AGILE, SCRUM, LEAN

Айрат Хасьянов

Примерный план

Разогрев — мир, в котором мы живем

Agile — методология? Сообщество? Философия?

Scrum — играем в Rugby!

Lean — причем здесь автомобили?

МИР, В КОТОРОМ МЫ ЖИВЕМ

Авторитетное мнение

Программная инженерия

Методология

...Менеджеры программных разработок — и в этом они не отличаются от других менеджеров, — чтобы преуспеть, должны овладеть основами менеджмента: **работать с людьми** и проявлять **решительность**, решать политические проблемы и придерживаться **графика работ**, поставлять продукт **хорошего качества**.

Carl Selinger

Программная инженерия

Разработка ПО ~ постройка звездолетовЭтим должны заниматься инженеры!Жизненный цикл продукта любой инженерии:

1. **Проектирование**
2. **Создание образца**
3. **Тестирование**
4. **Производство**
5. **Сопровождение (эксплуатация)**

Большая разница!

Стоимость производства исчезающе мала!

Сложно: нет объективных законов механики!
Тестирование продукта — единственный выход!

80% программных проектов терпят крах!

Очень мало опыта!

Почему мало опыта?

- **14 февраля 1946 — ENIAC - Начало**
- **1965-1985 — Кризис**
- ПО становится ***дороже*** железа
- ***1000-и*** разработчиков на проект
- ***Ненадежно***, приводит к порче имущества
- Первые случаи ***смерти из-за ошибок в ПО***

Поиск универсального решения

1985-1989 — Поиск волшебной палочки

- Инструменты: ООП, CASE-средства
- Формальные методы из инженерных наук
- Понятие **Методологии**:
  - **Жизненный цикл**
  - **Программный процесс**

- **1986** — Fred Brooks «No Silver Bullet»: *не будет 10x продуктивности за 10 лет, может, лет через 40...*

Осознание

1990-е — Интернет захватил власть над миром

XXI век — Легковесные методологии

Agile, Scrum, Lean и др.

AGILE

1. Авторитетное мнение
2. Идеи
3. Принципы

Agile — Гибкая разработка

А мы при разработке ПО используем фирменную методолгоию **"Отжайл"** Она основана на принципе отжиманий от пола

Игорь Зиновьев

Это даже не методология!

1957 — IBM, *Инкрементальная разработка*

1974 — Edmonds, *Адаптивный процесс (размышление, сотрудничество, обучение)*

Февраль 2001 — Agile Manifetso

Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Stephen J. Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas

Ранние Agile методологии

1995 — Scrum (Sutherland, Schwaber)

1996 — Экстремальное программирование (Beck)

Основные идеи, значимость которых нивелирована

Личности и их взаимодействия важнее, чем процессы и инструменты

Работающее программное обеспечение важнее, чем полная документация;

Сотрудничество с заказчиком важнее, чем контрактные обязательства;

Реакция на изменения важнее, чем следование плану.

Agile принципы 1

- **Удовлетворение клиента** за счёт ранней и бесперебойной поставки ценного ПО;
- Приветствие **изменения требований**, даже в конце разработки;
- Частая поставка **рабочего ПО**;
- Тесное, ежедневное **общение заказчика с разработчиками** на протяжении всего проекта;

Agile принципы 2

- Проектом занимаются **мотивированные личности**, обеспеченные условиями, поддержкой и доверием;
- Передача информации путем **личного общения**;
- **Работающее ПО** — лучший измеритель прогресса;
- Возможность поддержания постоянного темпа на **неограниченное время**;

Agile принципы 3

Внимание на **улучшение мастерства** и **качественную разработку**;

Простота — искусство не делать лишней работы;

Работа над формированием **КОМАНДЫ**;

Постоянная адаптация к изменяющимся обстоятельствам.

