

# О методологиях разработки

Юрий Литвинов  
[y.litvinov@spbu.ru](mailto:y.litvinov@spbu.ru)

06.07.2023

# Программа и программный продукт



Ф. Брукс, “Мифический человеко-месяц”

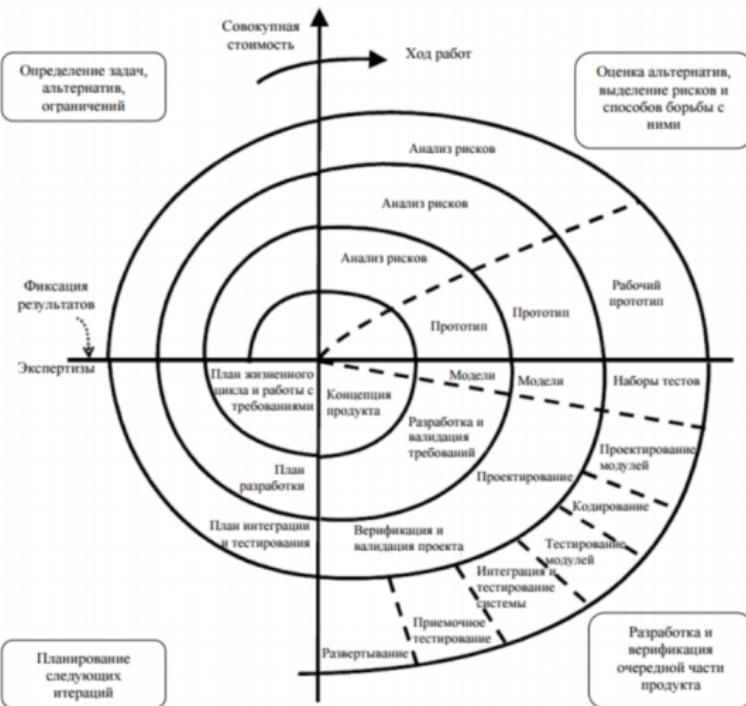
# Фазы жизненного цикла ПО

- ▶ возникновение и исследование идеи
- ▶ сбор и анализ требований
- ▶ планирование
- ▶ проектирование
- ▶ разработка
- ▶ отладка и тестирование
- ▶ сдача
- ▶ сопровождение
- ▶ вывод из эксплуатации

# Водопадная модель



# Сpirальная модель



# Методологии разработки

- ▶ Модели описывают последовательность фаз и что надо делать на этих фазах, методологии — как делать
- ▶ “Cowboy coding”
- ▶ Гибкие методологии
  - ▶ Agile Manifesto
    - ▶ Люди и взаимодействие важнее процессов и инструментов
    - ▶ Работающий продукт важнее исчерпывающей документации
    - ▶ Сотрудничество с заказчиком важнее согласования условий контракта
    - ▶ Готовность к изменениям важнее следования первоначальному плану
    - ▶ Не отрицая важности того, что справа, мы всё-таки больше ценим то, что слева
  - ▶ XP, Scrum

# Принципы Agile-разработки (1)

- ▶ Наивысшим приоритетом для нас является удовлетворение потребностей заказчика, благодаря регулярной и ранней поставке ценного программного обеспечения
- ▶ Изменение требований приветствуется, даже на поздних стадиях разработки. Agile-процессы позволяют использовать изменения для обеспечения заказчику конкурентного преимущества
- ▶ Работающий продукт следует выпускать как можно чаще, с периодичностью от пары недель до пары месяцев
- ▶ На протяжении всего проекта разработчики и представители бизнеса должны ежедневно работать вместе

# Принципы Agile-разработки (2)

- ▶ Над проектом должны работать мотивированные профессионалы. Чтобы работа была сделана, создайте условия, обеспечьте поддержку и полностью доверьтесь им
- ▶ Непосредственное общение является наиболее практическим и эффективным способом обмена информацией как с самой командой, так и внутри команды
- ▶ Работающий продукт — основной показатель прогресса
- ▶ Инвесторы, разработчики и пользователи должны иметь возможность поддерживать постоянный ритм бесконечно. Agile помогает наладить такой устойчивый процесс разработки

# Принципы Agile-разработки (3)

- ▶ Постоянное внимание к техническому совершенству и качеству проектирования повышает гибкость проекта
- ▶ Простота — искусство минимизации лишней работы — крайне необходима
- ▶ Самые лучшие требования, архитектурные и технические решения рождаются у самоорганизующихся команд
- ▶ Команда должна систематически анализировать возможные способы улучшения эффективности и соответственно корректировать стиль своей работы

