

Мыслить проектно: история и современность

Максим Цепков

Главный архитектор решений

Software Engineering Conference Russia
Москва, 12 октября 2018

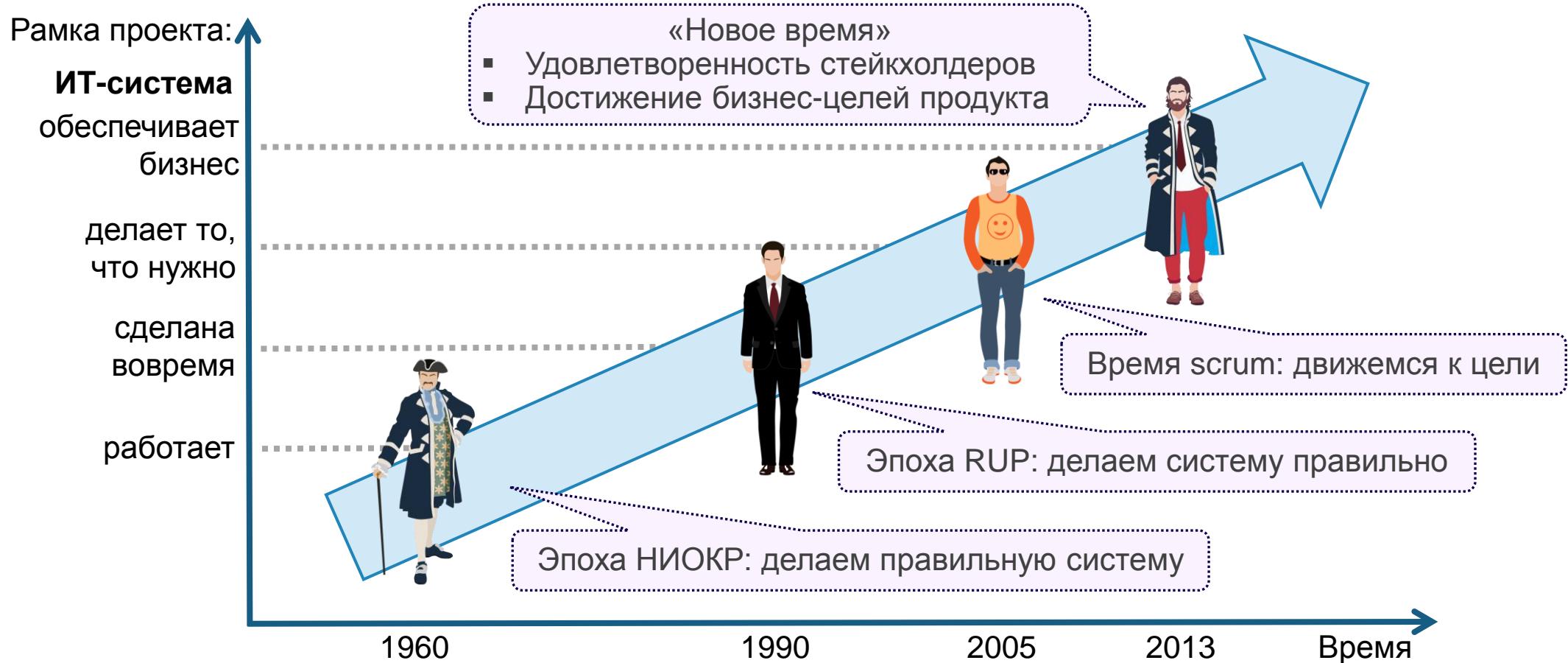
История культур программных проектов

Энтони Лаудер «Культуры программных проектов» (2008)

- ➔ История ИТ-отрасли делится на этапы
- ➔ Для каждого этапа характерен свой подход к ведению ИТ-проектов: представления об успехе, критерии качества и организации работ
- ➔ Выделяются четыре культуры
 - Научная
 - Заводская
 - Дизайнерская
 - Сервисная
- ➔ Каждая культура породила свои учебники, они основаны на представлениях того времени и согласованы между собой

[Оригинал](#), [перевод \(pdf\)](#),
[рецензия Стаса Фомина](#)

Смена культур ИТ-проектов



Моя схема отличается от схемы Энтони Лаудера. Если схема Лаудераозвучна больше – используйте ее, только доведите до настоящего времени

Пример: шаблоны документов в разных культурах

Задача: сделать механизм шаблонов для ввода типовых документов



Гибкая система шаблонов, с помощью которой можно настроить шаблон для любого документа



Длительная проработка задачи, определение критериев эффективности и описание «идеальной системы»



Быстрая серия прототипов и решений с последовательным усложнением шаблонов



Создание целевых групп для первых версий и работа именно с их кейсами. Может появиться несколько альтернативных механизмов, например, образцы документов вместо шаблонов

Настройка шаблонов окажется сложной даже для опытных сотрудников. Про удобный поиск и права доступа забудут

Шаблоны окажутся жесткими и пригодными только для узкого класса ситуаций, а расширить функционал будет сложно

Первые версии окажутся совсем не адекватными, на демо нужны конечные пользователи и фасилитация. Со сложными и редкими кейсами могут быть проблемы

