У нас само-управляема я команда

Асхат Уразбаев
Никита Филиппов
ScrumTrek

ScrumTrek
эффективные процессы

Организация самоорганизации команды

Самоорганизация команды

- Что это такое?
- Зачем это нужно?
- Как этого добиться?
- Что может помешать?
- Когда это **ОПАСНО**?

Вопрос

- У вас есть три колхозника
- Нужно прополоть 3 грядки
- Что вы сделаете
  - Будете указывать каждому его куст
  - Дадите каждому собственную грядку
  - Дадите им самоорганизоваться

Взгляд со стороны менеджера

- Куст. Микроменеджмент. Слишком много менеджерской работы
- Грядка. Это точно сработает
- Самоорганизация. Результат не гарантирован

Делегирование

- Так это и бывает
- Каждый получает свою область ответственности
- Потому что так **проще управлять**

Проблема №1. Проблема ответственности

- "Программисты не тестируют!"
- "А у меня на машине все работает!"
- "Настоящий мужик свои проблемы решает сам!"

К пуговицам претензии есть?

Проблема №2. Низкое качество

- Слабые программисты пишут плохие модули
- Хорошие программисты не учат молодых программистов
- Аналитики пишут хорошие документы, а не добиваются хорошего продукта

Проблема №3. Низкая мотивация

- Разработчики любят
  - Крутые технологии
  - Покопаться в коде
- Разработчики не любят
  - Писать документацию
  - Фиксить баги
- Разработчики слабо заинтересованы в достижении бизнес-целей проекта

- Людей мотивирует то, за что они ВНУТРЕННЕ несут ответственность
- Иначе говоря, чувствуют ОЩУЩЕНИЕ СОБСТВЕННОСТИ

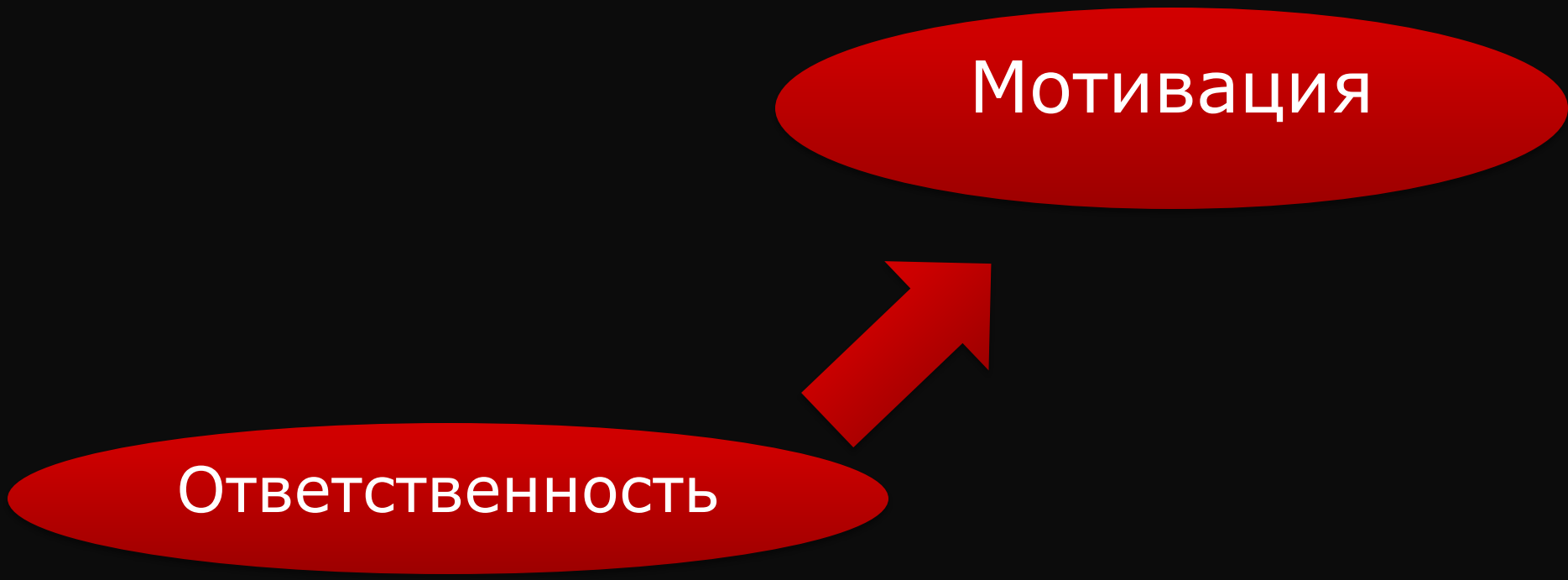
Ответственность это полномочия

- “Полномочия — это в первую очередь ответственность”

