Тема 1

«История предметной области, основные понятия и термины»

ИЛИ

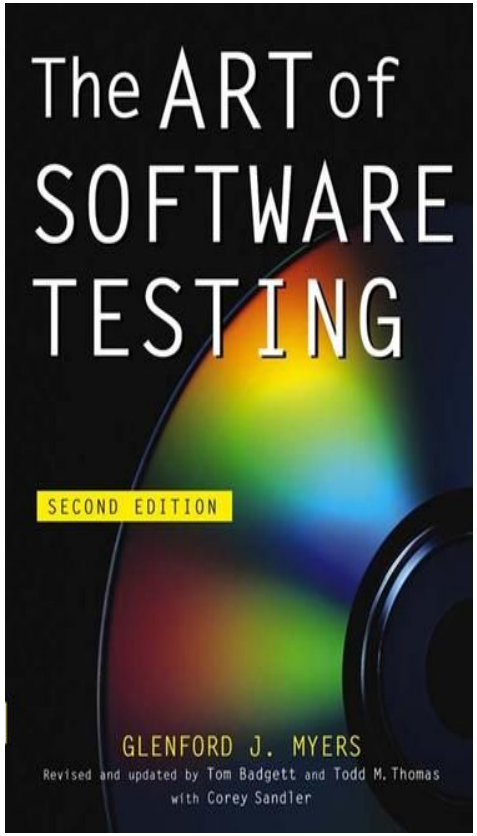
почему тестирование – это важно и интересно

Немного истории…

60-е годы.

60-е годы – «исчерпывающее тестирование»

20 вложенных операторов if => 1’048’576 ветвей выполнения



70…80-е годы.

70-е годы – «поиск дефектов»

Почему?

80-е годы – «предупреждение дефектов»

Какие методы предупреждения дефектов Вы знаете?

60-е годы – «программа работает»

VS

70-е годы – «программа НЕ работает»

80-е годы – «предупреждение дефектов»

80-е годы – «предупреждение дефектов»

ЭТО СРАБОТАЛО

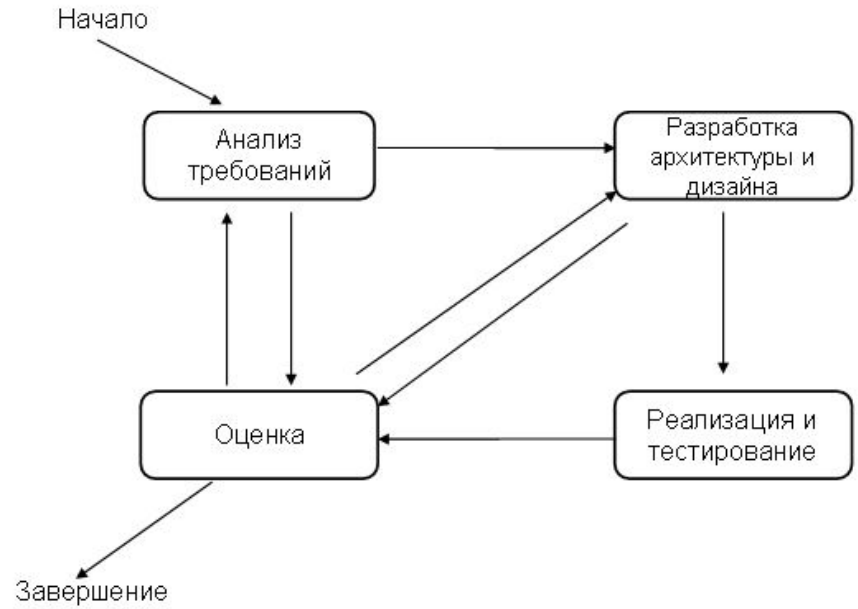
90-е годы – «обеспечение качества»

0-е ☺ годы – «тотальное обеспечение качества»