НИОКР, RUP и agile

Эпоха НИОКР: когда компьютеры были большими

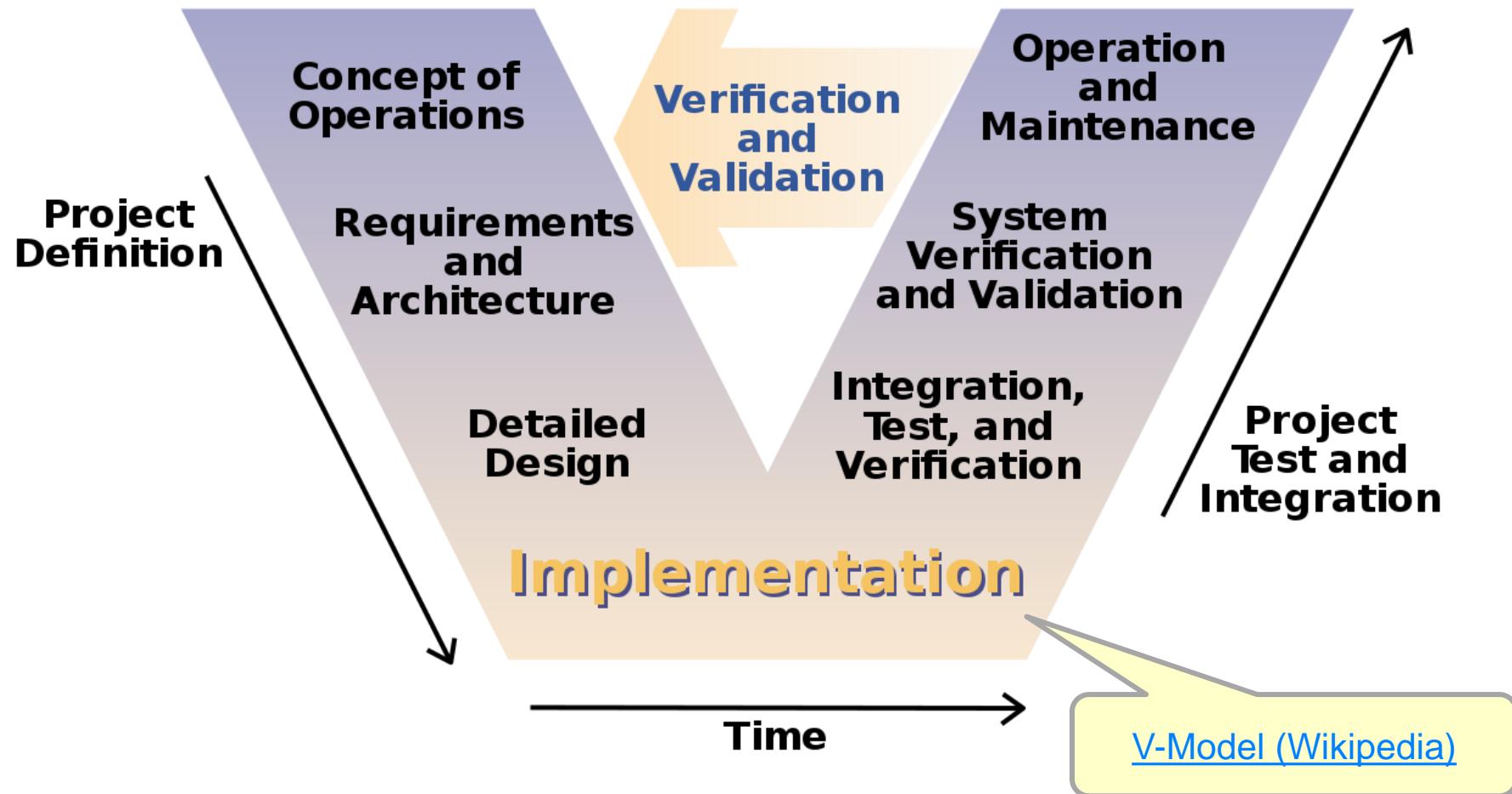
- ➡ Создавались большие и сложные системы
- ➡ Требования к системе редко менялись
- ➡ Проекты делал квалифицированный персонал
- ➡ Упор был на качество ИТ-системы

Ф. Брукс
«Мифический человеко-месяц»



Цель проекта – создать **совершенную ИТ-систему в одном экземпляре**

Представление о проекте – V-модель



Инженерная культура сейчас

- ➡ Построение ядра системы и совершенных фреймворков
- ➡ Фокус на стройности и архитектурном совершенстве
- ➡ Стремление поддержать все сложные кейсы
- ➡ Неприятие особых случаев, исключений и временных решений
- ➡ Слабая забота о тех, кто не будет работать со сложными решениями, даже когда они в большинстве



Пример из истории компании: объектно-учетное ядро на Oracle (1998, 2003) до сих пор служит основой систем и развивается

Эпоха RUP: массовая потребность в проектах потребовала много разработчиков

- ➡ Применим к ИТ-разработке принципы промышленного производства
- ➡ Разделим задачу на этапы: проектирование, разработка, внедрение
- ➡ Наладим процессы и разделим зоны ответственности

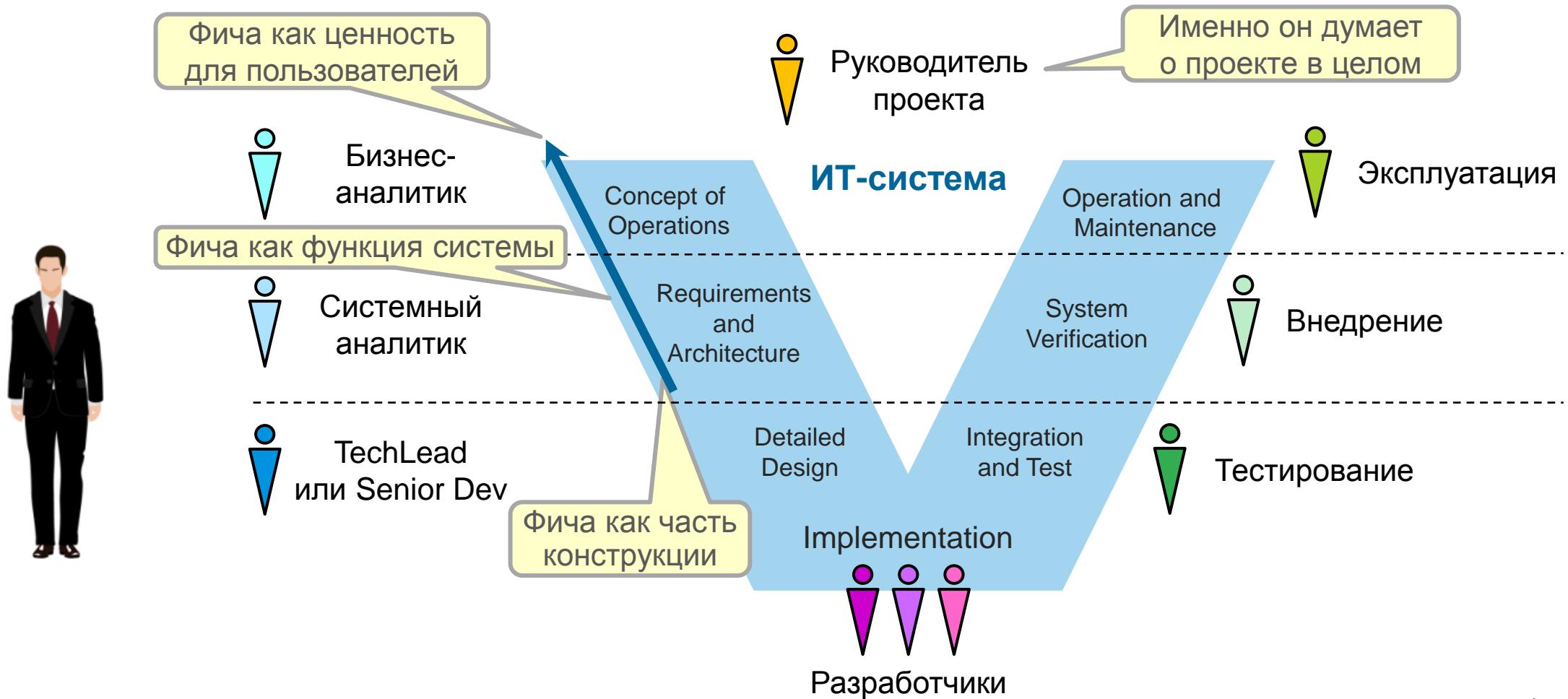
PMBOK 3 (2004)
RUP (2003)