Владимир Путин

Кто принимает решение - тот несет ответственность

Выше ответственность – выше мотивация



Самоорганизация

- Команда сама координирует свою работу
- Общая ответственность за продукт
- Общий пул задач
- Коллективное принятие решений

- Кроссфункциональность

Кроссфункциональность

By component

"Понятие команды, где никто никому не говорит, что делать и где отсутствуют персональные заслуги, едва ли понравится программисту, который по своей природе тщеславен и стремится обладать каким-то участком работы. Это, эгоистическое на первый взгляд, стремление предполагает ответственность и трепетное, личное отношение к части продукта"

http://thekonst.net/ru/propaganda/291

By Role

"По своей натуре программист не любит правила, аналитик любит, тестер к правилам толерантен. Посему отличный аналитик-программист-тестер в одном человеке, может вызвать в внутренний когнитивный диссонанс обостренный разстроением личности :-))))"

http://pmant.livejournal.com/7609.html

Команда

… небольшая группа людей с дополняющими навыками, с общей целью, стремящаяся улучшить свою производительность и чуствующая ответственность по отношению к друг другу…

Katzenbach, Smith, *“The Wisdom of Team”*

Типы кроссфункциональности

By Feature	• Biz. domains
By component	• Component owners
By Role	• Analyst, Tester, Programmer
By competence	• C++, Java, Silenium, Oracle etc
Full crossfunctionality	• Все могут делать все

This is not AGILE

Agile

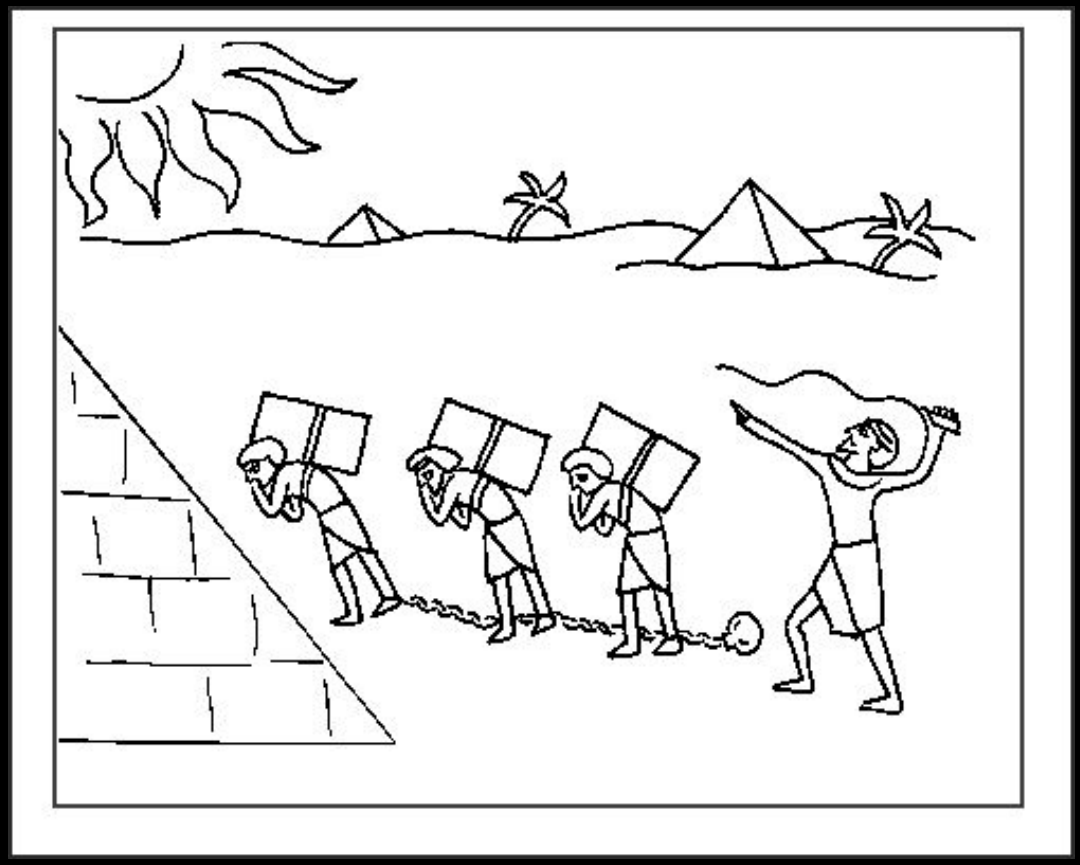
TRUE
AGILE

TRUE
TRUE
AGILE

Кроссфункциональность без самоорганизации

- Менеджер
  - Глубоко декомпозирует фичи
  - Раздает задачи
  - Управляет координацией работ
  - Вовремя обнаруживает проблемы на стыке
  - Связывает разработчиков
- Микроменеджмент!

