Software development processes in real life

И как всё это касается меня — разработчика?

- Вам нравится разрабатывать фичи, которые никто не будет использовать?
- Вам нравится работать по проектному плану, в который вы не верили с самого начала?
- Вам нравится кодировать архитектуру, которую кто-то продумал до вас, и в которой вы видите множество недостатков?
- Вам нравится работать в группе людей, где вы чувствуете себя одиночкой?

Now updated! The know-how you need to get the job done on time and within budget

Project Management FOR DUMMIES®

2nd Edition

Build and lead a winning project team

A Reference for the Rest of Us!®

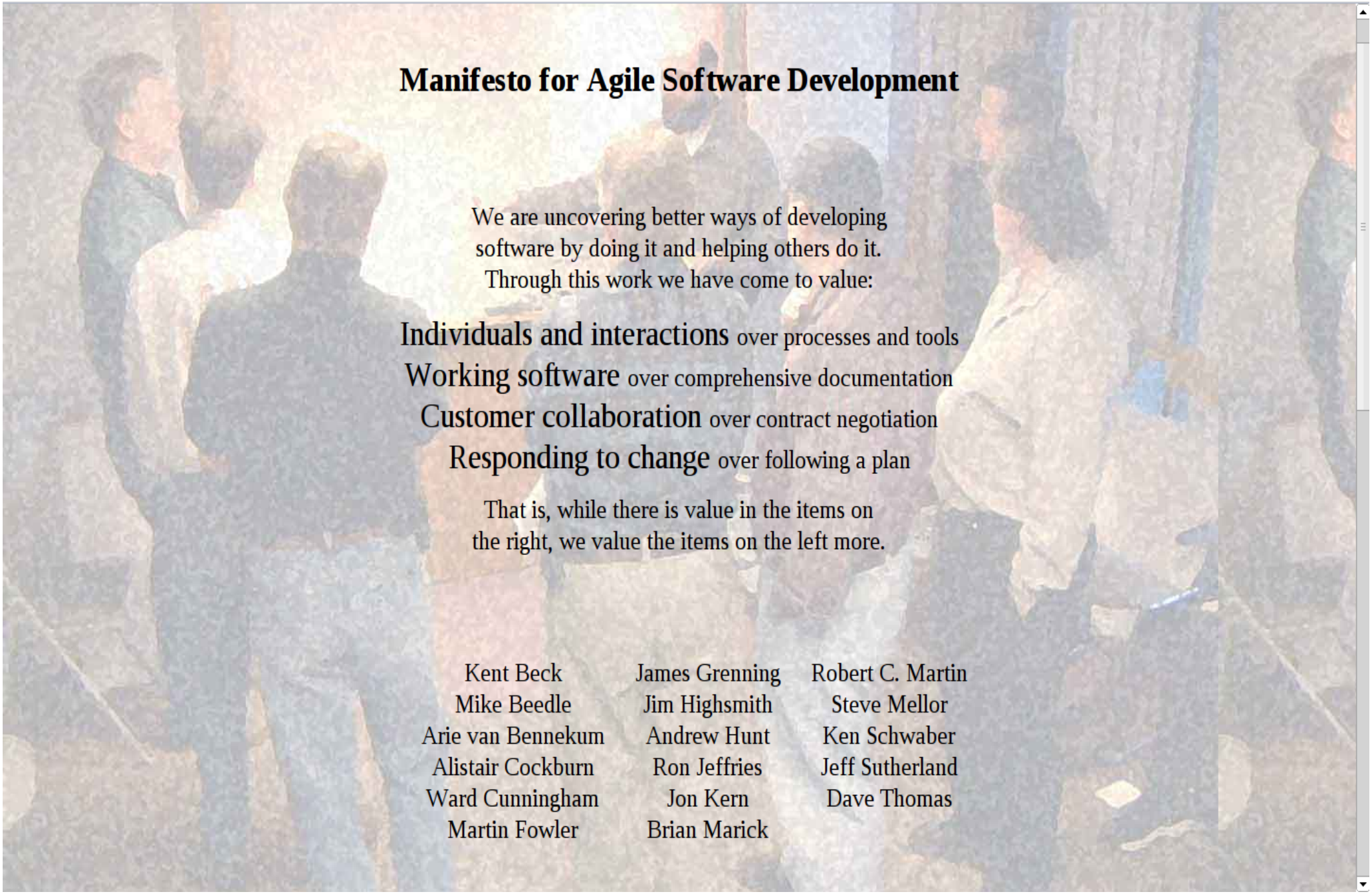
FREE eTips at dummies.com®

Stanley E. Portny
Project management consultant and certified Project Management Professional (PMP)

Agile software development

Гибкая методология разработки ПО - набор ценностей и принципов

11-13 февраля 2001 г.