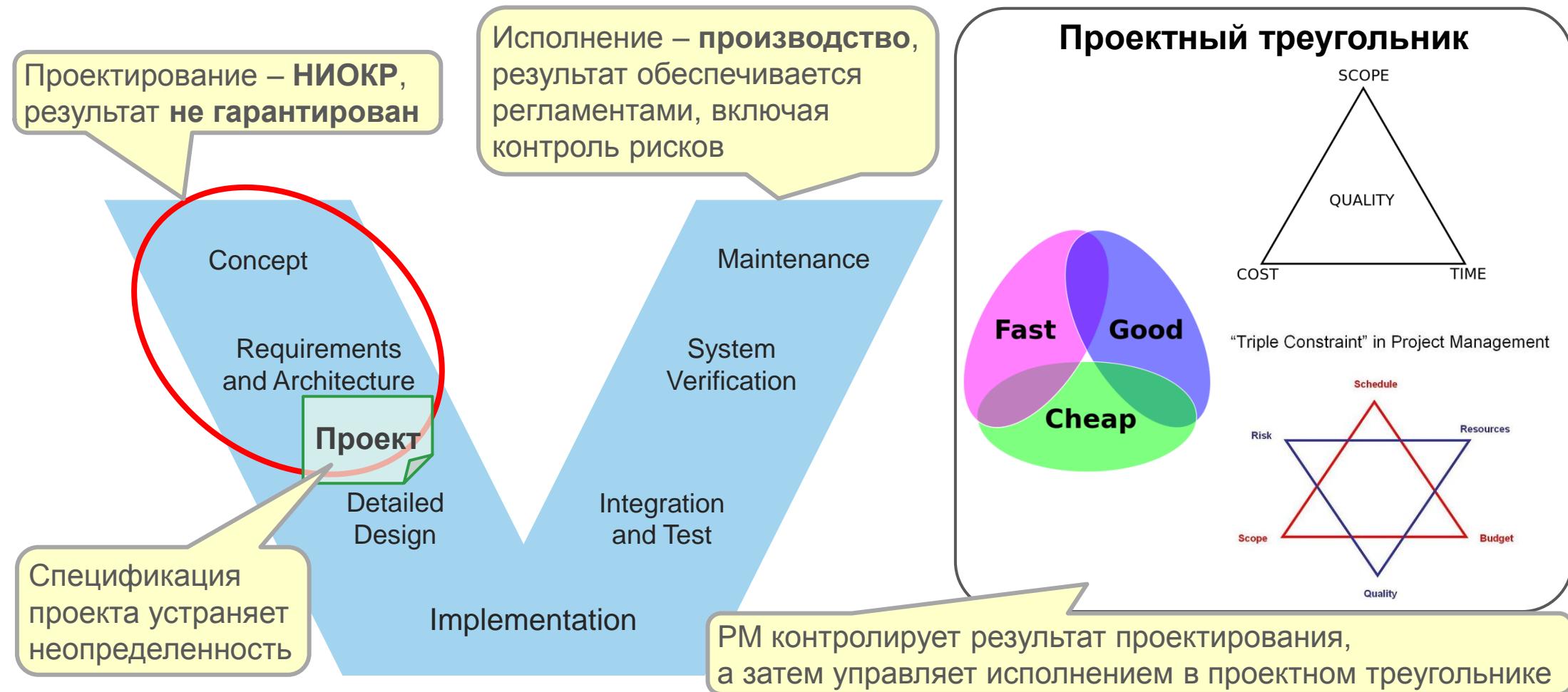
Оценка качества: удалось ли выполнить проект в срок, уложиться в бюджет и достичь ожидаемых результатов

Получалось не очень

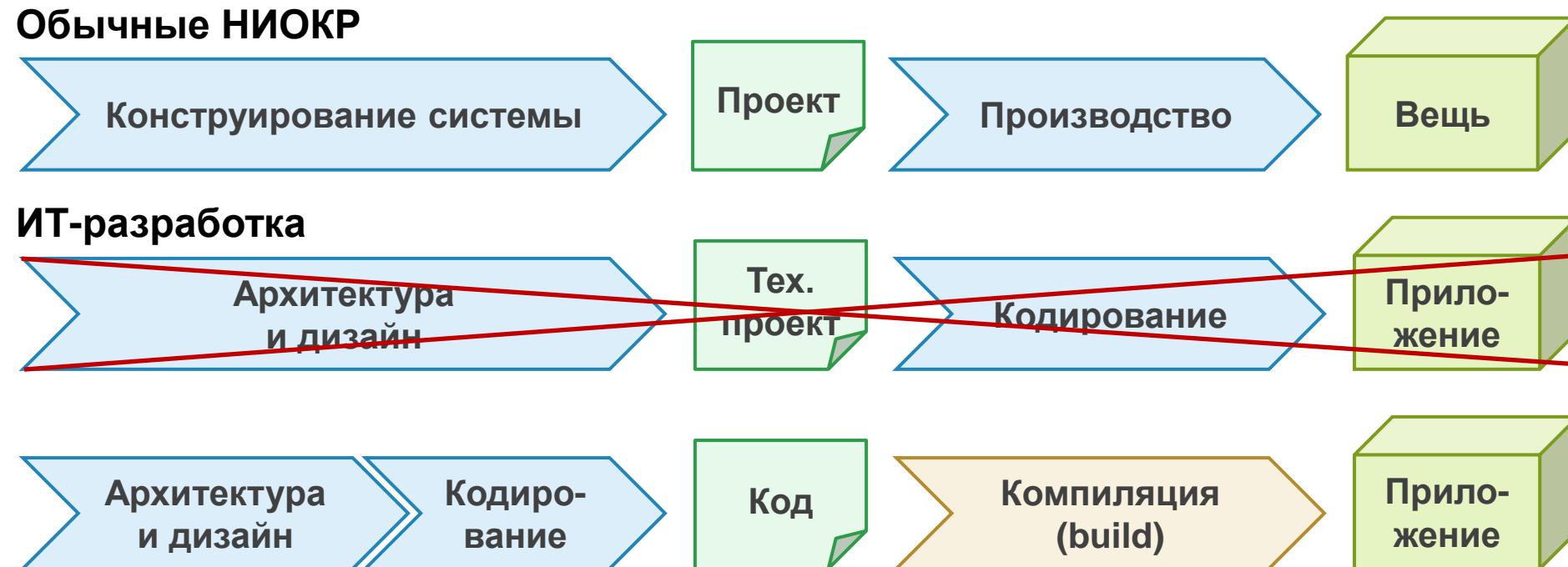
Специализации в проекте



Неопределенность – в проектирование!



Разработка кода – часть проектирования



[Jack W. Reeves. What is software design, 1992 \(перевод\)](#)

Неудача RUP

- ➔ Успех проекта определяют люди – это было осознано еще в 1980-х и обосновано Томом ДеМарко в книге «Человеческий фактор» (1987)
 - Критика регулярного управления (книга Ф. Брукса «Мифический человеко-месяц», 1975)
 - Разработка софта – НИОКР, а не производство ([статья Jack W. Reeves](#), 1992)
 - Решения рядового разработчика влияют на успех всего проекта
 - Производительность разработчика в разных условиях отличается на порядок
- ➔ Этому не поверили, и в конце 1990-х был поставлен эксперимент по нормированию процессов – RUP. Он окончился неудачей
 - Стоимость выросла многократно без гарантий успеха
 - Обеспечить реакцию на изменения требований не получилось



Но эксперименты продолжаются, культура живет

Вызовы, на которые не ответили в RUP

- ◆ **Стоимость:** процедура увеличивает ее кратно, не сильно повышая вероятность успеха
- ◆ **Изменчивость:** потребности меняются быстрее, чем проходит цикл разработки, и нужно учесть эти изменения
- ◆ **Управленческие кадры:** где их брать, особенно руководителей групп?
- ◆ **Нормирование аналитической работы:** попробовали в РМВОК 4 – не получилось

В стандарте
признано

Итерации в RUP –
тяжелые

Появление персоналок кратно усилило вызовы

- ➡ Возникла возможность автоматизировать бизнес компаний – но бизнес-процессы за время проекта успевают измениться
- ➡ Резкий дефицит квалифицированных кадров
- ➡ Конкуренция компаний за специалистов, а не разработчиков – за рабочие места
- ➡ Профессиональная самореализация – одна из главных ценностей разработчика, как совместить ее с коллективным результатом?

