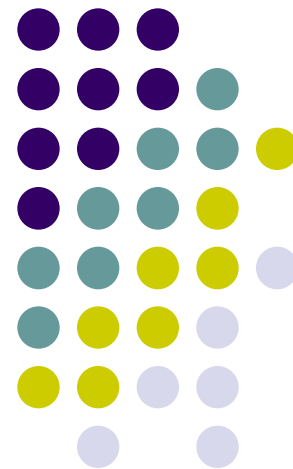


# Agile.

*Петух на церковном шпиле, хоть и железный,  
быстро сломался бы под ударами ветра,  
если бы не овладел высоким искусством  
поворачиваться, куда ветер дует.*

Генрих Гейне





# Проблемы разработки ПО

- Отсутствие эффективной методики разработки ПО ведет к непредсказуемости, повторению одних и тех же ошибок и пустой трате сил.
- Заказчики недовольны постоянно переносимым графиком сдачи, растущим бюджетом и низким качеством.
- Разработчики приходят в уныние от того, что приходится сидеть на работе все дольше, а продукт становится только хуже.

# Проблема №1



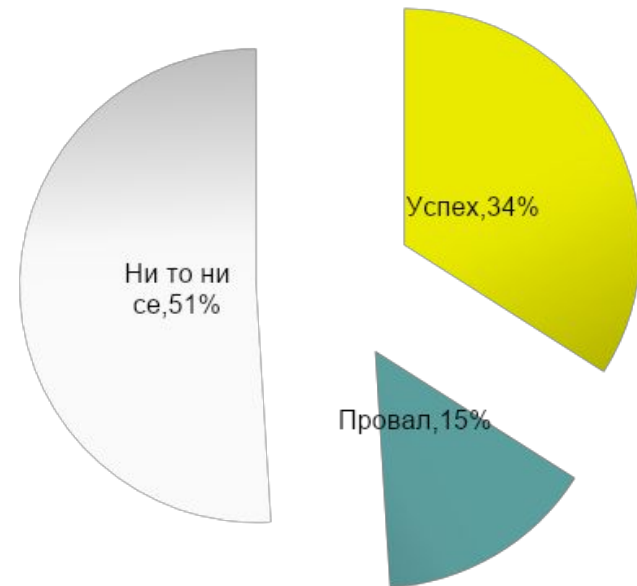
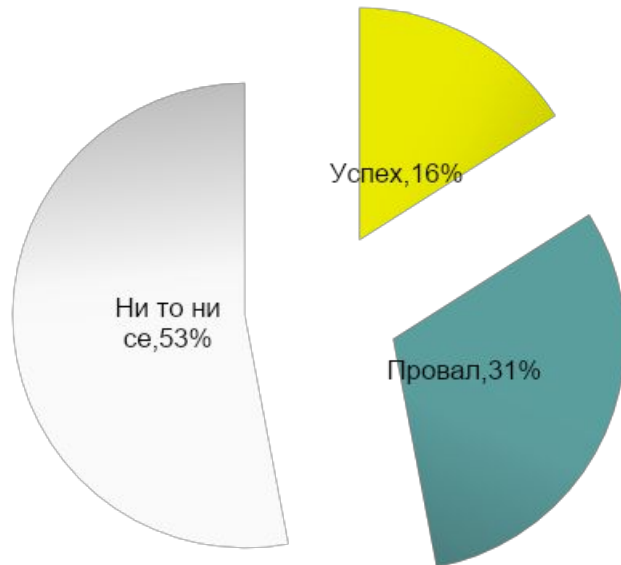
- Написание софта – сложная задача