Итерации без самоорганизации



Сравнение производительности

- Команда без самоорганизации
- Самоорганизующаяся команда

Условия модели

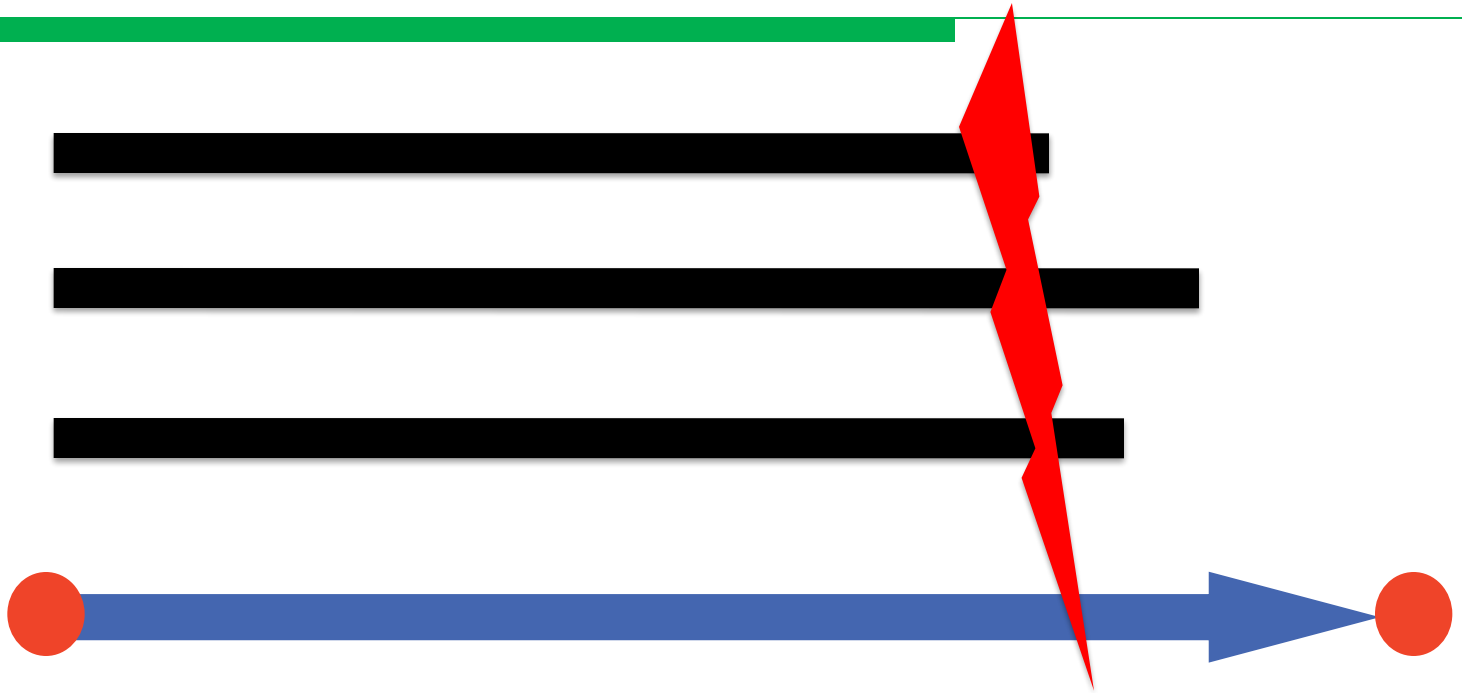
- 3 человека
  - Производительность в SP/итерацию
- Фичи
  - Оценка в Story Points
  - Реальные трудозатраты