Agile и scrum: ответ на вызовы

- ➡ Планирование не работает – наблюдаем за траекторией движения проекта и приближением к цели
- ➡ Коллективное преодоление неопределенностей: все члены команды думают о движении проекта
- ➡ Концепция SMART-целей, измеримость достижения
- ➡ Требования изменяются вместе с целью

Гибкость
и наблюдаемость



Качественный проект – это частые
инкрементальные поставки **нужного** софта

User story – мышление за пользователя

Как <роль> я хочу <сделать что-то> для того чтобы <достичь целей>

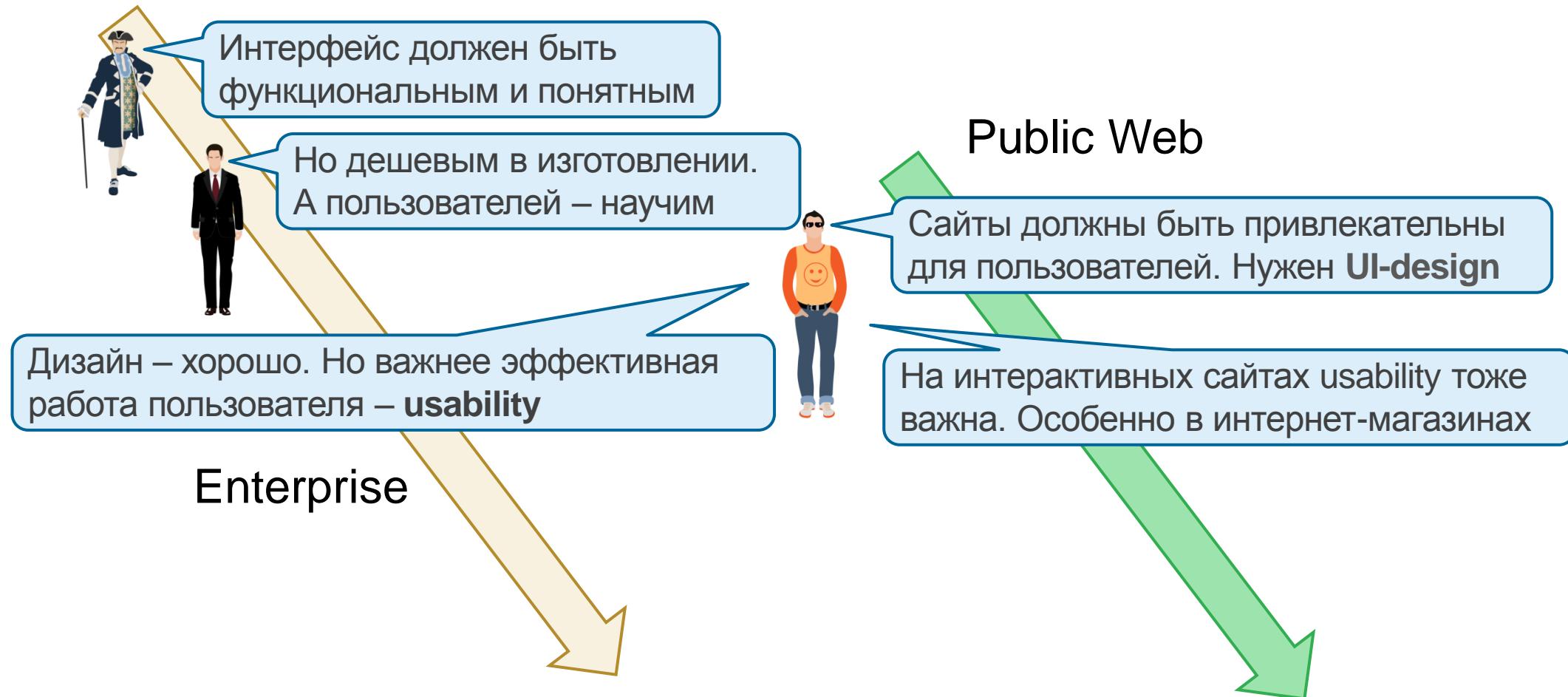
- ➡ Часть «для того чтобы»
 - Позволяет разработчику занять позицию пользователя при реализации фичи
 - Говорит о бизнес-целях использования
 - Позволяет принимать решения в процессе реализации
- ➡ Эта часть появилась не сразу, а из опыта использования
- ➡ О фиксации позиции пользователя заговорили и в других форматах требований

Пример: аналитик в отделе платежей

Аналитик пришел в отдел платежей, чтобы узнать детали проведения платежей по сделкам

- ◆ **Неверно:** быстро выяснить текущие вопросы и уйти
- ◆ **Верно:**
 - Узнать и записать всех, кто выполняет платежи и будет участвовать во внедрении
 - Спросить о проблемах нынешнего процесса проведения платежей и о других трудностях
 - Не забыть спросить про SLA прохождения платежей

UI-design и usability



Современный этап развития – «новое время»