# Проблема №2

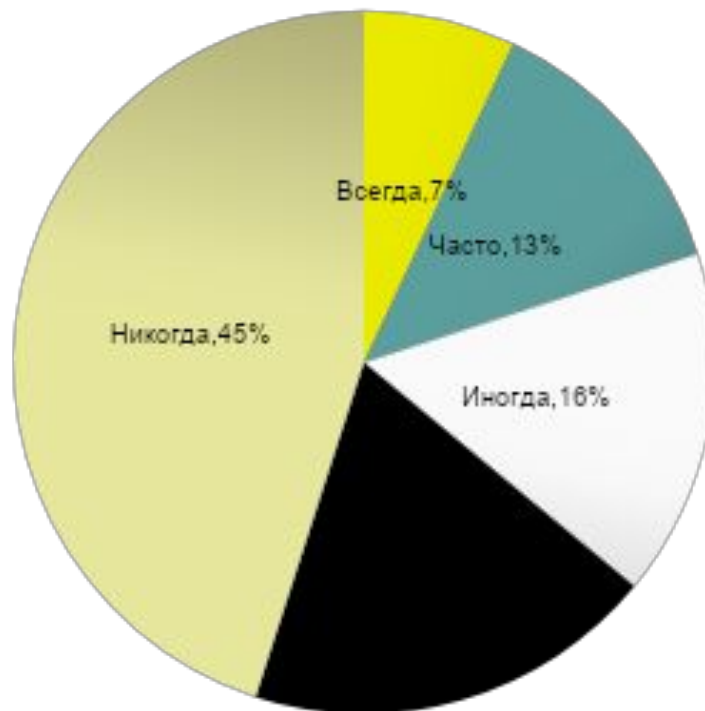
- Очень мало успешных проектов





# Проблема №3

- Программа делает не то, что нужно пользователям



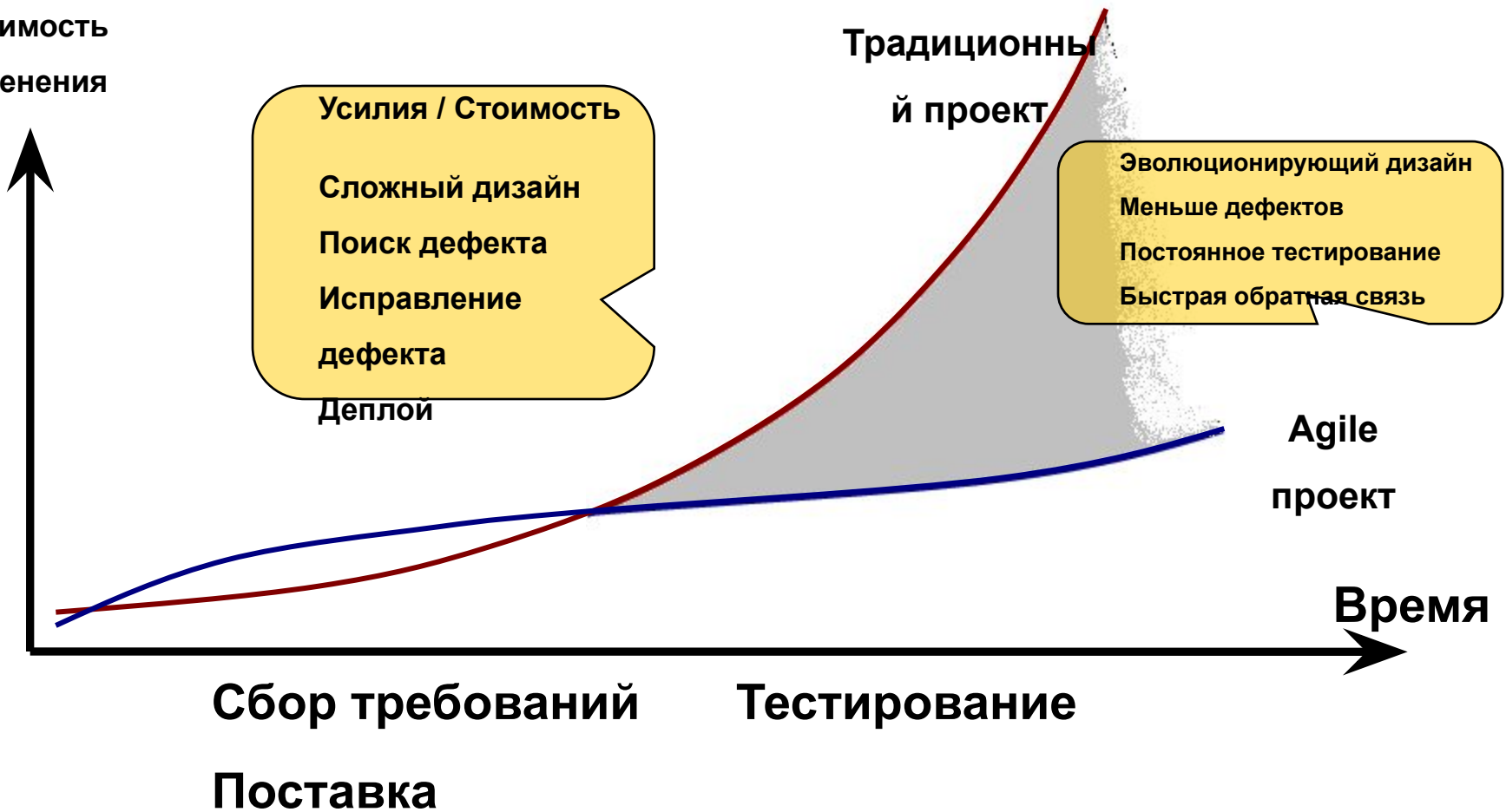
Средний процент фактического использования функциональности успешных проектов, в которых использовался обычный подход к сбору требований и документации



# Проблема №4

- Сложно вносить изменения

Стоимость  
изменения



# Методологии

- Waterfall
- Spirale
- Agile
  - Scrum
  - XP
  - Lean
  - ...





# Водопад

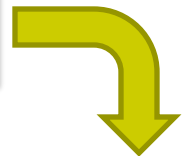
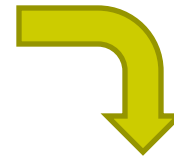
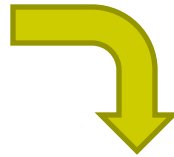
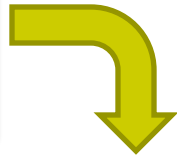
Анализ  
требовани

Дизайн

Разработка

Тестировани

Поддержка







# В чем проблема?

- Единственный период, когда можно что-то узнать о проекте – начало.
- Тестирование откладывается на последнюю фазу, когда уже поздно что-то менять
- Обозначение проблемы становится проблемой
- Избыточная специализация
- «Это не моя проблема»



# Чего мы хотим?

- Любое изменение, в любое время, в любом порядке
- Продуктивность, качество, низкая стоимость, высокая мораль
- Реальный прогресс
- Учиться на ошибках как можно раньше
- Меньше административной работы, больше работы, которая приносит пользу

# Ограничения



# Организация Agile Alliance

## <http://agilemanifesto.org>



- Группа экспертов, назвавшаяся *Agile Alliance* (сообщество профессионалов, занимающихся гибкими методологиями разработки), собралась в 2001 году, чтобы обсудить ту шкалу ценностей и принципы, которые позволили бы коллективам программистов быстро вести разработку и оперативно реагировать на изменения.