Manifesto for Agile Software Development

We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on
the right, we value the items on the left more.

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

Разъяснение принципов Agile

- Удовлетворение клиента за счёт ранней и бесперебойной поставки ценного ПО;
- приветствие изменения требований, даже в конце разработки (это может повысить конкурентоспособность полученного продукта);
- частая поставка рабочего ПО (каждый месяц или неделю или ещё чаще);
- тесное, ежедневное общение заказчика с разработчиками на протяжении всего проекта;

Разъяснение принципов Agile

- проектом занимаются мотивированные личности, которые обеспечены нужными условиями работы, поддержкой и доверием;
- рекомендуемый метод передачи информации — личный разговор (лицом к лицу);
- работающее ПО — лучший измеритель прогресса;
- спонсоры, разработчики и пользователи должны иметь возможность поддерживать постоянный темп на неопределенный срок;

Разъяснение принципов Agile

- постоянное внимание на улучшение технического мастерства и удобный дизайн;
- простота — искусство НЕ делать лишней работы;
- лучшие технические требования, дизайн и архитектура получаются у самоорганизованной команды;
- постоянная адаптация к изменяющимся обстоятельствам.

Agile modeling

- Моделируйте чтобы понять
- Моделируйте чтобы общаться
- Чаще используйте простые средства для моделирования - доска, стикеры, несколько основных видов UML диаграмм.
- Избавляйтесь от устаревших моделей, если они уже выполнили свою роль.
- Подтверждайте модели кодом.
- Код - единственный артефакт, которому можно верить.

Scrum

Agile методология управления проектами

Scrum

Product owner - уполномоченный принимать решения о том, какую фичу стоит разрабатывать раньше, какую позже.

Обычно работает на стороне заказчика или является бизнес аналитиком.

Разработчикам проще, если такой человек есть, и он один на проект (во избежание конфликтов).

Scrum

Product Owner поддерживает список требований - пожеланий к продукту. Этот список он сортирует (приоритезирует) по принципу «ценное сверху, менее ценное снизу».

Такой список называется «**Product Backlog**».

Scrum

Собираясь на планирование итерации или «спринта», члены команды выбирают из Product Backlog ту верхнюю часть, которую, как они считают, реально закончить за итерацию.

Product Owner принимает активное участие в подобном планировании.

Т.о. заказчик ожидает получить некий ценнейший набор функционала за довольно короткое время, команда получает план, в который верит.

Scrum

По истечению итерации команда и заказчик встречаются для просмотра и обсуждения результатов. Это называется **демонстрацией**.

На выходе такого митинга имеются:

- уменьшенные риски, за счёт доказательства работоспособности команды, технологий и требований;
- новые идеи по развитию продукта;
- доверие заказчика и команды к друг другу.

Scrum

После демонстрации команда собирается и обсуждает серию экспериментов и задач, которые как она считает помогут более эффективно работать в течение следующей итерации (спринта).

Этот процесс называется **ретроспектива**.

Scrum

Во время спринта членам команды скорее всего придётся синхронизировать свои усилия и помогать друг другу для более слаженной работы и достижения общего результата. Говорят, что хорошо это делать раз в день в течение 10−15 минут в присутствии всех членов команды. Это и называется **Daily Scrum** или **митинг**.

Scrum

В течение митинга каждый член команды отвечает на 3 вопроса.

- Что сделано с момента предыдущего митинга до текущего?
- Что будет сделано с момента текущего митинга до следующего?
- Какие проблемы мешают достижению целей спринта? Над решением этих проблем работает **Scrum Master**.