# Scrum

- ▶ На самом деле, методологический фреймворк
- ▶ Не акроним (на самом деле, что-то про командную работу из рэгби)
  - ▶ Позволяет настраивать процесс, чем многие и злоупотребляют
- ▶ Ken Schwaber/Jeff Sutherland, 1995 (на самом деле, аж в 1986 году)
- ▶ Инкрементальная методология со странной организацией команды, к которой никто до сих пор не может привыкнуть

# Роли в команде

- ▶ Основные

- ▶ Product owner
- ▶ Scrum master
- ▶ Команда разработки

- ▶ Вспомогательные

- ▶ Пользователи
- ▶ Заказчики
- ▶ Вспомогательный менеджмент
- ▶ Эксперты, консультанты



# Product owner

- ▶ Представляет интересы пользователей
  - ▶ Его задача — делать так, чтобы продукт был полезен
- ▶ Формирует видение проекта и доносит его команде
- ▶ Работает с требованиями
- ▶ Приоритезирует задачи
- ▶ Отвечает за приёмку результата в конце итерации
- ▶ Единая точка принятия решений о задачах
  - ▶ Один человек
- ▶ Часть команды, работает у исполнителя
- ▶ НЕ начальник

# Scrum master

- ▶ Отвечает за соблюдение методологии
  - ▶ Организует митинги, организует процессы, считает метрики и т.п.
- ▶ Разрешает конфликты
- ▶ Защищает команду от внешних факторов
- ▶ Помогает команде самоорганизоваться
- ▶ НЕ начальник

# Команда разработки

- ▶ Те, кто, собственно, фигачат код
- ▶  $7 \pm 2$  человек
  - ▶ Принцип двух пицц
  - ▶ Должна быть очень эффективная коммуникация
- ▶ Самоорганизация
- ▶ Кроссфункциональность
- ▶ Коллективная ответственность
- ▶ Отсутствие подкоманд

# Product backlog

- ▶ Список задач
  - ▶ Фичи, баги, всякая вспомогательная работа (в т.ч. рефакторинги), мероприятия
- ▶ Единственный источник требований
- ▶ Ведётся Product owner-ом
  - ▶ Оценка задач выполняется командой
  - ▶ Уточнение задачи и декомпозиция — совместно product owner и команда
- ▶ Постоянно пополняется
- ▶ Позиция задачи в бэклоге — её приоритет
  - ▶ Поддерживается product owner-ом

# Пример бэклога

|    | A               | B        | C      | D         | E  | F          |
|----|-----------------|----------|--------|-----------|--|------------|
| 1  | Product Backlog |          |        |           | Team Velocity  | 25         |
| 2  | Priority        | Estimate | Sprint | User Type | Story  | Story Type |
| 3  | 1               | 1        | 1      | Customer  | I can see when the next show will begin for the show page I am on  | Story      |
| 2  | 2               | 1        |        | Editor    | I can select what I want to display for each "section" within the editorial content section of the page. My options include last episode, next episode, selected forum posts, selected editorial articles (tvg generated), no selection and free form text | Story      |
| 4  |                 |          |        |           |  |            |
| 3  | 2               | 1        | 1      | Editor    | I can select what picture (if any) I want to display for the corresponding content section   | Story      |
| 5  |                 |          |        |           |  |            |
| 4  | 5               | 1        | 1      | Editor    | I can select the default tab for the user to see upon visit to the page, for each show   | Story      |
| 6  |                 |          |        |           |  |            |
| 7  | 5               | 1        | 1      | Customer  | I can roll over the fields in the media player and see the various tabs change   | Story      |
| 8  | 6               | 13       | 2      | Editor    | I can modify the existing headline for any show page   | Story      |
| 9  | 7               | 1        | 2      | Customer  | I can select another show page in the drop down list next to the countdown clock   | Story      |
| 10 | 8               | 1        | 2      | Customer  | I can click "remote record" and have the show for the show page I am on record on my tivo device   | Story      |
| 11 | 9               | 1        | 2      | Customer  | I can click "join the discussion" button (or link) on the show page which takes me to the appropriate forum page for that show   | Story      |
| 12 | 10              | 1        | 2      | Customer  | I can see how many recent posts have been posted in the forum for the show page I am on  | Story      |
| 13 | 11              | 3        | 2      | Customer  | I can see how many recent replies have been posted in the forum for the show page I am on  | Story      |
| 14 | 12              | 5        | 2      | Customer  | I can blog about the show for the show page I am on (I need to be signed in to see this)   | Story      |
| 15 | 13              | 13       | 3      | Customer  | If I am not signed in, I can see a link to sign in   | Story      |
| 16 | 14              | 13       | 3      | Customer  | If I am logged in, I can click "favorites" and have the show page added to my favorites menu on the site   | Story      |
| 17 | 15              | 13       | 4      | Customer  | If I have not contributed to the poll, I can see the poll questions and submit to the poll   | Epic       |

# Ещё пример

The screenshot shows a backlog management interface with two main sections: "Current Iteration/Backlog" and "Icebox".