# Авторы Agile манифеста



Jeff  
Sutherland



Ken  
Schwaber



Mike  
Cohn



Alistair  
Cockburn



Martin  
Fowler



Ron  
Jeffries



Kent Beck



# Agile Manifesto



Люди и их  
взаимодействие

важнее,  
чем

Процессы и  
инструменты

Работающее  
программное  
обеспечение

важнее,  
чем

Исчерпывающая  
документация

Взаимодействие с  
заказчиком

важнее,  
чем

Обсуждение  
контракта

Реагировать на  
изменения

важнее,  
чем

Следовать плану

# Кому нужен этот ваш Agile?



Google

Яндекс

Microsoft

Рамблер

Yahoo

LinguaLeo

Philips

Adv

Siemens

Red Keds

Nokia

Luxoft

IBM

Deutsche Bank

BBC

Альфа банк

# Agile



- Гибкий подход к разработке ПО.
- Лучшие практики:
  - Scrum
  - XP
  - TDD, etc.



**"Agility is not a technology, science, or product but a culture"** (Philippe Kruchten)



# Люди и их взаимоотношения важнее процессов и инструментов



- Люди – важнейшая составная часть успеха.
- Самый лучший процесс не спасет проект от провала, если в команде нет сильных игроков.
- Однако плохой процесс может сделать даже сильнейших игроков неэффективными.

# Работающая программа важнее исчерпывающей документации (1)



- Программа без документации – это ужас. Код – не самое подходящее место для описания назначения и структуры системы.
- Команда обязана подготовить понятные человеку документы, в которых описывалась бы система и обосновывались принятые проектные решения.

# Работающая программа важнее исчерпывающей документации (2)



- Однако слишком много документации хуже, чем слишком мало.
- На создание огромных томов документации уходит много времени и еще больше – на синхронизацию их с кодом.
- Если же документация не соответствует коду, то становится большой и запутанной ложью и источником дезинформации.

# Первый закон документирования Мартина



- *Не создавайте документ, пока необходимость в нем не станет очевидной и срочной.*



# Плодотворное сотрудничество с заказчиком важнее формальных договоренностей по контракту (1)



- Чтобы проект оказался успешным, необходимо регулярное и частое общение с заказчиком.
- Заказчик программы должен не полагаться на контракт или техническое задание, а плотно контактировать с командой разработчиков, регулярно высказывая свое мнение о том, что они сделали.

# Плодотворное сотрудничество с заказчиком важнее формальных договоренностей по контракту (2)



- Самые лучшие контракты – те, что оговаривают способы взаимодействия заказчика с разработчиками.



# Оперативное реагирование на изменения важнее следования плану



- Способность реагировать на изменения часто определяет успех или провал программного проекта.
- Строя планы, необходимо следить за тем, чтобы они были гибкими и могли адаптироваться к изменениям в бизнесе и технологиях.
- Ход работы над программным проектом нельзя планировать на длительный срок.

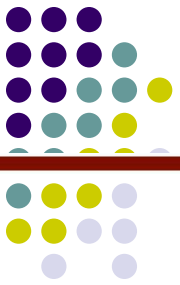


# 12 ПРИНЦИПОВ АГИЛЕ-МАНИФЕСТА





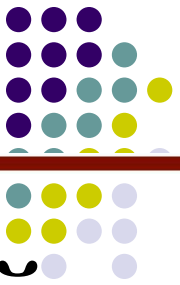
# Agile-манифест, 12 принципов



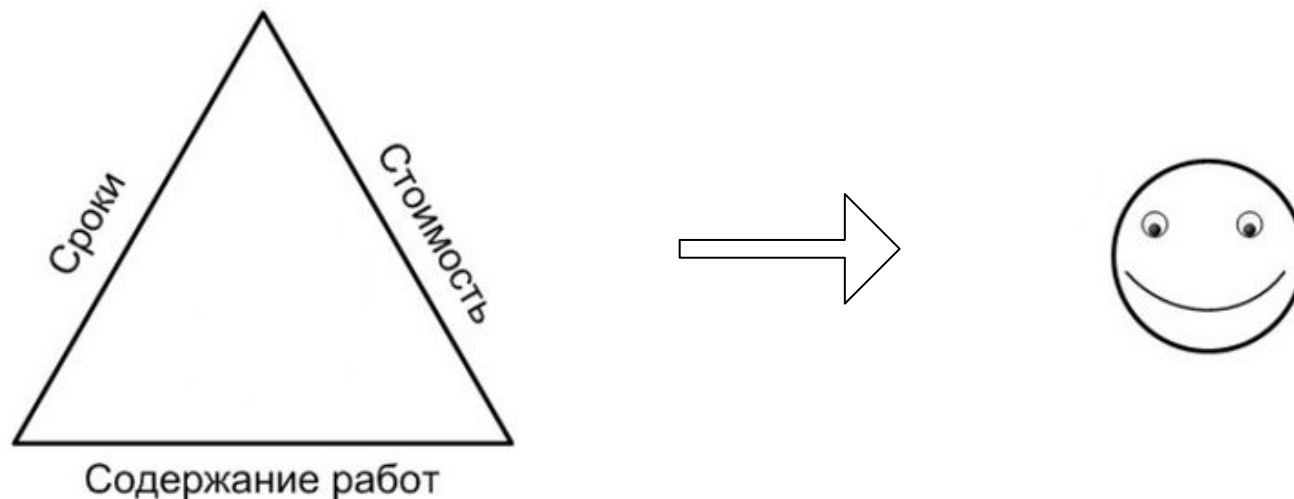
## Основополагающие принципы Agile-манифеста