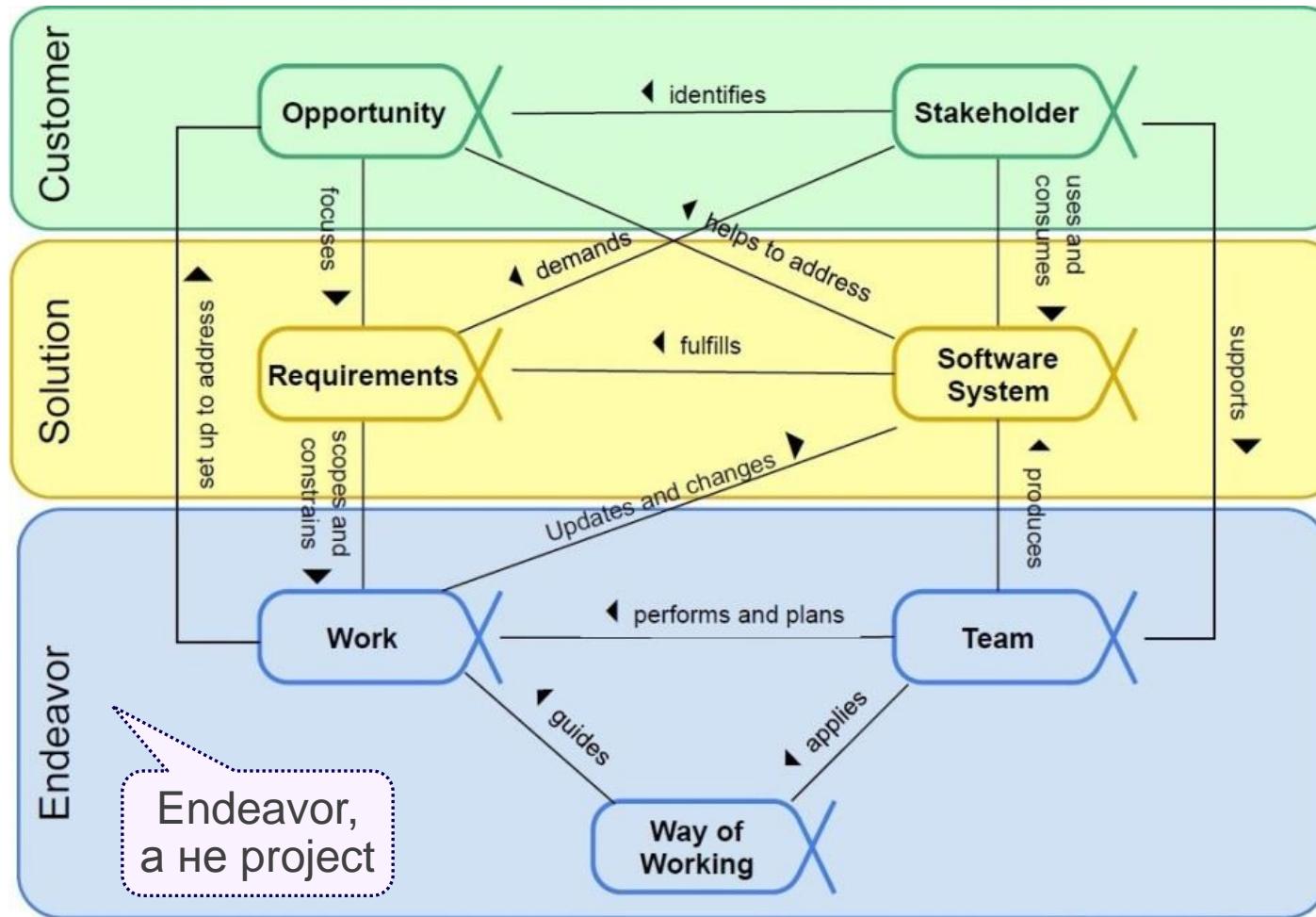
Современные вектора развития

- ➡ От проектной деятельности – к непрерывному развитию продукта
 - Канбан в ИТ (2010)
 - DevOps (2012)
- ➡ От качества ИТ-системы –
 - PMBOK 5 (2013)
(частично)
- ➡ От создания системы – к достижению возможностей для бизнеса и пользователя
- ➡ Каждой ИТ-разработке – свой метод
 - OMG Essence (2012)

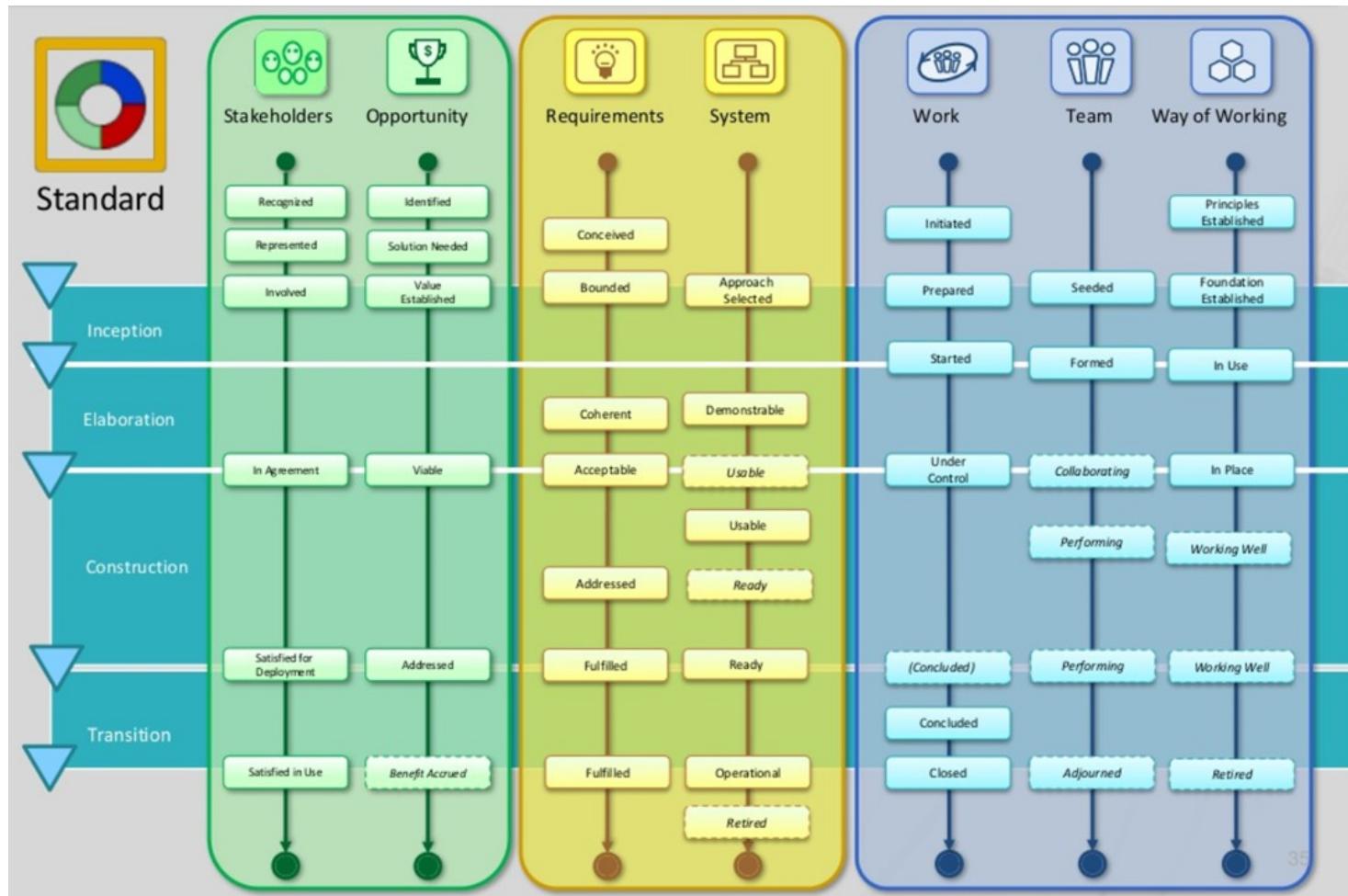


Качественная ИТ-разработка удовлетворяет стейкхолдеров и обеспечивает возможности для бизнеса