Scrum - не непрерывное совещание

Работающее программное обеспечение и реальный код важнее документации и диаграмм

Extream programming

Agile методология разработки ПО

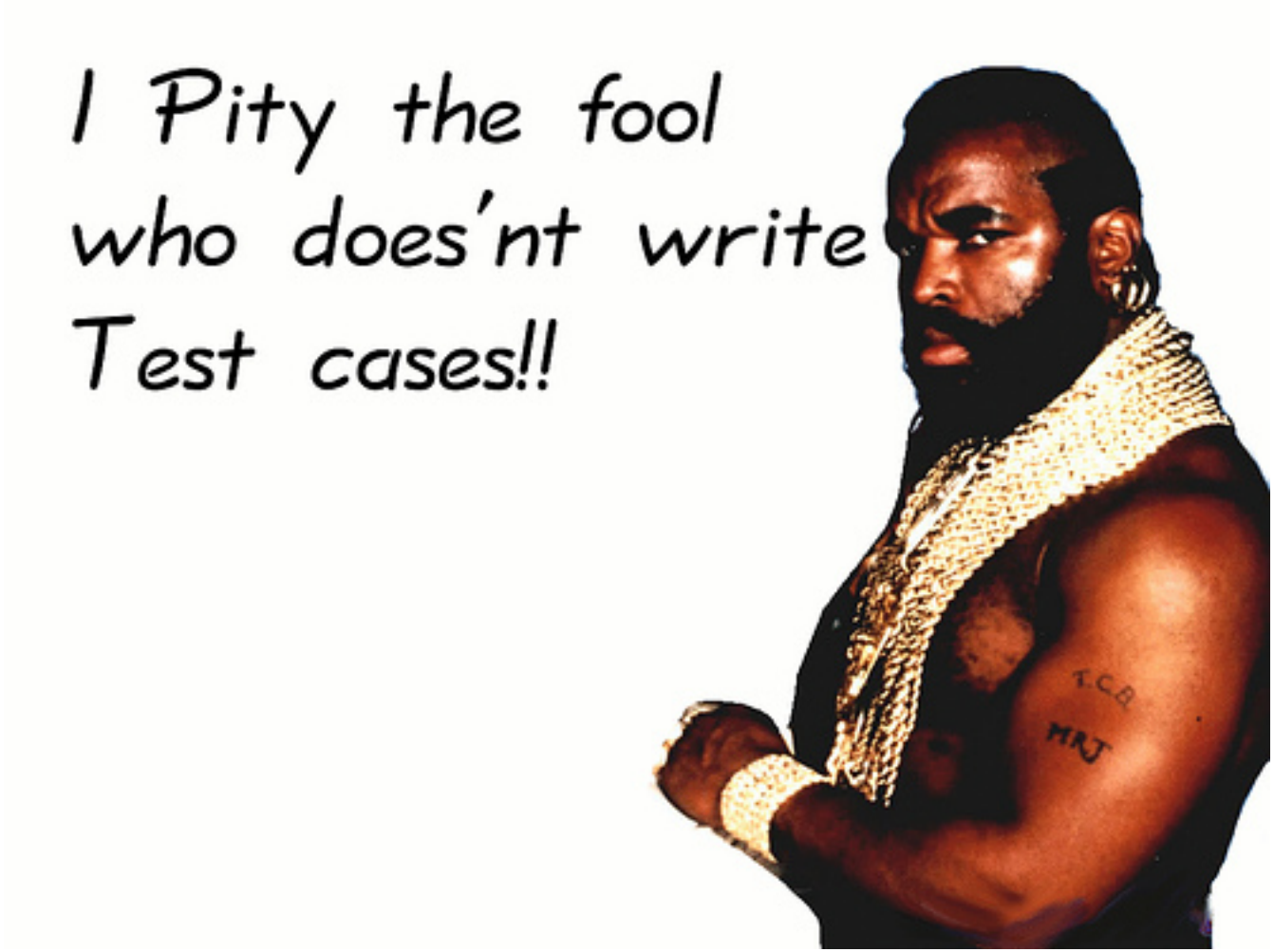
Кент Бек, Уорд Каннингем, Мартин Фаулер

12 основных принципов XP

Короткий цикл обратной связи (Fine scale feedback)

- Тестирование, TDD
- Игра в планирование (Planning game)
- Заказчик всегда рядом (Whole team, Onsite customer)
- Парное программирование (Pair programming)

Мне жаль тех, кто не пишет тесты!!



Непрерывный, а не пакетный процесс

- Непрерывная интеграция (Continuous Integration)
- Рефакторинг (Design Improvement, Refactor)
- Частые небольшие релизы (Small Releases)

Понимание, разделяемое всеми

- Простота (Simple design)
- Метафора системы (System metaphor)
- Коллективное владение кодом (Collective code ownership) или выбранными шаблонами проектирования (Collective patterns ownership)
- Стандарт кодирования (Coding standard or Coding conventions)

Социальная защищенность программиста (Programmer welfare)

Стабильный рабочий ритм (Sustainable pace),
40-часовая рабочая неделя (Forty hour week)

Microsoft Solutions Framework (MSF)

Методология разработки ПО от Microsoft, 1994 г

Обобщенная MSF содержит:

- Модели:
  - Проектной группы
  - Процессов
- Дисциплины:
  - Управление проектами
  - Управление рисками
  - Управление подготовкой

Т.о. поддерживается весь жизненный цикл ПО.