Расчет

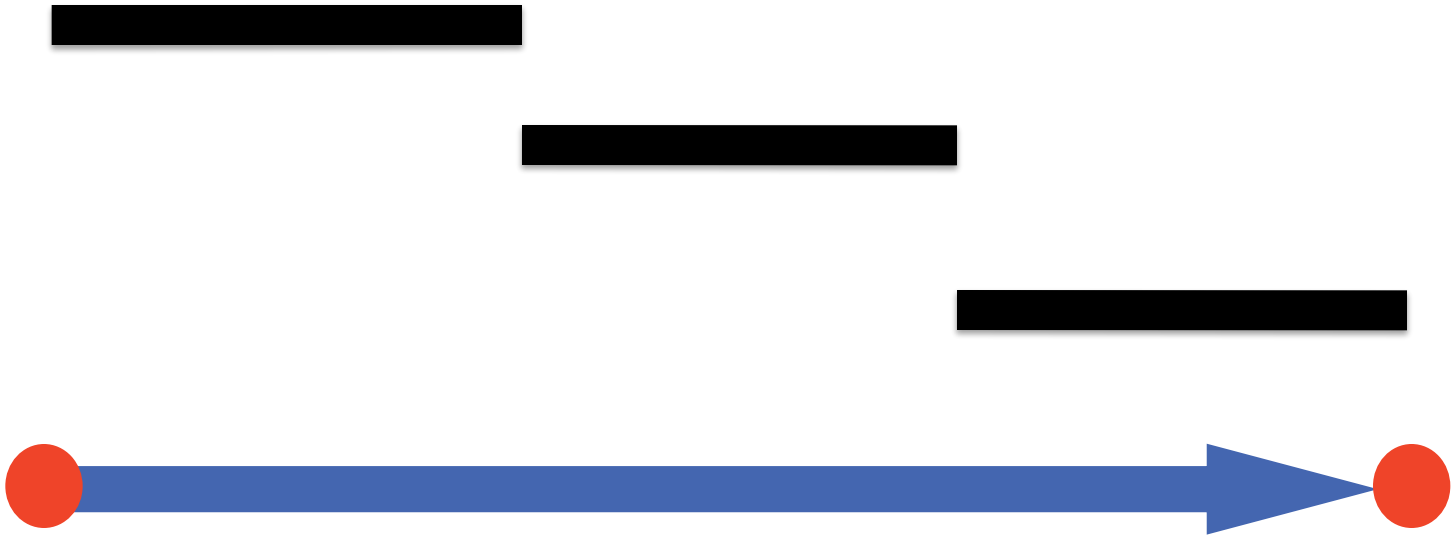
- Команда без кроссфункциональности
  - Каждую фичу может взять один человек
  - Заранее известно, кто какую фичу делает
- Самоорганизующаяся команда
  - Фичу разрабатывают совместно
- Несделанная фича переносится на следующую итерацию