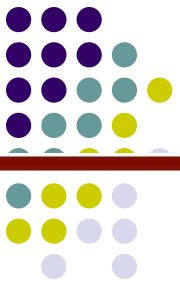
# Agile-манифест, принцип №1



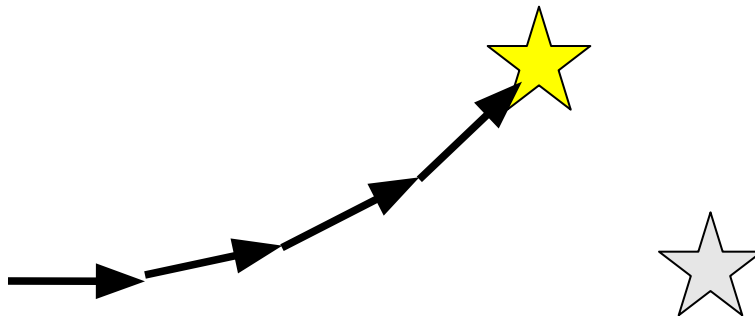
**Удовлетворение потребностей заказчика, благодаря регулярной и ранней поставке ценного программного обеспечения**



# Agile-манифест, принцип №2

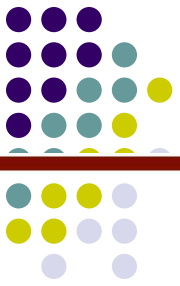


**Изменение требований  
приветствуется, даже на  
поздних стадиях разработки**



# Agile-манифест, принцип №3

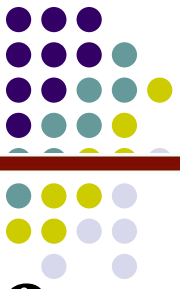
---



## Частая поставка рабочего программного обеспечения



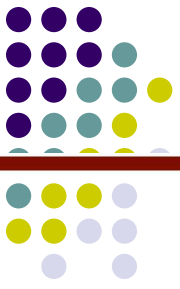
# Agile-манифест, принцип №4



**Е**жедневное общение заказчика с разработчиками на протяжении всего проекта



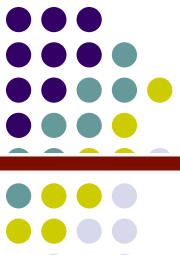
# Agile-манифест, принцип №5



**Проектом занимаются  
мотивированные личности,  
которые обеспечены нужными условиями  
работы, поддержкой и доверием**



# Agile-манифест, принцип №6

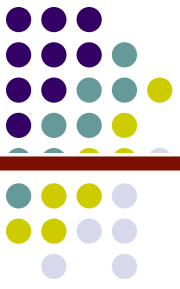


## Рекомендуемый метод передачи информации — личный разговор





# Agile-манифест, принцип №7



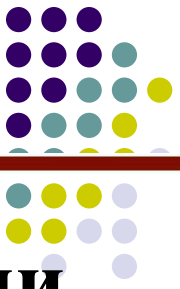
**Работающий продукт —  
основной показатель прогресса**





# Agile-манифест, принцип №8

---

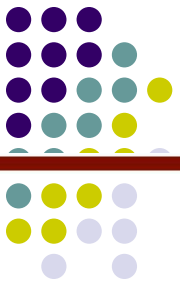


**Спонсоры, разработчики и пользователи  
должны иметь возможность поддерживать  
ПОСТОЯННЫЙ ТЕМП  
на неопределённый срок**



# Agile-манифест, принцип №9

---

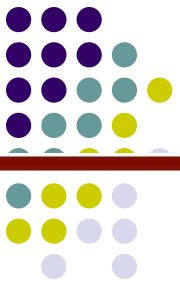


## Внимание к техническому совершенству и качеству проектирования



# Agile-манифест, принцип №10

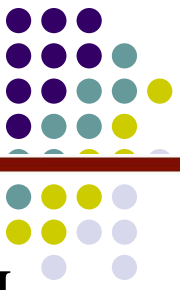
---



**Простота —  
искусство не делать лишней работы;**



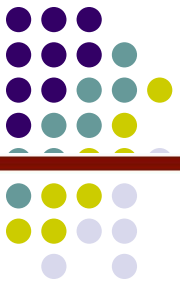
# Agile-манифест, принцип №11



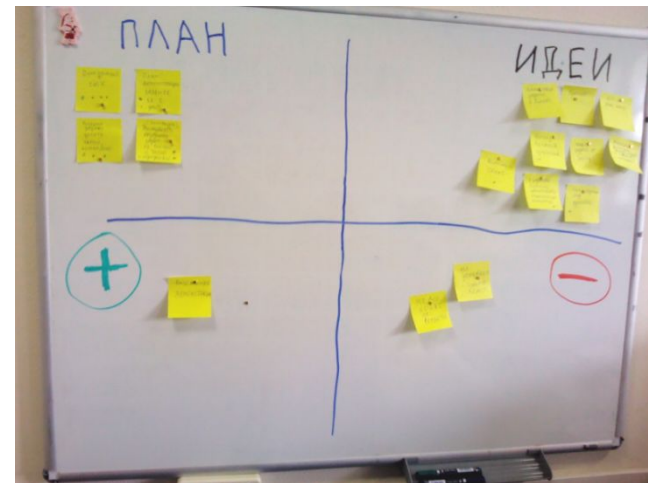
**Лучшие требования, архитектурные и технические решения рождаются у самоорганизующихся команд.**



