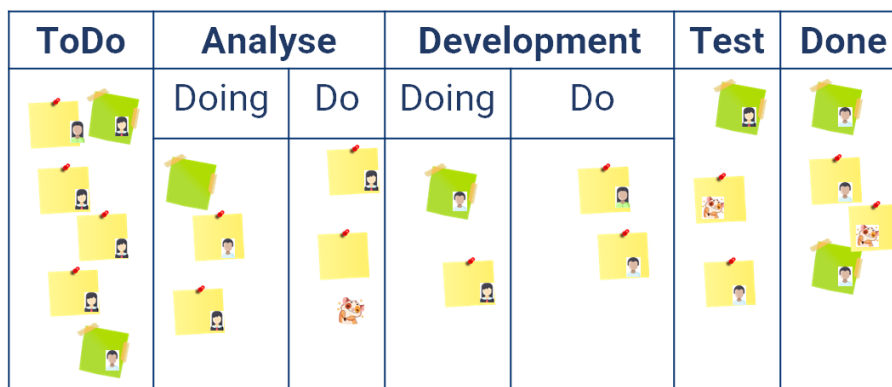


Рисунок 5 – Доска KANBAN

В каждой графе видны задачи, которые находятся на одной стадии решения. Work in progress – работа, находящаяся в системе от точки принятия обязательств до точки отдачи обязательств.

Цель – уменьшение количества «выполняющейся в данный момент работы». Если команда видит проблемы на каком-то этапе, то менее загруженные по работе сотрудники приходят на помощь коллеге.

Канбан-метод может помочь в определенных целях:

- достичь гибкой системы планирования;
- оптимизировать временные затраты;
- выстроить прозрачность, доверительные отношения между клиентами и командами;
- оперативно решать проблемы по выстраиванию процесса.

Сравнение практик между собой показано в табл.2. Основные критерии сравнения: по количеству обязательных совещаний, по наличию точки отсчета по выполнению каждой задачи проекта, по типу команды, возможности внесения изменений, деления на роли внутри команды [5-6].

Таблица 2 – Сравнение Kanban и Scrum

Kanban	Scrum
Нет совещаний	Есть совещания
Фиксация точки начала работы	Точку начала работы не фиксируют
Могут работать узкопрофильные команды	Только кроссфункциональная команда
Последовательные и плавные перемены	Кардинальные перемены
В команде нет разделения на роли	В команде есть разделение на роли

Фреймворк SCRUM подходит для изменчивого рынка, новых разработок и тесного взаимодействия разработки со стейкхолдерами.

Набор практик KANBAN не требует радикальных изменений в работе организации, но вынуждает их сменить приоритеты и акцентировать внимание на решении задач.

4. Визуализация процесса разработки

Важной составляющей рассмотренных подходов к руководству проекта является визуализация процесса разработки. Рассмотрим пример доски проекта (рис.6) в облачной системе Kaiten (<https://kaiten.ru/>).

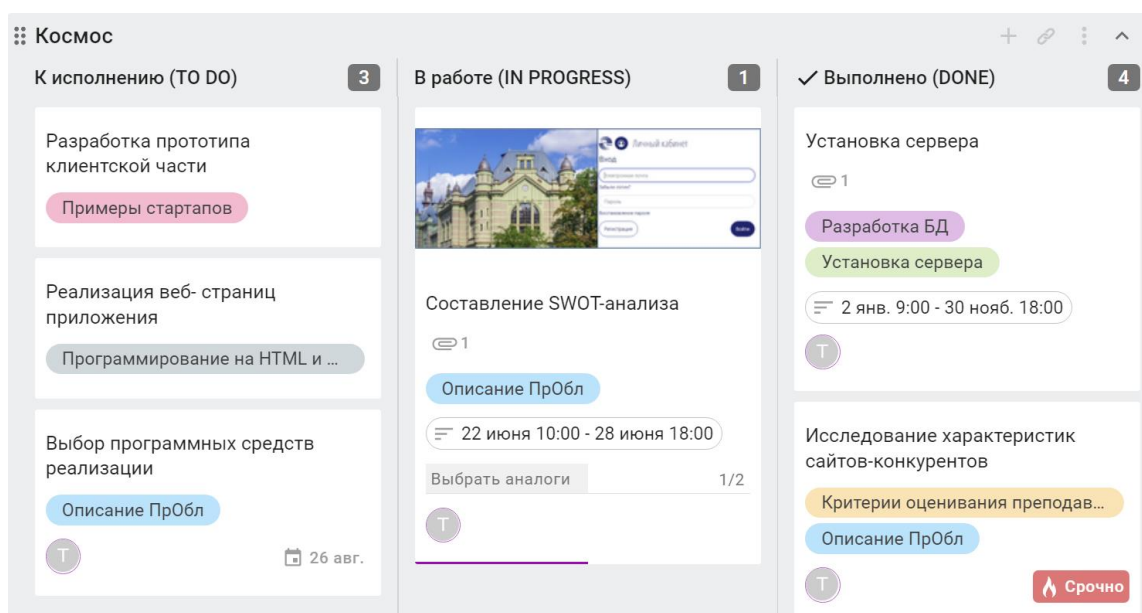







Рисунок 6 – Доска проекта в облачной системе Kaiten



В каждую группу KANBAN-доски попадают карточки с описанием задач (рис.6). Таким образом, группы превращаются в некие to-do-списки общего назначения. Карточки повышают наглядность отображаемой информации и упрощают ориентирование команды в большом потоке задач.