Разработка внутри итерации

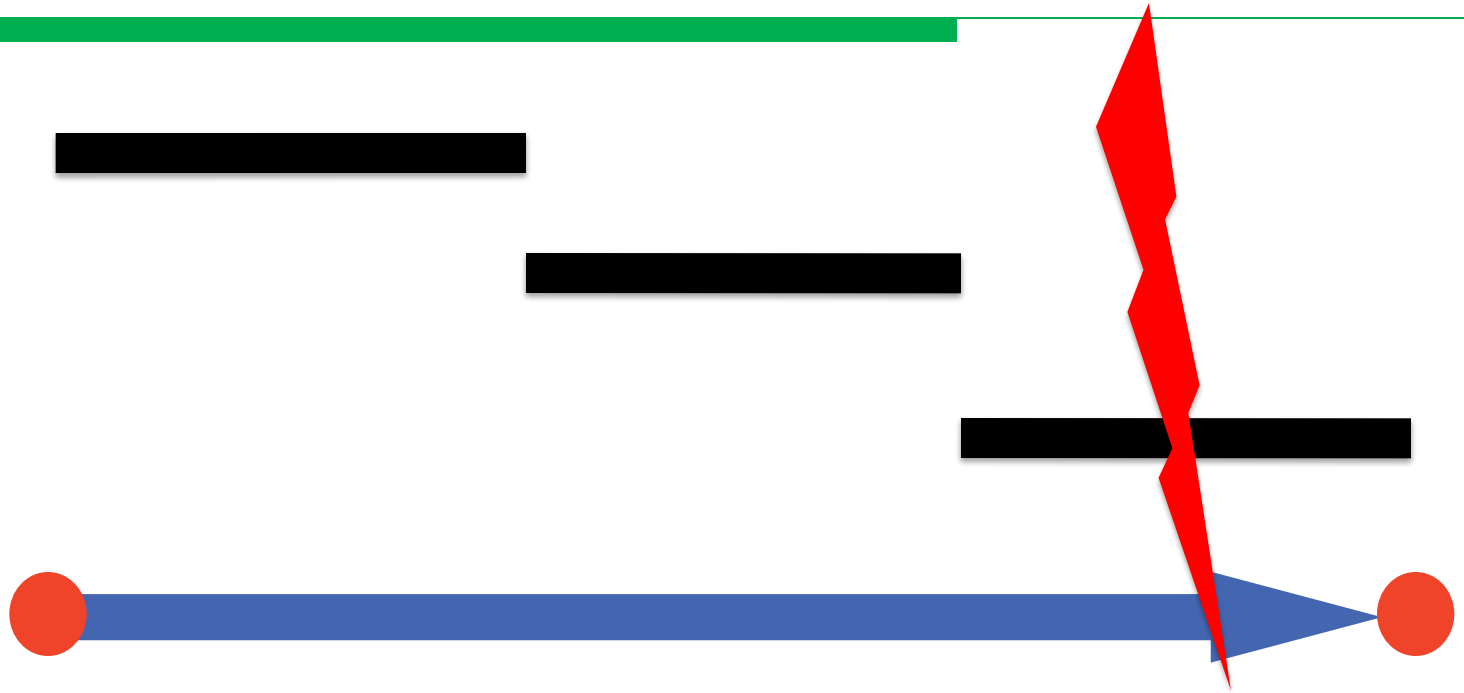
Разработка внутри итерации



Разработка внутри итерации



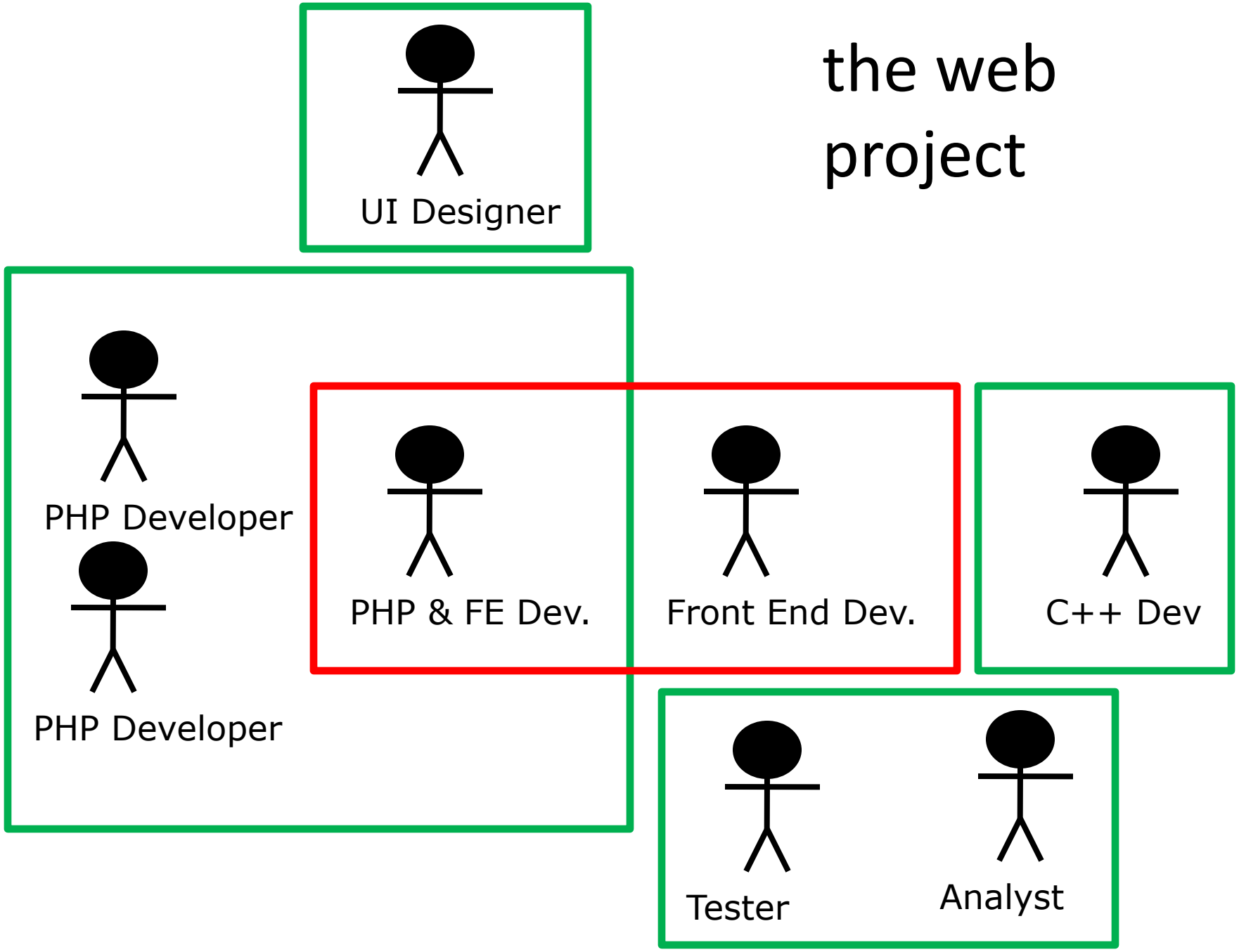
Разработка внутри итерации



Выводы

- Команда без кроссфункциональности
  - Менее производительна
  - Менее предсказуема
- Нет кроссфункциональности?
  - Длиннее итерация
  - Глубже декомпозиция по фичам