# Agile-манифест, принцип №12

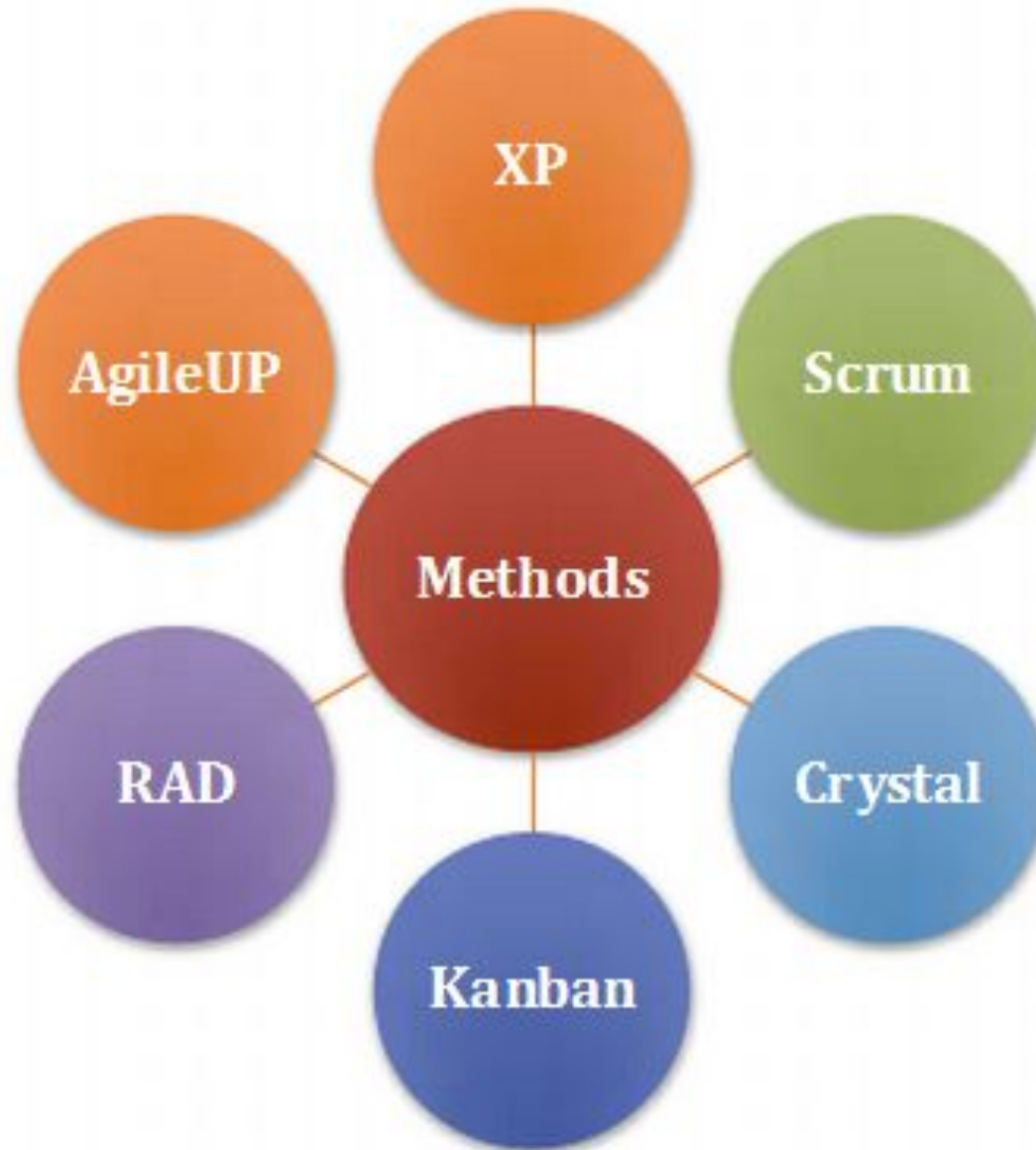
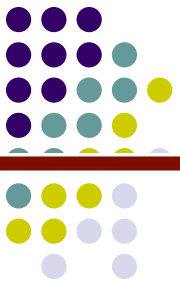


**Команда должна систематически анализировать возможные способы улучшения эффективности и соответственно корректировать стиль своей работы**