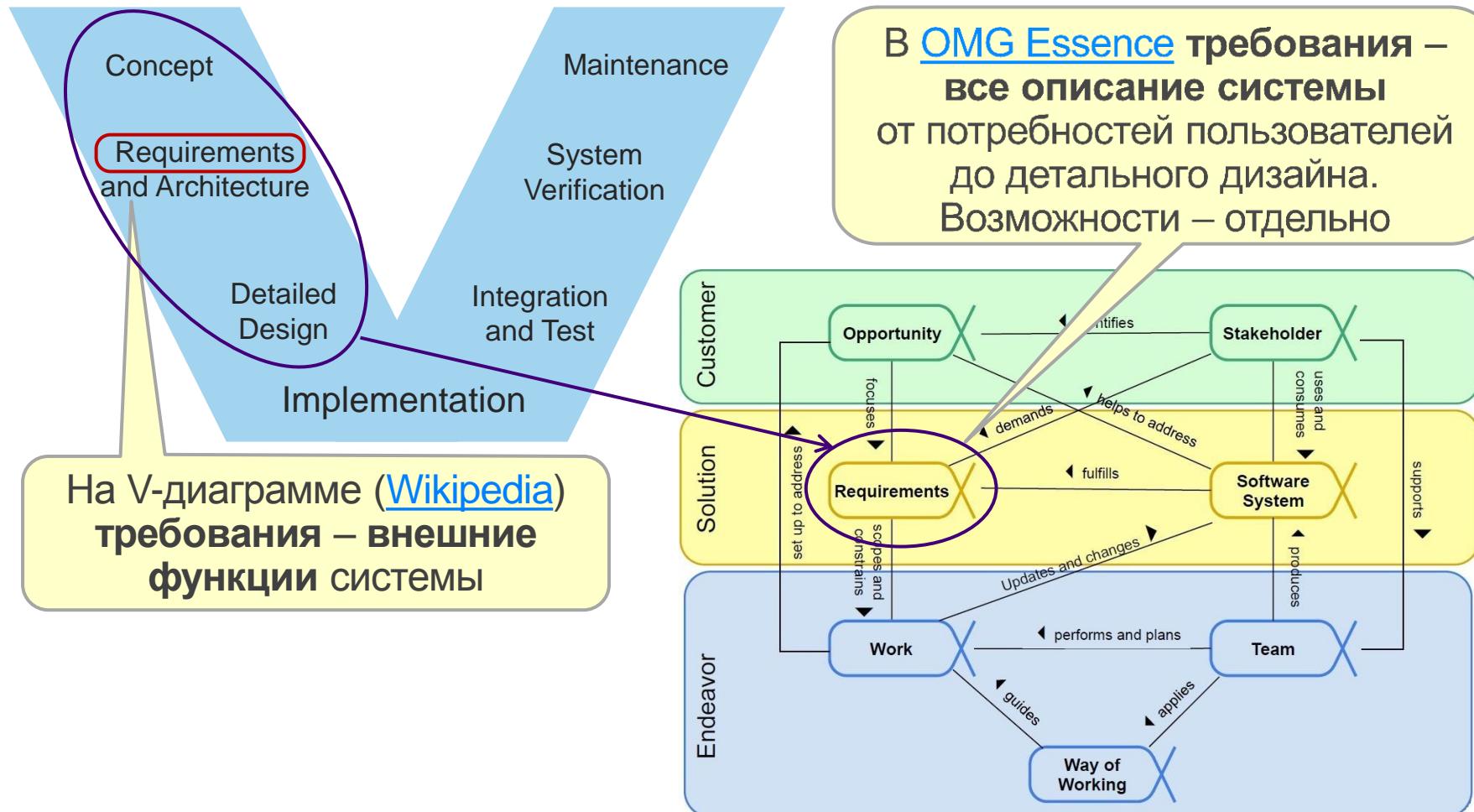
Альфы проекта – множество фокусов



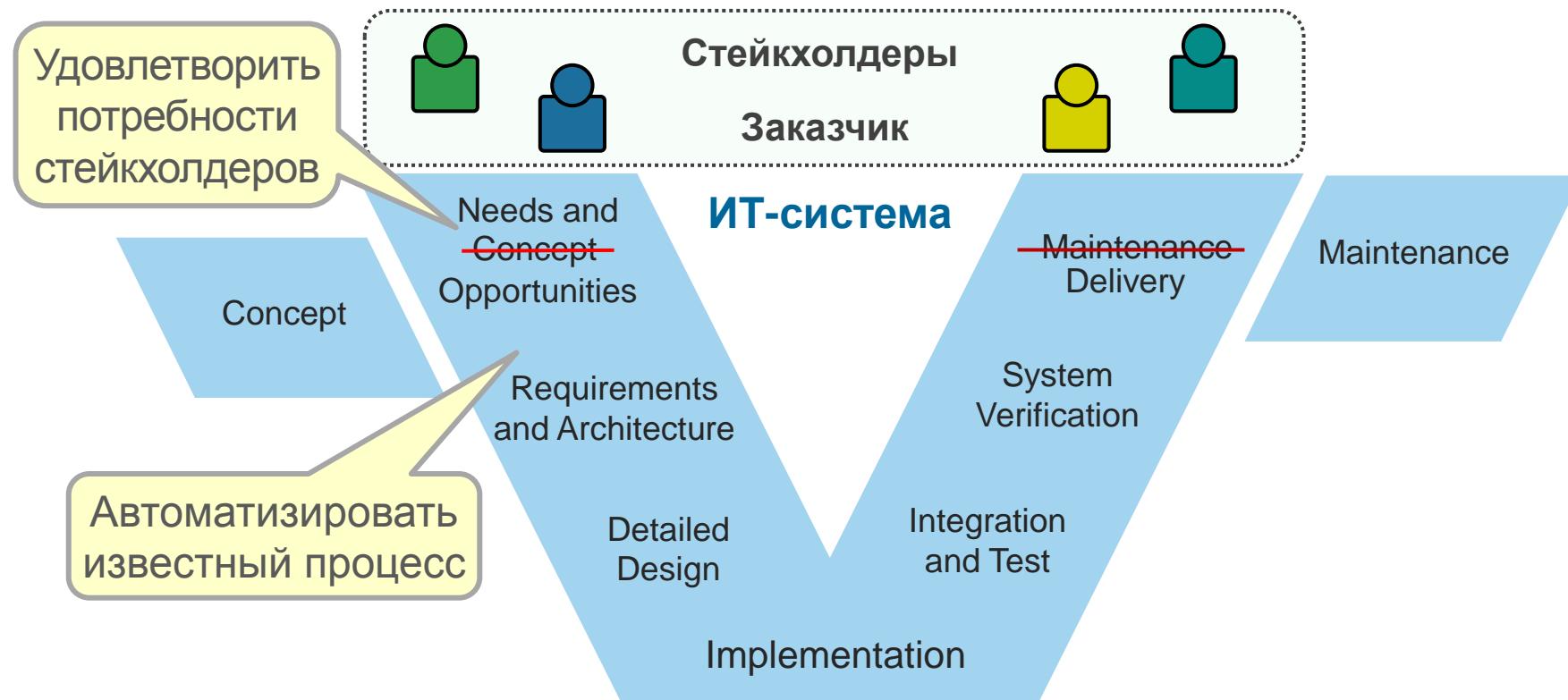
Каждая альфа живет собственной жизнью



Что такое «требования»: два ответа



Что является целью проекта?