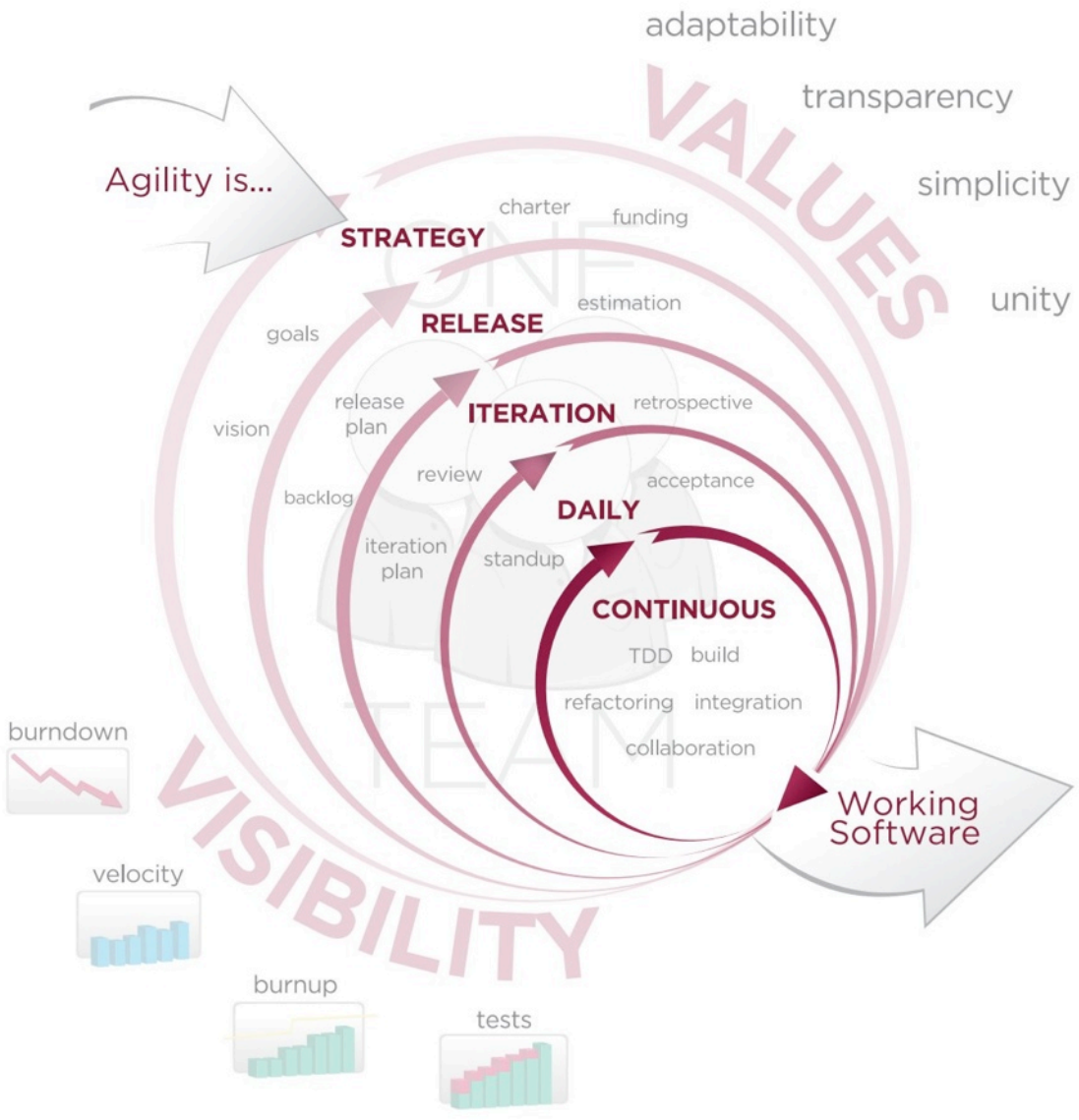
Советы

Адаптируйте Ваш процесс под Ваши условия;

Используйте **подходящий инструмент** (trac, redmine, teamer, pivotal tracker, Oracle и пр.);

Создавайте наилучшие программы с **наименьшим необходимым числом функций**;

AGILE DEVELOPMENT



ACCELERATE DELIVERY

SCRUM

1. Авторитетное мнение
2. Описание процесса
3. Принципы

Scrum — Agile методология

Наши манагеры используют гибкую методологию.
Называется «Scream»