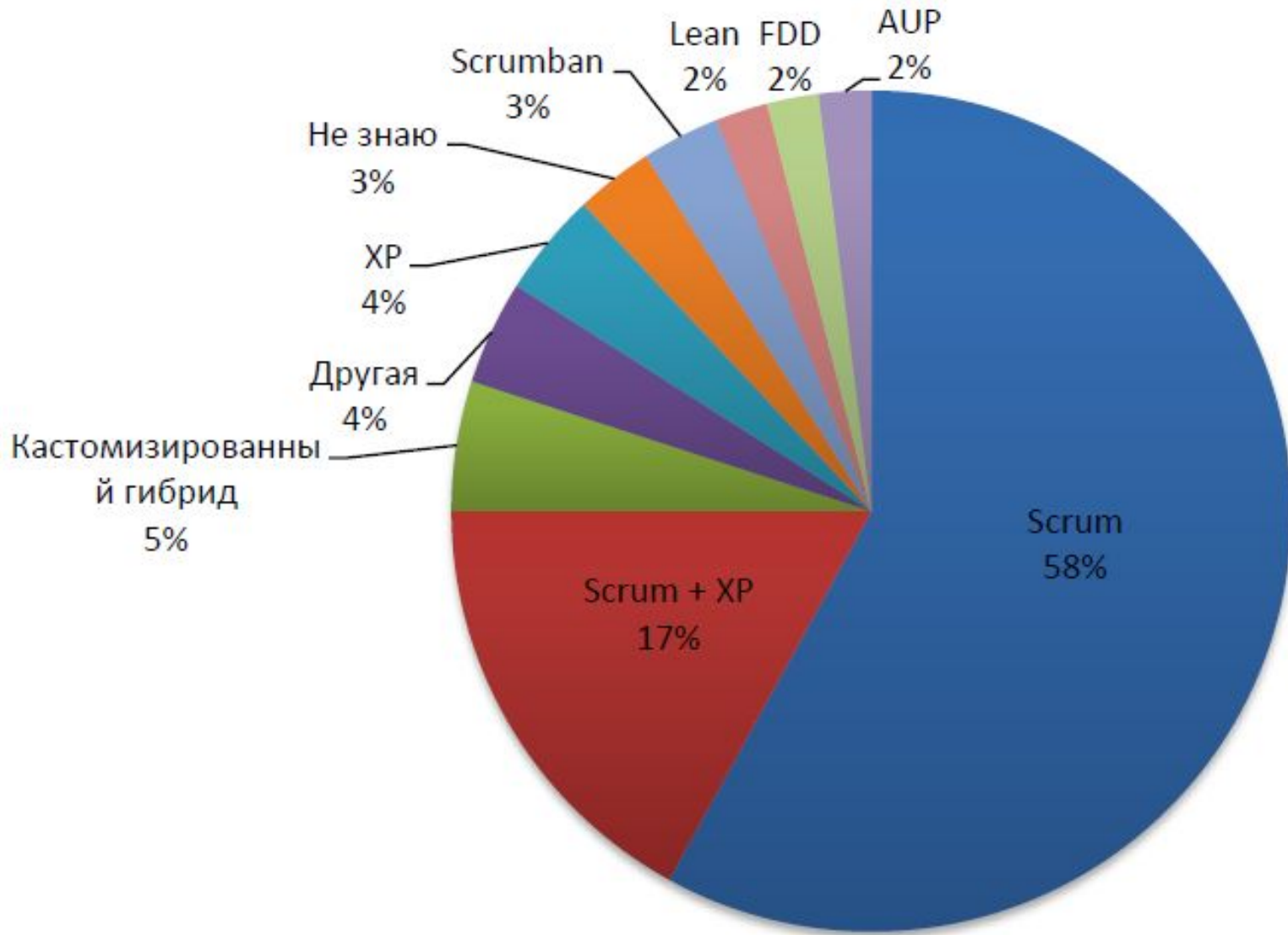
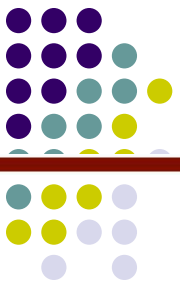
# КТО ЭТО Agile?

---

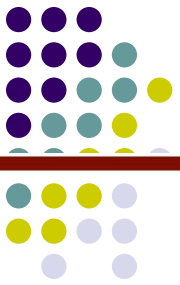




# Кто это Agile?



# Кто это Agile?







Мы собираемся  
попробовать кое-что  
под названием  
"Экстремальное  
Программирование".



www.dilbert.com scottadams@aol.com

Это значит, больше  
никакого планирования и  
никакой документации.  
Просто начинайте писать  
код и жаловаться.



© 2007 Scott Adams, Inc./Dist. by UFS, Inc.

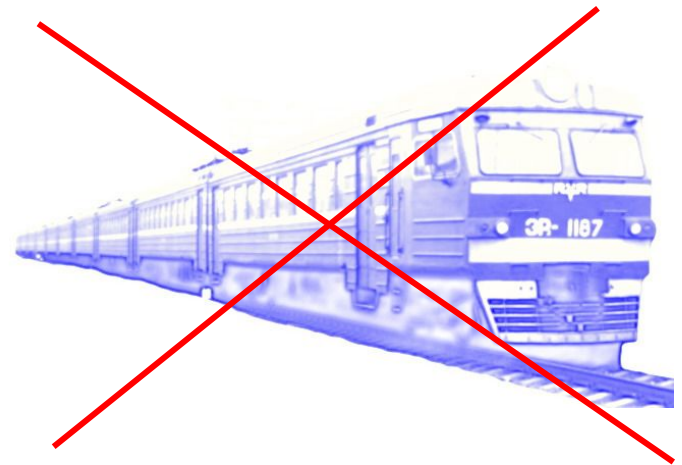
Так вот  
как это  
называется **Твоя  
ШКОЛА.**





# Ценности Agile

- Гибкость и простота
- Частые релизы
- Самоорганизующаяся команда
- Больше общения





# Гибкость и простота

- Agile-процессы готовы к изменениям требований даже на поздних этапах разработки.
- Важна простота - искусство увеличения объема работ, которых удалось избежать.