**Current Iteration/Backlog:**

- Iteration:** 0 of 10 points
- Points:** 383 + 12 - 18 Jira + -86
- Stories:**
  - EV3: Падение Ibus при закрытии среды при безуспешных попытках подключения [20]
  - EV3: В интерпретаторе по web какое-то не работает [20]
  - В 2-м модуле возможность послать сообщение на робот [70]
  - Поддержка онлайн-курса
  - Завершить поддержку хоткеев [20]
  - Хотки в пакете [20]
  - термин
  - Сделать более user-friendly документацию по системе проверки заданий
  - окончательно или проверка заданий
  - Userneeds/Issues/Insights
- Remaining:** 10 points
- Points:** 384 + 19 - 25 Jira + -86
- Stories:**
  - Входные данные для робота
  - окончательно или проверка заданий
  - Возможность задавать координаты старта робота и его направление через входные данные
  - окончательно или проверка заданий
  - Олимпиада НТИ, режим проверки заданий
  - Олимпиада НТИ, режим эмулятора
  - Сдача этапа 1 ЦНИИ РТК
  - Сдача этапа 3 ЦНИИ РТК
  - Поддержка задания координат оси вращения робота
  - окончательно или эмулятор этап 2

**Icebox:**

- Автоматизированный режим Лего на Linux
- Попробовать сборку на ОВС
- Попробовать собирать Пират
- Избавиться от запуска под путём на линуксе
- Добавить подтверждение личику о replaceBy
- Добавить группу эй-мотоцик, с дополнительными опциями типа повернуть на угол и прочее (есть альт)
- Добавить файл скриптов для вымы, чтобы выполнялось в стиле 70-х
- Починить работу В. Захарова, собрать легких роботов хотя бы по аналогии с #162114750
- Android легкие приложения с виджетами блоками
- Обновление всех трехсторонних компонент
- Сборка инсталляторов с помощью CI
- Масштабирование в 2d с сохранением пропорций 28-модель
- Разгон/Упрощение
- Прозрачность мак
- Комплекс комментариев и тд
- исключительное право
- Подсветка JS 2D пропала?
- Плагин для ЦНИИ РТК, базовые блоки
- Программа для этапа 1
- Базовая генерация в C++

# Спринты

- ▶ Итерация в 1-4 недели
- ▶ Результат — готовая работающая версия (инкремент)
  - ▶ Или хотя бы что-то осязаемое
- ▶ Фиксируется объём работ, практически не меняется во время спринта
- ▶ Жизненный цикл:
  - ▶ Планирование
  - ▶ Разработка
  - ▶ Демо
  - ▶ Ретроспектива

# Планирование

- ▶ Обычно полдня-день в начале спринта
- ▶ Вся команда
- ▶ Определение целей спринта и набора задач
  - ▶ Формируется sprint backlog
  - ▶ Обычно просто как верхушка Product backlog, которая лезет по объёму в спринт
- ▶ Оценка и упорядочивание задач
  - ▶ Декомпозиция, добавление технических задач
  - ▶ Planning poker
- ▶ Коллективная оценка — команда не знает, кто какую задачу будет делать
- ▶ Оценка скорости команды (team velocity)

# Planning poker

- ▶ Условные единицы, не привязанные напрямую к линейному времени
  - ▶ team velocity позволяет весьма точно пересчитать в линейный срок
- ▶ Фибоначчиева шкала (1 — сейчас сяду и сделаю, держите меня семеро; 8 — без идей как делать)
- ▶ Каждый член команды выбирает свою оценку самостоятельно
- ▶ Потом оценки оглашаются и берётся среднее (почему и покер)
- ▶ Если есть сильные расхождения, задачу обсуждают, может, декомпозириуют, и повторяют голосование
- ▶ И так по каждой задаче