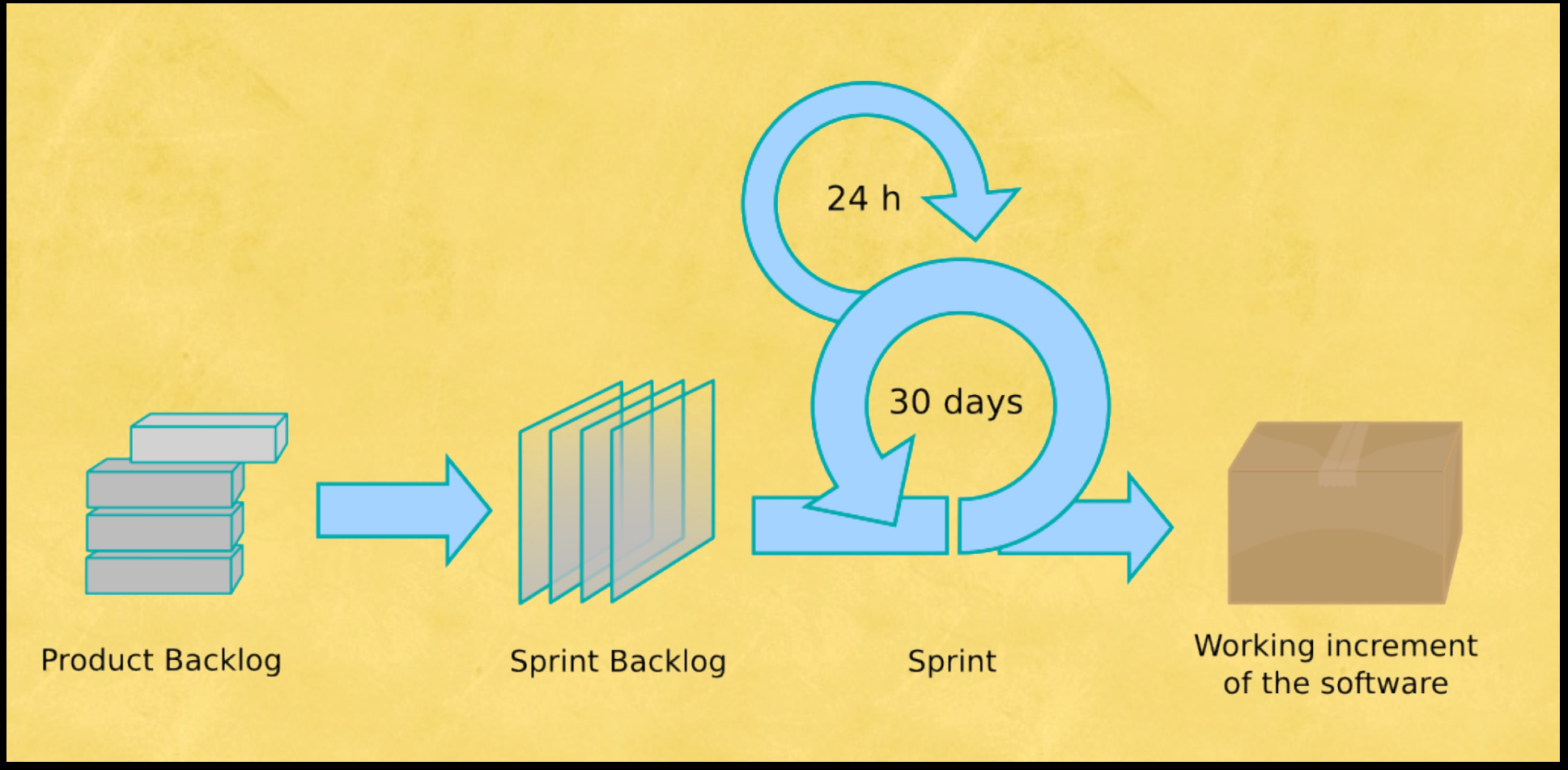
История

1986 — Hirotaka Takeuchi и Ikujiro Nonaka сравнили разработку ПО и **рэгби**

1991 — DeGrace и Stahl "Wicked Problems, Righteous Solutions" (scrum)

1995 — Sutherland и Schwaber - статья на воркшопе конференции OOPSLA'95

Scrum — описание процесса



Артефакты Scrum

- **Sprint** — инкремент, 1-4 недельный этап поставки
- **Product Backlog** — ранжированный по приоритетам список требований
- **Sprint Backlog** — требования попавшие в Sprint;
- **Daily Scrum** — командная сверка статуса проекта;
- **Burn down** — график оставшейся работы в Sprint Backlog.

Роли в Scrum

Scrum Team — Разработчики, Product Owner и Scrum Master

Scrum Master — менеджер проекта

Product Owner — спонсор, представляет заинтересованных лиц проекта;

Daily Scrum — командная сверка статуса проекта;

Team — Scrum Team + остальные участники работ над проектом.