# Частые релизы

- Наивысший приоритет - удовлетворенность заказчика:
  - ранние и периодические поставки ПО
  - ПО работающее и ценное для заказчика
- Продолжительность каждой итерации - от пары недель до пары месяцев.
- Предпочтение - коротким интервалам.

# Самоорганизующаяся команда



- Над проектом работают мотивированные люди.
- Создаются все условия, поддержка и полное доверие.
- Самые лучшие архитектуры, требования и дизайны систем создаются самоорганизующимися командами.
- Команда сама организует оптимальный процесс.





# Больше общения

- Потенциальные пользователи системы и разработчики должны работать **вместе** на протяжении всего проекта.
- Самый действенный и эффективный способ обмена информацией как внутри команды разработчиков, так и с внешним миром - **непосредственное общение.**

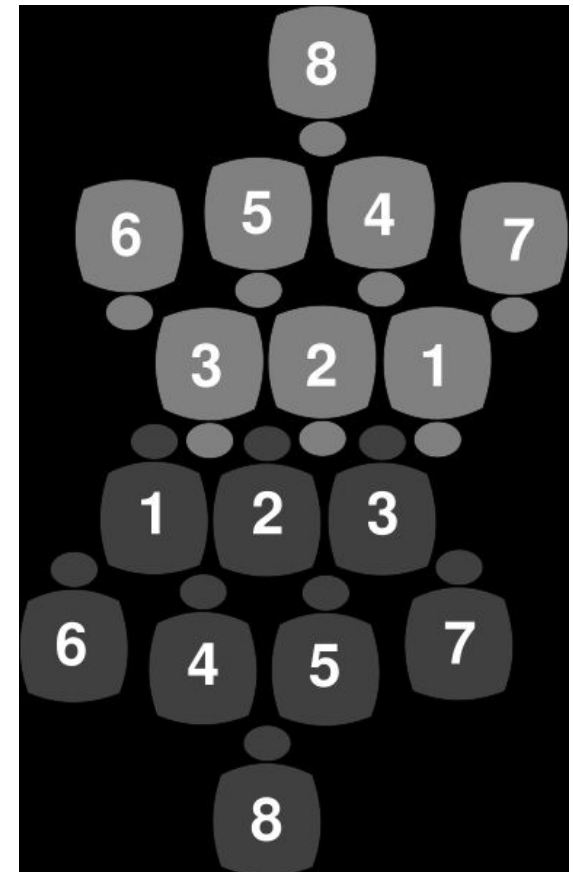
# Scrum



Наиболее распространенная практика разработки в Agile.

Ключевые термины:

- Product backlog
  - User story
  - Product owner
- Sprint
  - Sprint backlog: tasks
- Daily scrum
  - Scrum master
  - Taskboard





# Product Backlog

- Содержит список функциональных единиц системы (“user stories”), запланированных на след релиз

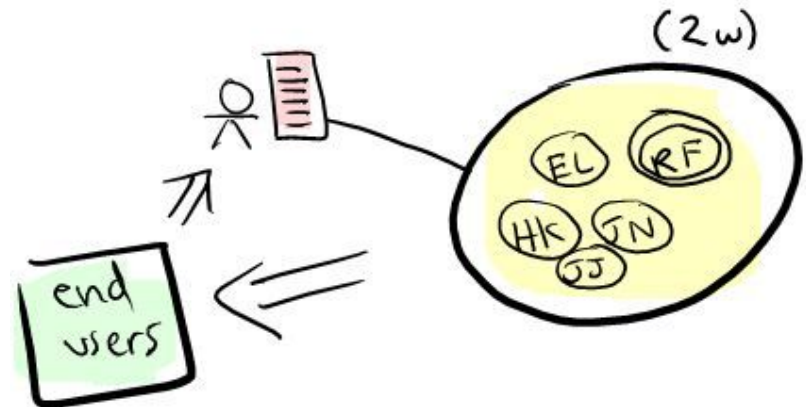
ID	Важн	Название	Описание	Как показать
248	75	Заставка (splash screen)	Как пользователь я хочу видеть заставку пока приложение открывается.	1. Запустить приложение – заставка показ. до появления главного окна





# Product Backlog

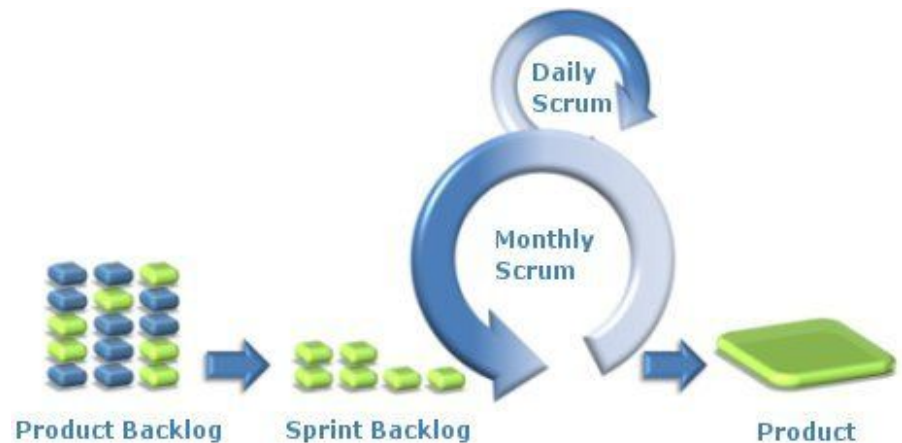
- Product backlog один на весь релиз
- Им владеет менеджер продукта (“**product owner**”)
- Он не статичен – записи можно добавлять, удалять, менять им приоритет
- Общедоступен, но поддерживается одним человеком