the web project



Учимся самоорганизации

Ускорить отчеты

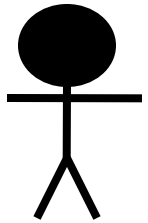
Product Owner

Java Dev

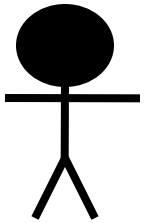
Oracle Dev

JS Developer

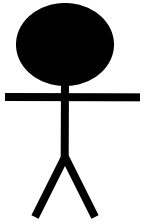
Создать бизнес правило



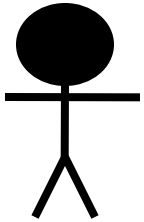
Product Owner



Oracle Dev



Java Dev



JS Developer

Web2.0!

Product Owner

Oracle Dev

Java Dev

JS Developer

Учимся кросс-функциональности

- Совместное планирование
- Общий план (а не по людям)
- Planning poker для задач
- Выбор задач по приоритетам
- Limit WIP

Разные степени кроссфункцональности

- В чем различие при…
  - Планирование
  - Standup
  - Итерация

Планирование

- By Feature
  - Все декомпозируют и оценивают свои задачи самостоятельно
- By Component
  - Все декомпозируют задачи совместно и оценивают каждый свою
- By Role & By competence
  - Все декомпозируют задачи совместно и оценивают в своих группах совместно
- Full
  - Все совместно

Daily Scrum

- By Feature, By Component
  - Daily Scrum не нужен
- By Role
  - «Что ты СДЕЛАЛ вчера?»
- By Competence & Full
  - «Что ты ДЕЛАЛ вчера?»

Изменение плана на итерацию

- By Feature, By Component
  - Нельзя менять план на итерацию
- By Role, By Competence, Full
  - Можно заменить еще не сделанные фичи

Принятие решений

- Персонально вне команды
  - Product Owner, организация, другие команды
- Коллективно командой
  - Совместно на планировании, ретроспективе, DSM и прочих митингах
- Персонально членом команды
  - Команда доверяет члену команды самостоятельно принять решение

Например, так

Снаружи	Vision Backlog	Архитектура	Учет времени Часы присутствия
Коллективно	Acceptance Tests	Дизайн	Роли Практики и регламенты Coding Styles
Персонально	Детали	Код	Инструменты

Коллективное принятие решений

- Если у команды НЕДОСТАТОЧНО информации для принятия правильного решения, то решение лучше принимать СНАРУЖИ
- Примеры
  - Vision, Backlog
  - Архитектура в большом проекте
  - Coding Styles

Инженерное и бизнес- принятие решений

Недостаток информации — Избыток информации

Характерно для…	**БИЗНЕСА**	**ИНЖЕНЕРИИ**
Mindset	Интуиция	Анализ и расчет
Приоритет	Скорость	Качество
Важность обратной связи	Очень высокая	Высокая
Эффективность	**Низкая**	**Высокая**

Что может помешать самоорганизации?

Главный враг самоорганизации

- Персональная ответственность за результат:
  - Зафиксированная сфера ответственности
  - Большой стек задач
  - Дифференцированное персональное поощрение за успехи