К тому же, в цифровом виде распределение по карточкам дает возможность задать дополнительные атрибуты (рис.7). Например, показать, кто ответственен за выполнение задачи, определить, когда она должна быть сделана, или оставить заметку с дополнительной информацией (рис.8).



Основное

-  Срок
-  Метки
-  Чек-лист
-  Ссылка
-  Загрузить файл

Связи

-  Родительская карточка
-  Дочерняя карточка

Остальное

-  Критерии приёмки
-  Google Drive

Поля





-  Создать новое поле
-  Размер
-  Timeline
-  Тип

Рисунок 7 4– Атрибуты карточки проекта

Составление SWOT-анализа

#8113796 Заказчик turnetskaya, перемещена 4 дня назад

+

→ ВЫПОЛНЕНО (...)

!

↻

⋮

Расположение

Тип

Участники

Срок

Метки

Timeline

Космос / В работе (IN PROGRESS) Q

Card

Ответственный +

Установить срок


Описание Пробл

22 июня 10:00 - 28 июня 18:00

Описание

СОСТАВЛЕНИЕ SWOT-АНАЛИЗА И РАЗРАБОТКА КОНЦЕПЦИИ ВЕБ-РЕСУРСА

Файлы




ЛЭТИ.png (591.64 KB)
Добавлен 23 июля 2023 г., 21:22, turnetskaya

Комментарии

Напишите комментарий

turnetskaya 23 июля 2023 г., 21:23

Прикрепил(а) ЛЭТИ.png (591.64 KB)



Личный кабинет

Вход

Регистрация

Забыли пароль?

Создать аккаунт

Рисунок 85 – Карточка проекта с атрибутами

5. Ретроспектива появления практик и моделей разработки ПО

Рассмотрим ретроспективу появления практик и моделей разработки ПО. В ней не показаны корпоративные методологии, например, Rational Unified Process. Внимание сосредоточено на классических моделях: водопадной, спиральной и на Agile-практиках (рис.9). Каждая модель и практика появляется как решение проблем разработки качественного ПО. В настоящее время активно развивается DevOps-практика.

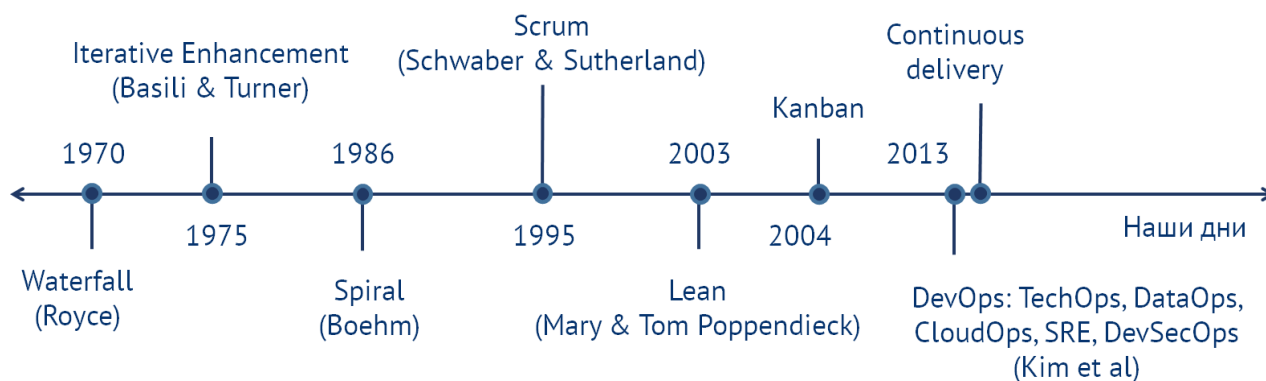


Рисунок 9 – Ретроспектива появления практик и моделей ПО

DevOps позиционируется как Agile-подход для устранения организационных и временных барьеров между командами разработчиков и других участников жизненного цикла ПО (тестировщиками, администраторами, техподдержкой), чтобы они могли быстрее и надежнее собирать, тестировать и выпускать релизы программных продуктов.

DevOps в целом – направление, специалисты которого обеспечивают процесс автоматизации и выстраивают соответствующую культуру и свод практик внутри компании. Различают разные сферы применения этого подхода при разработке ПО:

- TechOps – приближенная к области системного администрирования;
- DataOps – с фокусировкой на анализ данных;
- CloudOps – связанная с облачными решениями;
- SRE (Site Reliability Engineering) – с фокусом на обеспечение бесперебойной работы высоконагруженных сервисов;
- DevSecOps – направленная на обеспечение безопасности в рамках пайплайнов автоматизации.

Список источников

1. Орлов С.А. Программная инженерия. Технологии разработки программного обеспечения: учебник. СПб: Питер, 2020. С.640
2. Agile Manifesto. URL: <https://agilemanifesto.org/iso/ru/manifesto.html>
3. Практики Kanban. URL: <https://kanban.university/>
4. Официальный сайт SCRUM. URL: <https://scrumguides.org/scrum-guide.html>
5. Турнецкая Е.Л. Методологии и технологии проектирования информационных систем : учебно-методическое пособие / А. В. Аграновский, В. С. Павлов, Е. Л. Турнецкая ; С.-Петербург. гос. ун-т аэрокосм. приборостроения. - Санкт-Петербург : Изд-во ГУАП, 2021. - 111 с.

6. Турнецкая, Е. Л. Программная инженерия. Интеграционный подход к разработке / Е. Л. Турнецкая, А. В. Аграновский. — Санкт-Петербург : Лань, 2023. — 216 с. — ISBN 978-5-507-46898-0. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/352307> (дата обращения: 17.07.2024). — Режим доступа: для авториз. Пользователей