Удовлетворение стейкхолдеров означает, что их оценка результатов не изменилась с «проект принес пользу бизнесу» на «опять ничего не получилось». Фокус сместился примерно в 2010 году

Работа с ожиданиями стейкхолдеров



Техника описания ожиданий – [ArchiMate Motivation Model](#)
и [Модель i* \(i-star\) описания целей](#)

OMG Essence – удовлетворенность стейкхолдеров как критерий успеха проекта



Организовать стейкхолдеров – часть проекта

- ◆ Agile возлагал эту задачу на product owner единолично, раньше ее выполнял руководитель проекта
- ◆ Практика показывает: система работает плохо, когда решение принимает единственный арбитр
- ◆ Задача проекта (руководителя, аналитиков и других) – создать социальную систему, обеспечивающую баланс интересов и удовлетворение стейкхолдеров

ИТ- и бизнес-проект объединяются



Пример: вложенные проекты – реинжиниринг ИТ банка

▶ Для банка в целом

Что: каждому подразделению – отдельная система

Зачем: независимое развитие сегментов бизнеса поддерживается ИТ

Как: компонентная архитектура и единая шина интеграции

▶ Для центральной бухгалтерии – главная книга

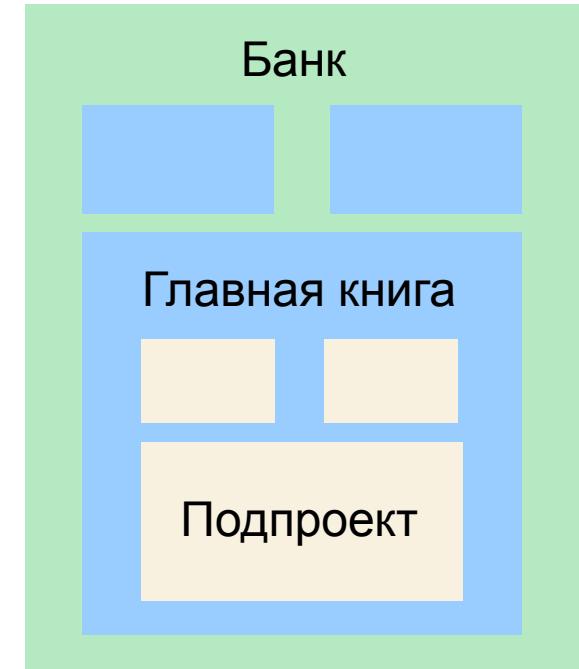
Что: учет операций всеми в соответствии с учетной политикой

Зачем: ведение достоверного учета и отчетности для регулятора

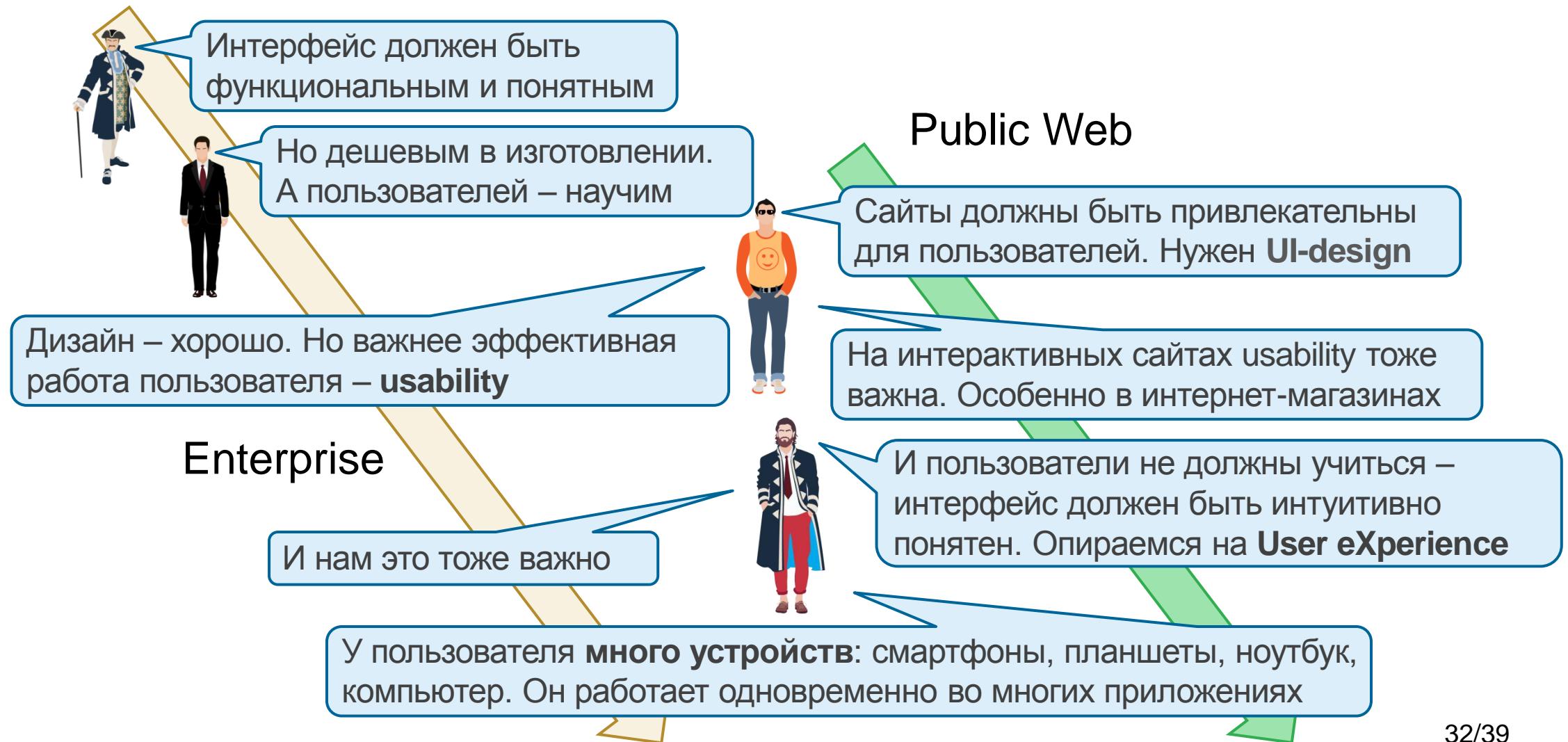
Как: правила корреспонденции счетов, контрольные отчеты, техническое решение – подтверждение операций главной книгой

▶ Подпроект внутри – исправительные документы

Что: автоматизировать работу с исправительными документами, не нарушив нормативное регулирование, рассчитанное на бумажные документы