MSF Team Model

- Определяет ролевые кластеры, их области компетенции и зоны ответственности.
- Возникла в результате осмысления недостатков иерархической структуры традиционных проектных групп.
- Небольшие многопрофильные команды.
- Члены распределяют между собой ответственность и дополняют области компетенций друг друга.
- Единое видение проекта, высокие требования к качеству работы и желание самосовершенствоваться.

MSF Team Model

- Ответственность за управление проектом распределенна между лидерами ролевых кластеров внутри команды.
- Каждый член имеет необходимые полномочия для выполнения своих обязанностей и уверен, что получит от коллег все необходимое.
- Разбиение больших команд (более 10 человек) на малые многопрофильные группы направлений (feature teams).

MSF process model

- Представляет общую методологию разработки и внедрения IT решений.
- Гибкости и отсутствие жестко навязанных процедур.
- Гибрид каскадной (waterfall) и спиральной (spiral) методологий разработки.

MSF process model

- Артефакты создаются, как правило, итеративными методами.
- MSF рекомендует начинать разработку решения с построения, тестирования и внедрения его базовой функциональности.
- Затем к решению добавляются все новые и новые возможности.

Risk management

- Одна из ключевых дисциплин MSF.
- Изменения - неотъемлемая часть жизненного цикла ПО.
- Отстаивает превентивный подход к работе с рисками в условиях неопределенности.
- Непрерывное оценивание рисков на протяжении всего жизненного цикла проекта.
- Девиз MSF — мы не боремся с рисками — мы ими управляем.

Управление подготовкой

- Посвящена управлению знаниями, профессиональными умениями и способностями, необходимыми для создания успешных решений.
- Описывает фундаментальные принципы MSF и дает рекомендации по применению превентивного подхода к управлению знаниями на протяжении всего жизненного цикла информационных технологий.
- Рассматривает планирование процесса управления подготовкой.

Visual Studio Team System

- Team Foundation Server
- Team Test Load Agent
- Интеграция с Visual Studio
- Интеграция с Microsoft Office

Quick tour to software development philosophies

by

http://en.wikipedia.org/wiki/List_of_software_development_philosophies

A-B

- Agent-oriented programming
- Agile software development

Agile Unified Process (AUP)

- Aspect-oriented Programming
- Behavior Driven Development (BDD)
- Big Design Up Front (BDUF)
- Blind Men And Elephant Approach (BMAEA)
- Brooks's law

C

- Cathedral and the Bazaar
- Code and fix
- Cone of Uncertainty
- Constructionist design methodology (CDM)
- Continuous integration
- Control tables
- Conway's Law
- Cowboy coding
- Crystal Clear

D-F

- Dependency injection
- Design-driven development (D3)
- Design Driven Testing (DDT)
- Domain-Driven Design (DDD)
- Don't Make Me Think
- Don't repeat yourself (DRY)
- Dynamic Systems Development Method (DSDM)
- Evolutionary prototyping
- Extreme Programming (XP)
- Feature Driven Development

G-L

- Good Enough For Now (GEFN)
- Hollywood Principle
- Inversion of control
- Iterative and incremental development
- Joint application design
- Kaizen
- Kanban
- KISS principle
- Lean software development
- Literate Programming

M-R

- Microsoft Solutions Framework (MSF)
- Model-driven architecture (MDA)
- MoSCoW Method
- Open source
- Open Unified Process
- Parkinson's Law
- Quick-and-dirty
- Rapid application development (RAD)
- Rational Unified Process (RUP)

Release early, release often (see also The Cathedral and the Bazaar)

- Responsibility-driven design (RDD)

S-T

- Scrum
- Separation of concerns (SoC)
- Service-oriented modeling
- Software Craftsmanship
- Software System Safety
- Solid (object-oriented design)
- Spiral model
- Structured Systems Analysis and Design Method (SSADM)
- Team Software Process (TSP)
- Test-driven development (TDD)
- Two Tracks Unified Process (2TUP)

U-Z

- Unified Process (UP)
- Unix philosophy
- V-Model
- Waterfall model
- Wheel and spoke model
- When it's ready
- Win-Win Model
- Worse is better
- You Ain't Gonna Need It (YAGNI)

Спасибо!