Некомандный игрок

Некомандное поведение

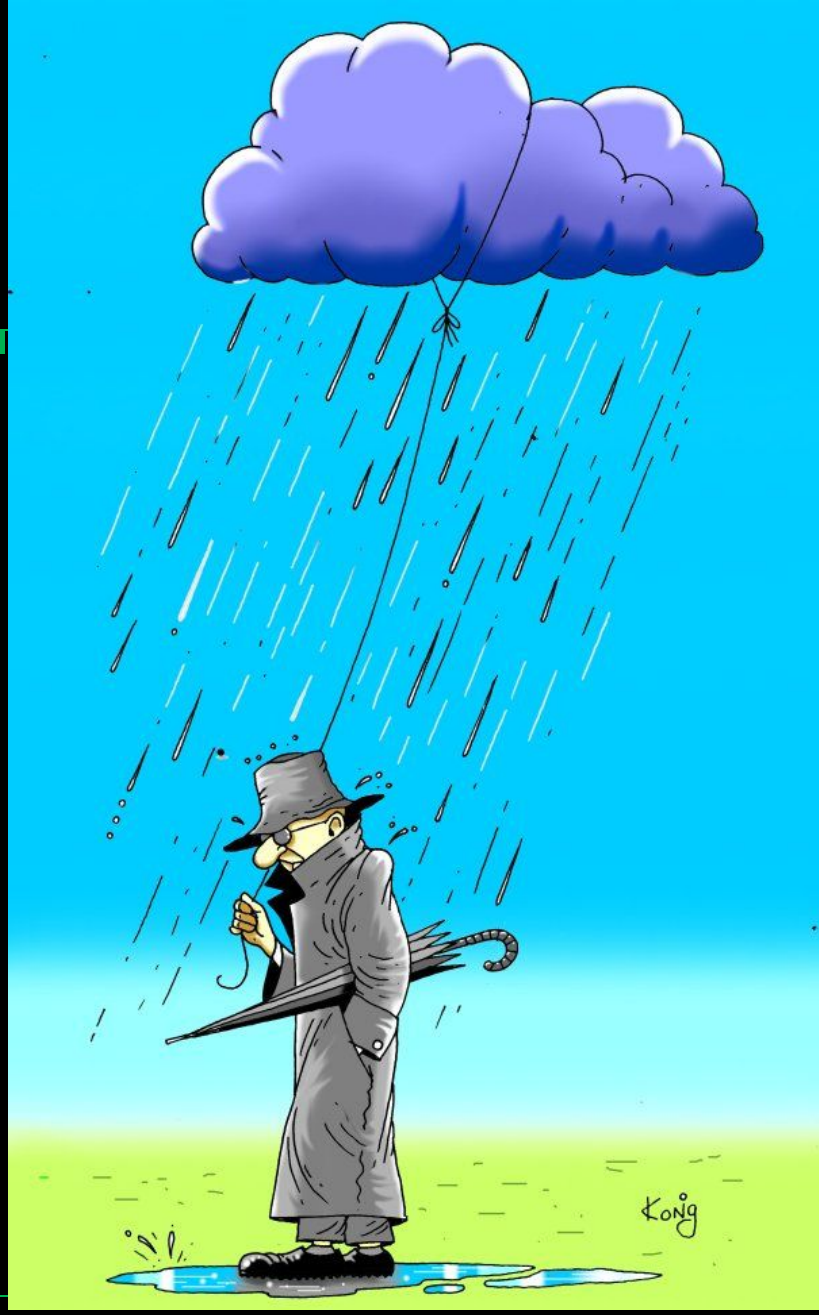
- Неспособность взять на себя ответственность вместе с командой
- Неспособность отвечать перед командой
- Несогласие с общей целью