Sprint Planning Meeting

- Каждые **7-30 дней**
- **Что делать** в Sprint'е
- Подготовка **Sprint Backlog** с временем работ, участвует вся команда
- **8-часовой лимит:**
  - 4 часа на определение приоритетов
  - 4 часа набросок плана для Sprint'а => Sprint Backlog;

Sprint Review Meeting

- В конце **каждого Sprint'a**
- Обзор завершенных и незавершенных работ
- **the Demo** для заинтересованных лиц
- Незавершенная работа не демонстрируется
- **4-часовой лимит**

Sprint Retrospective

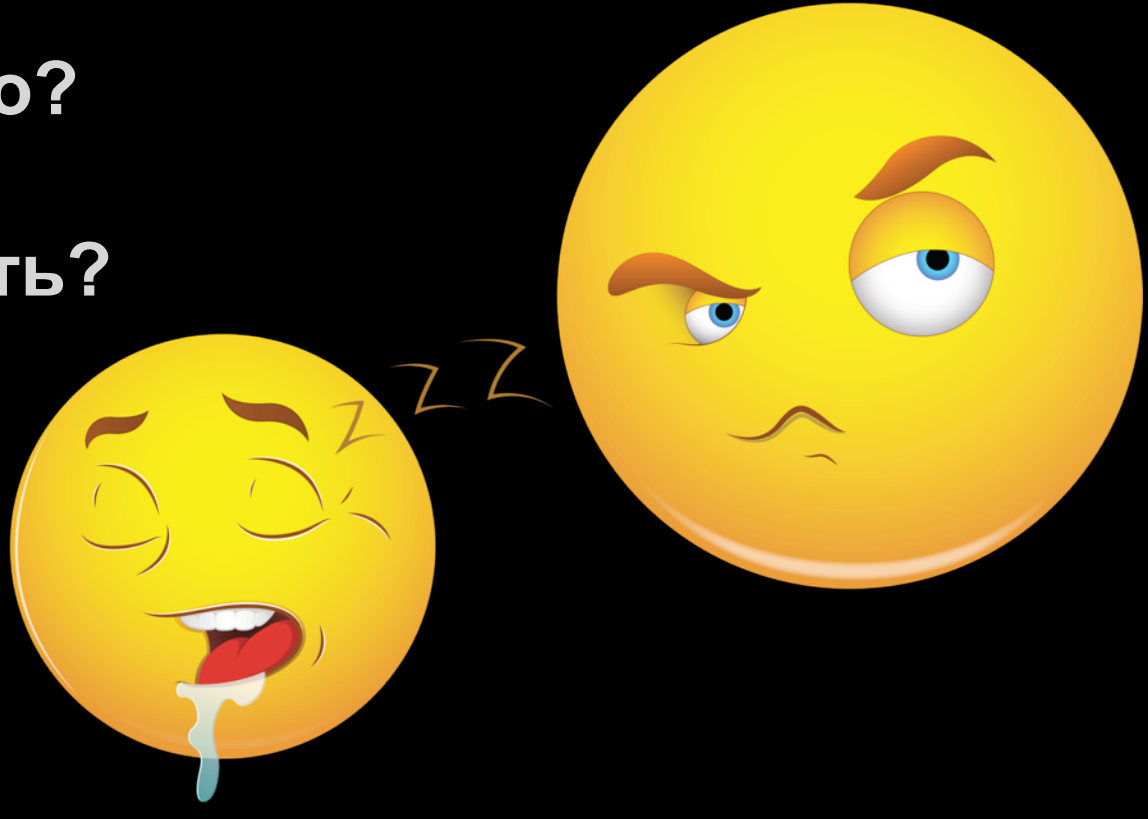
Обсуждение прошедшего Sprint'a

Постоянные усовершенствования процесса

Что было **правильно?**

Что следует **улучшить?**

3-часовой лимит



Другие артефакты

Impediment — препятствие для члена команды

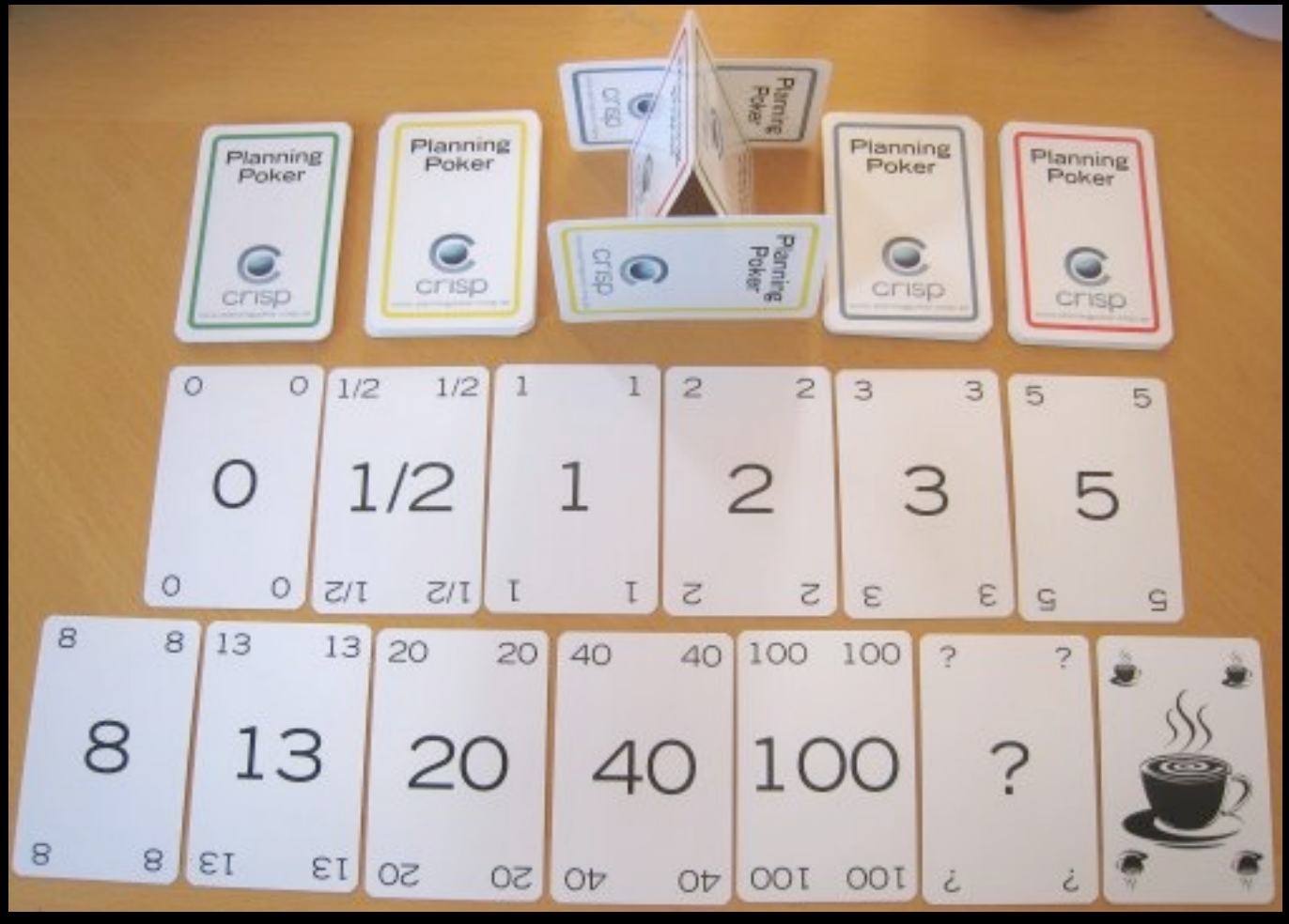
Sashimi — отчет, который иногда делают

Planning Poker — Sprint planning meeting *как покер*

Point Scale — система оценки сложности (s, m, L, XL)

Definition of Done — условие завершения проекта

Planning Poker



LEAN — БЕРЕЖЛИВАЯ РАЗРАБОТКА

1. Авторитетное мнение
2. История
3. Принципы

Бережливое производство

Береженного Бог бережет!

История

- **Япония, 1945** Проблемы крупносерийного производства
- **Toyota — Production System,** Тайити Оно
- Mary Poppendieck, Tom Poppendieck — 22 принципа

Исключение лишних затрат

- Разработка **ненужного** функционала;
- **Простои** при разработке;
- “**Излишние запасы**” в требованиях;
- **Бюрократизация** процесса;
- **Неэффективное** внутреннее взаимодействие.

Делать правильно с первого раза

- Короткие циклы разработки: цикл Деминга (**Plan, Do, Check, Act**)
- **Раннее тестирование**;
- **Полное покрытие** кода модульными тестами;
- **Непрерывная** интеграция и постоянный контакт с заказчиком;
- **Рефакторинг -** постоянное улучшение кода;
- Культура постоянного **совершенствования**.