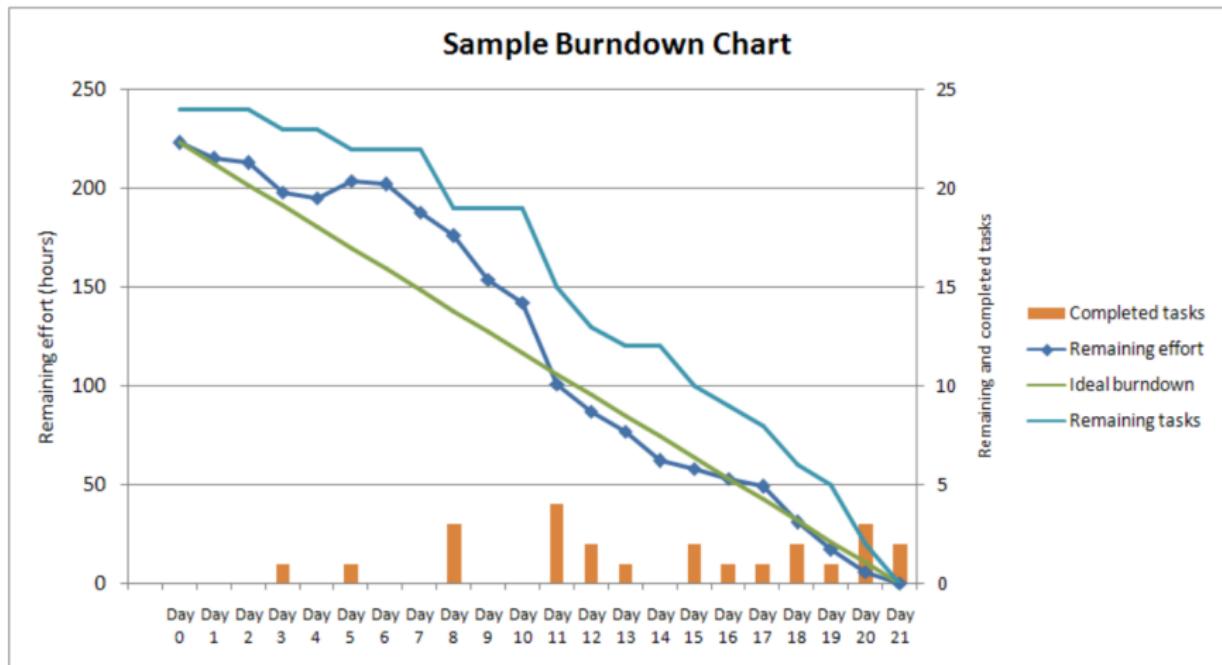
# Аналоговая канбан-доска

Артефакт доковидной эпохи



# Burndown chart

Burnout chart, как шутят коллеги



# Стэндапы

Они же “daily scrum”

- ▶ Ежедневно, не более 15 минут
  - ▶ Стоя, чтобы не было желания задерживаться (опять-таки, в былые доковидные времена)
- ▶ Всегда в одно время
- ▶ Участвуют только члены команды
- ▶ Что делал? Что буду делать? Какие проблемы возникли?
- ▶ Только делимся информацией, не решаем проблемы
- ▶ Если команд несколько, есть ещё скрам скрамов

# Демо

Оно же “Ревью”

- ▶ В конце спринта, не более 4 часов
- ▶ Демонстрация реализованного инкремента product owner-у или заказчику
- ▶ Приглашаются все участники проекта
- ▶ Проводится довольно неформально
- ▶ По результату обновляют backlog

# Ретроспектива

Чаще просто “Ретро”

- ▶ В конце спринта, тоже недолго (обычно демо и ретро в один день)
- ▶ Только команда разработки (и скрам-мастер, если он не в команде)
- ▶ Обсуждается процесс и всякие организационно-хозяйственные вопросы
  - ▶ Что было хорошо
  - ▶ Что можно улучшить
  - ▶ Например, “Купите Ваше компьютер побыстрее”

# Большая красивая картинка про спринт

## The Agile: Scrum Framework at a glance

Inputs from Executives,  
Team, Stakeholders,  
Customers, Users



Team selects starting at top as much as it can commit to deliver by end of Sprint

Sprint Planning Meeting



# ScrumBut

- ▶ Персонализация задач/багов
  - ▶ “Твой баг, ты и правь”
  - ▶ “Маша — тестировщик, задачи на тестирование ей”
  - ▶ Распределение задач в начале спринта
- ▶ Водопад внутри спринта
- ▶ Ориентация на тулы вместо прямой коммуникации
- ▶ 6-12-недельные спринты, перерывы между спринтами
- ▶ Big Design Up-Front (BDUF)
- ▶ Отсутствие Scrum master'а
- ▶ Демо по принципу “Да мы ничего особого не сделали в этом спринте”
- ▶ Наличие тимлида

# Когда Scrum может работать плохо

- ▶ Fixed-cost/fixed-time проекты
- ▶ Безответственные, низко мотивированные работники
- ▶ Слишком узкоспециализированные работники
- ▶ Неполноставочники
- ▶ Большое количество внешних зависимостей
- ▶ Legacy или системы повышенной надёжности
- ▶ Распределённые команды
  - ▶ За последний год все научились работать удалённо уже

# Заключение