# Спринт (Sprint)

- Фаза разработки состоит из нескольких итераций – спринтов.
- Обычно спринт длится 2-4 недели.
- Этапы:
  - Планирование
  - Разработка
  - Демонстрация
  - Ретроспектива



Sprint Planning	Daily Scrum	Sprint Review
Review	Inspect	Accept or Reject
Estimate	Adapt	Review Backlog
Commit	Commit	Retrospection



# Sprint Backlog

- Описывает задачи, запланированные командой на спринт
- Задачи – действия, необходимые для реализации запланированной на спринт функциональности
- В описание задачи входит ее оценка



# Планирование (Sprint Planning)

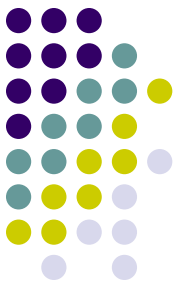


- Проводится в начале спринта
- Участвует вся команда
- User stories разбиваются на задачи и оцениваются членами команды
- В результате команда подписывается на ту функциональность, на которую хватает времени спринта



# Оценка

- Для оценки выбирается единица – идеальный человеко-день...или зеленый крокодил
- Следует оценить помехи (например *focus factor* между 0 и 1) перед каждым спринтом
- Результаты предыдущего спринта помогают лучше запланировать следующий



# Ежедневный скрам (Daily Scrum)



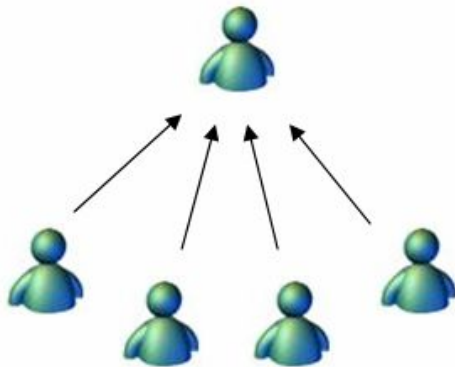
- Проводится каждый день в фиксированное время
- Рекомендуется проводить стоя в течение 10-15 минут
- Если что-то нужно обсудить, назначается время после скрама



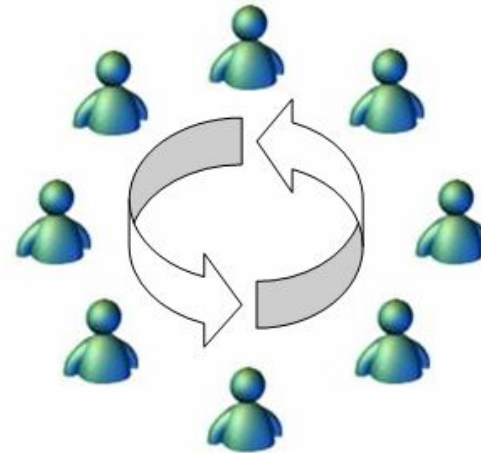


# Вопросы

- *Scrum Master* спрашивает каждого:
  - Что ты делал?
  - Что ты собираешься делать?
  - Какие были проблемы?









Flow of Conversation in  
a Daily Pseudo-Scrum Meeting



Flow of Conversation in  
a Daily (real) Scrum Meeting

# Sprint whiteboard



PLANNED	IN PROGRESS	READY FOR TEST	DONE	BURN-DOWN
				 <p>Remaining work</p> <p>Days</p>
				UNPLANNED
				



# Демонстрация (ревью)



- В конце каждого спринта проводится ревью
- Это демонстрация реализованной функциональности
- В ней может участвовать любой человек, задействованный в проекте
- В идеале после каждой демонстрации можно отправлять продукт заказчику

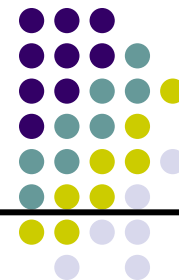




# Ретроспектива спринта

- После каждого спринта (ревью)
- Участвуют все члены команды
- Цель - осознать:
  - Что было хорошо?
  - Что могло бы быть лучше
- Это обсуждение процесса, а не технических сложностей

# Обзор активностей



Активность	Проводит	Участники	Артефакты
Планирование	Скрам-мастер	Команда	Product, Sprint backlog
Ежедневный скрам	Скрам-мастер	Команда	Sprint whiteboard, backlog
Ревью спринта	Скрам-мастер	Все	Работающее ПО
Ретроспектива	Скрам-мастер	Команда	Записи



# Ссылки

- <http://agilemanifesto.org/>
- [http://en.wikipedia.org/wiki/Agile\\_software\\_development](http://en.wikipedia.org/wiki/Agile_software_development)
- <http://agilerussia.ru/index.php>
- *Agile Project Management with Scrum.* By Ken Schwaber.