UI-design и usability



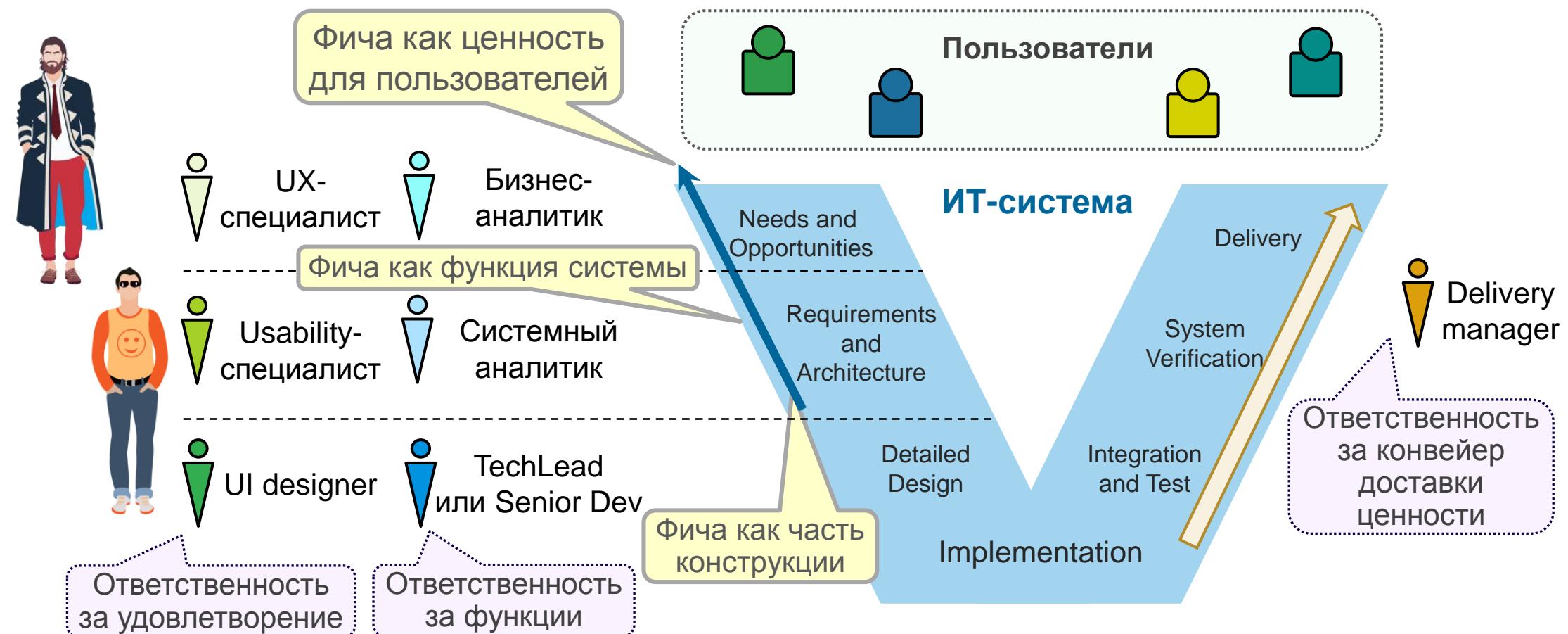
Пример: отбор товара по интернет-заказу в магазине

Сотрудник магазина отбирает в мобильном приложении товары по интернет-заказу

- ▶ **Неверно:** сотрудник сканирует товар – и строки в заказе помечаются
- ▶ **Верно:** как сотрудник узнает товар – нужно ли показать фотографии? нужно ли показать, где товар лежит? что сотрудник делает с отобранным товаром: кладет в коробку, где тот ждет покупателя? как найдут коробку, когда покупатель придет? как напечатать список товаров из мобильного телефона?
- ▶ **Верно:** сколько у сотрудника заказов? бывает ли, что их много и надо, чтобы подключились другие?
- ▶ **Верно:** что делать, если товар не найден? как долго его надо искать?

Над решением работаем совместно с технологами магазина

Удовлетворить пользователей → новые специализации



Подводим итоги

Что такое успешный проект?



Мы создали **качественную систему**
в соответствии со спецификацией требований



... и при этом **уложились в сроки**
и бюджет проекта – заказчик доволен



Мы создали тот **софт, который нужен**
заказчику, опираясь на обратную связь
и сотрудничая с ним



Главное, что **стейкхолдеры проекта**
могут достигать своих бизнес-целей
в соответствии с ожиданиями от проекта

Если требования неверны и нужна другая система?



Жаль, что все это выяснилось так близко к сдаче проекта.
Все сделано по требованиям – **вы должны это принять**.
А потом мы готовы сделать новый проект за новые деньги



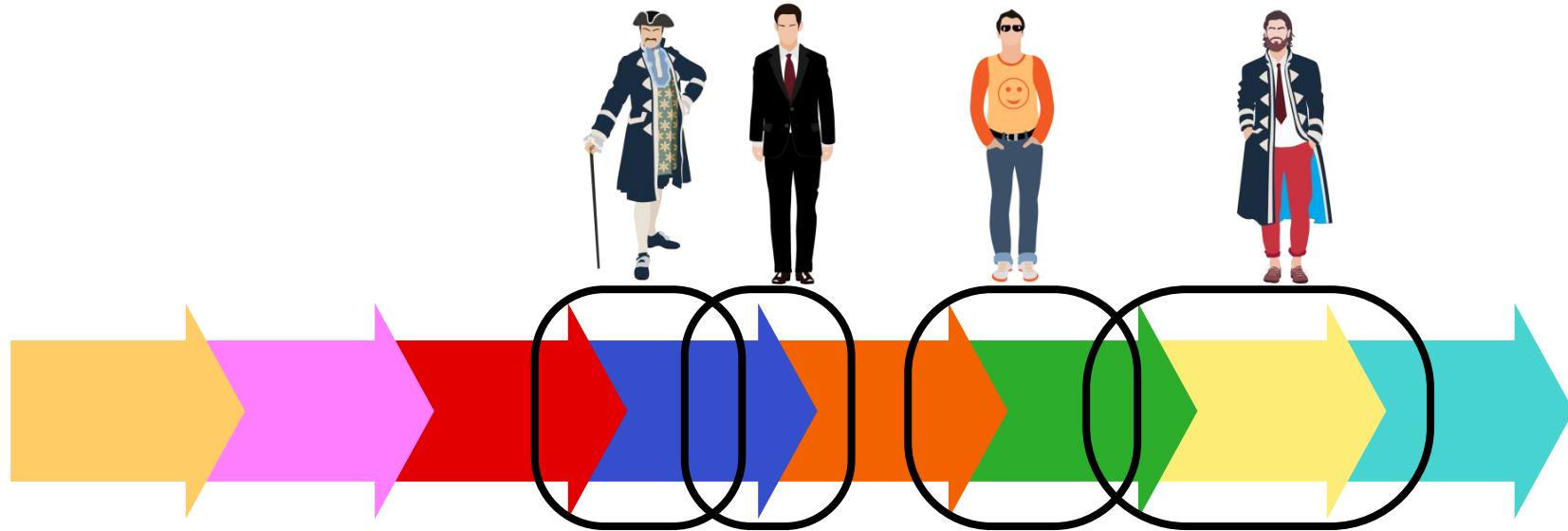
Частые демонстрации работающего софта
позволяют проверить его адекватность задачам
заказчика и **скорректировать движение проекта**



Если при очередной демонстрации выясняется, что софт
не позволяет решить задачу бизнеса, **команда вместе**
со стейкхолдерами заказчика ищет решение. Успех проекта –
реализация такого решения. Деньги и сроки – предмет переговоров

Культура ведения проектов – это mindset

Изменения mindset описывает спиральная динамика, и культуры укладываются в ее картину уровней



На каждом уровне **понятия переосмысливаются**: партнерство, успех проекта, работа с ожиданиями и другие

О спиральной динамике – смотри [мои доклады и статьи](#)

Подумайте: какая культура у вас в проекте?



Максим Цепков
<http://mtsepkov.org>

На сайте много материалов по [архитектуре](#)
[и анализу](#), [ведению проектов](#), [agile](#), мои
доклады, статьи и [конспекты книг](#)