Делегирование полномочий и поддержка тех, кто добавляет ценность

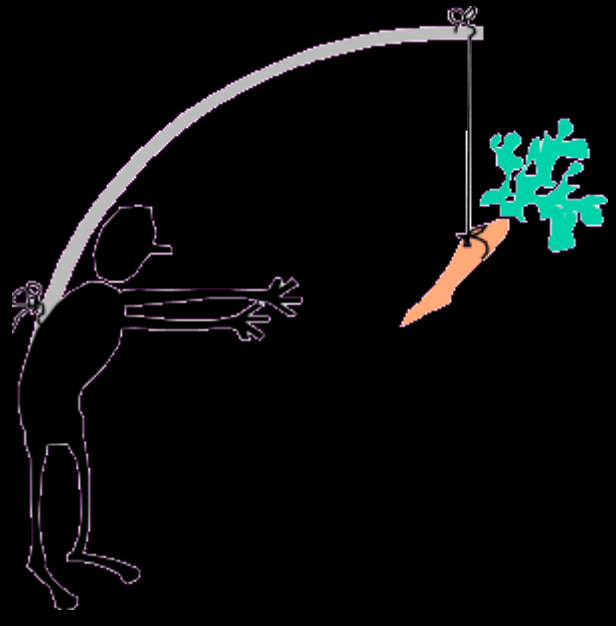
- **Делегирование** принятия решений на по возможности нижний уровень орг. структуры;
- Если проект встречается с трудностями, разработчикам предоставляются **инструменты для оценки и улучшения своей работы**;
- **Процесс** не фиксируется жестко и **подлежит изменениям** в соответствии с параметрами развития проекта;
- Люди, выполняющие работу, должны **САМИ** определять детали выполнения работы!

Мотивация команды

Проект - больше, чем список заданий. **Работа должна быть интересной!**

Каждый **проект учит** участников чему-то новому!

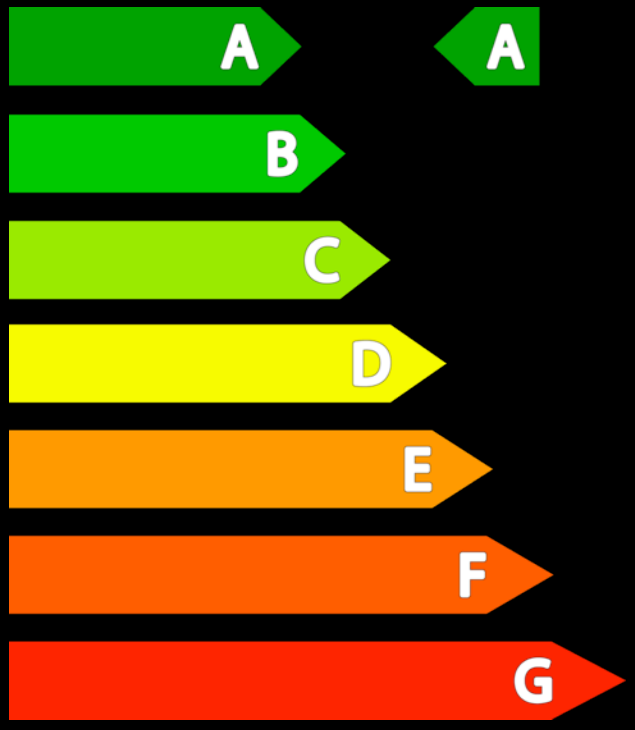
Если команда не вовлечена в проект и не жаждет его успешного завершения, мы **отказываемся от проекта**!



Разделение всеми разработчиками принципов бережливого производства

- Мыслить широко;
- действовать мало;
- промахиваться быстро;
- учиться стремительно.

Повторное использование компонентов



- Использовать **готовое решение**, если оно есть при разработке;
- Писать артефакты проекта так, чтобы их можно было использовать в **другом проекте**;
- По возможности использовать артефакты, **созданные ранее** в новых проектах.

Запрет локальной оптимизации

- Локальная оптимизация часто **приводит к ухудшению** глобальной оптимизации;
- **Лучшее - враг хорошего**: прежде чем тратить время на оптимизацию, убедитесь, что оптимизируемый фрагмент кода действительно является узким местом.
- **Принцип Кента Бека**: “Заставьте программу работать; заставьте ее работать верно; затем, сделайте ее быстрой!”.

ПОЛЕЗНЫЕ ССЫЛКИ

http://agilemanifesto.org/

http://blog.htc-cs.ru/post/lean_programming.aspx

http://www.scrum.org/

http://www.scrum.org/

http://members.cox.net/risingl1/Articles/IEEEScrum.pdf

Frederick P. Brooks, The Mythical Man-Month: Essays on Software Engineering

Спасибо за внимание!

Айрат Хасьянов