Некомандное поведение

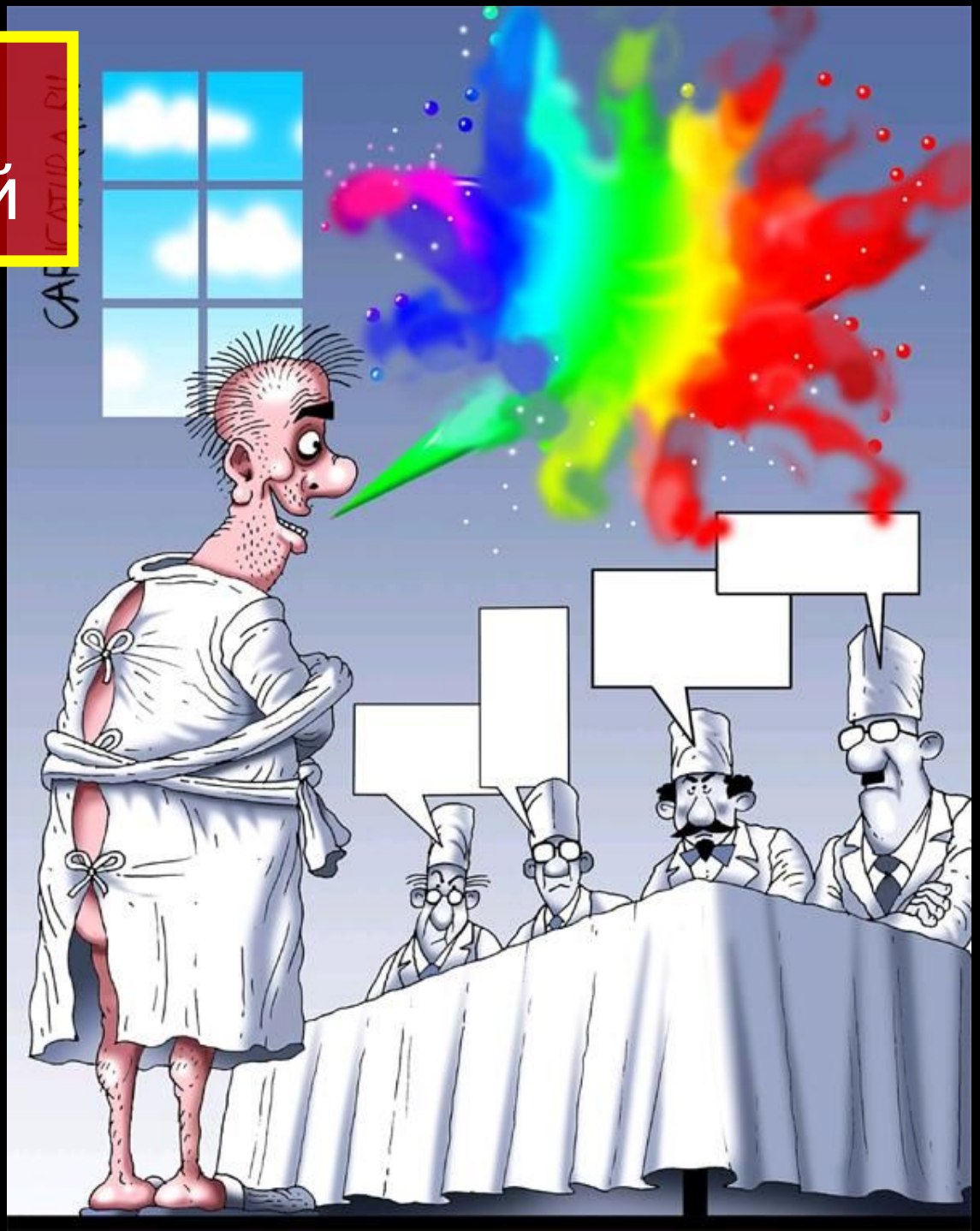
- Менеджер удаляет некомандного человека
- Команда **не принимает** таких решений
- Удаление некомандного человека все равно *немного* демотивирует команду

УГ

- Команда состоит из скучных и безинициативных товарищей



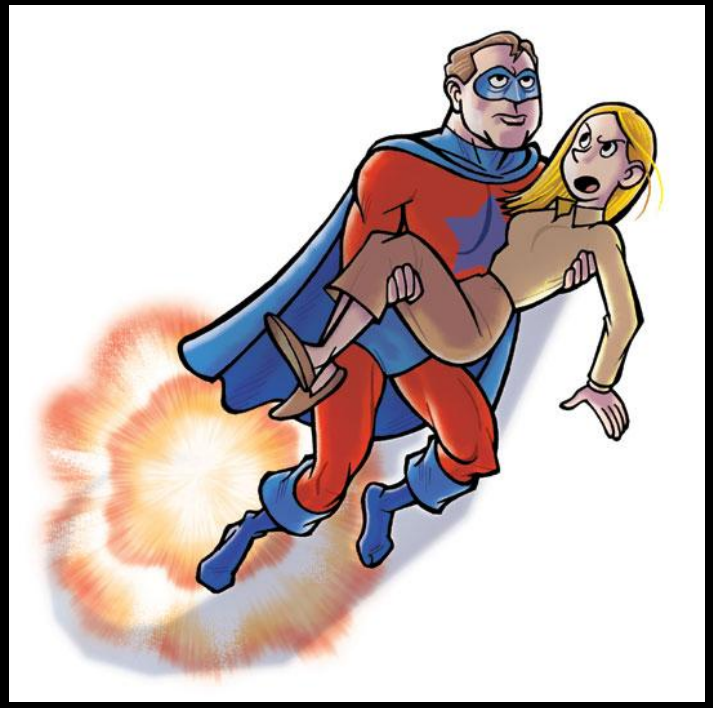
Добавляем ярких людей



Супермен

- Думает, что знает как надо делать
- ИЛИ
- Действительно знает как надо делать

- Команда делегирует ему принятие решений



Когда самоорганизация ОПАСНА?

Тролль

- Его цель отличается от вашей
- Он имеет большое влияние на команду
- Самоорганизация приведет к неразрешимому конфликту между вами и командой
- Его увольнение может привести к уходу всей команды

Некоторые менеджеры могут оказаться лишними

А что делать менеджерам?

- Управлять самоорганизацией
  - Область ответственности
  - Состав команды
  - Обмен разработчиками

Спасибо!

ВОПРОСЫ?

http://blog.scrumtrek.ru

Be ag;)e

Картиники: Игорь Конденко

http://caricatura.ru/parad/kondenko/