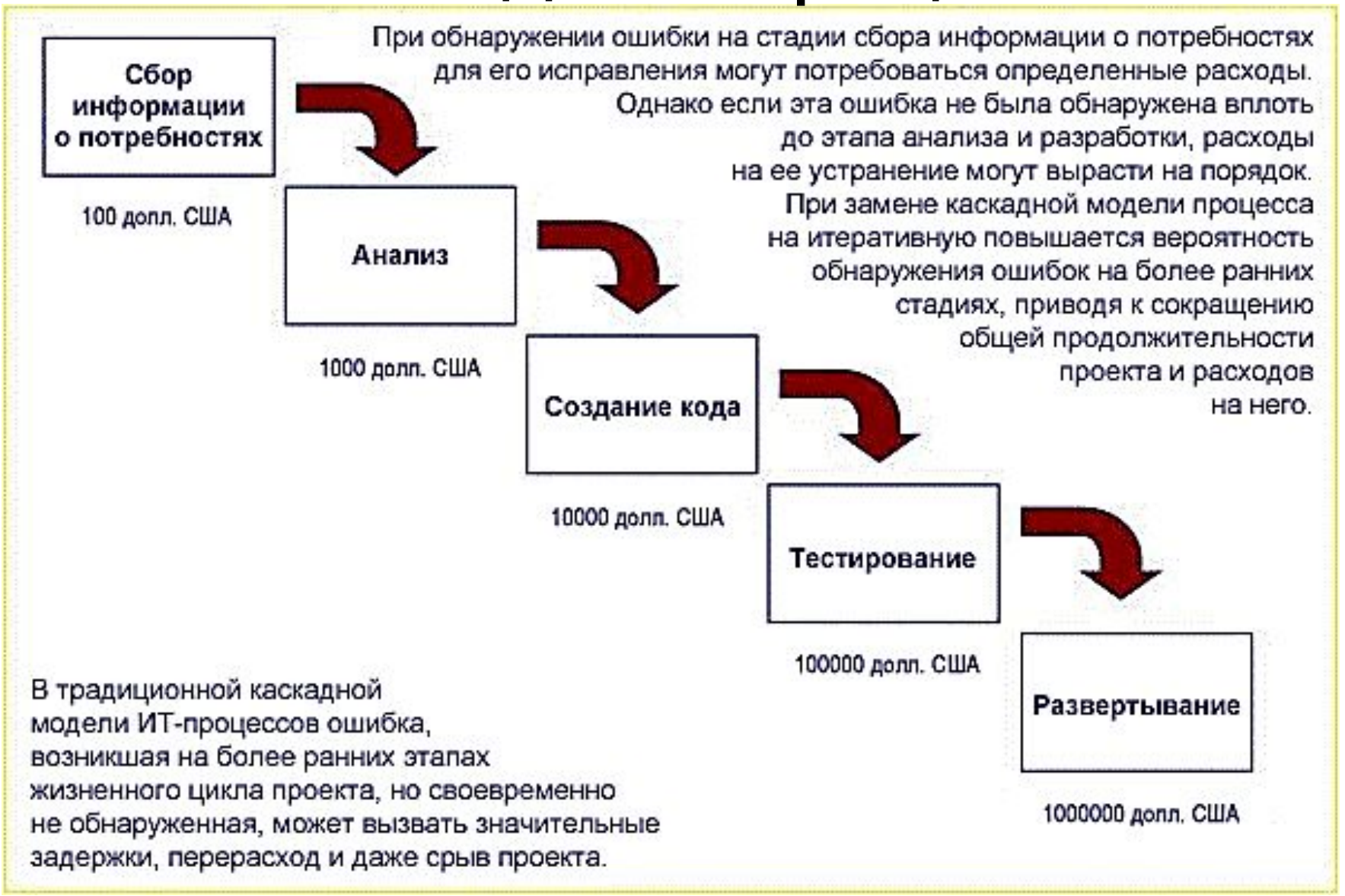
Современный этап – «гибкие методологии, тесная интеграция с разработкой, автоматизация»

Кстати, о методологиях…

Классические методологии и модели разработки ПО: водопадная, итерационная...



Каскадный процесс



http://msdn.microsoft.com/ru-ru/library/ee909663.aspx

Agile Manifesto разработан и принят 11-13 февраля 2001 года на лыжном курорте The Lodge at Snowbird в горах Юты.

Манифест подписали представители следующих методологий:

- Extreme programming
- Scrum
- DSDM
- Adaptive Software Development
- Crystal Clear
- Feature-Driven Development
- Pragmatic Programming.

Содержит 4 основные идеи и 12 принципов.
Не содержит практических советов!

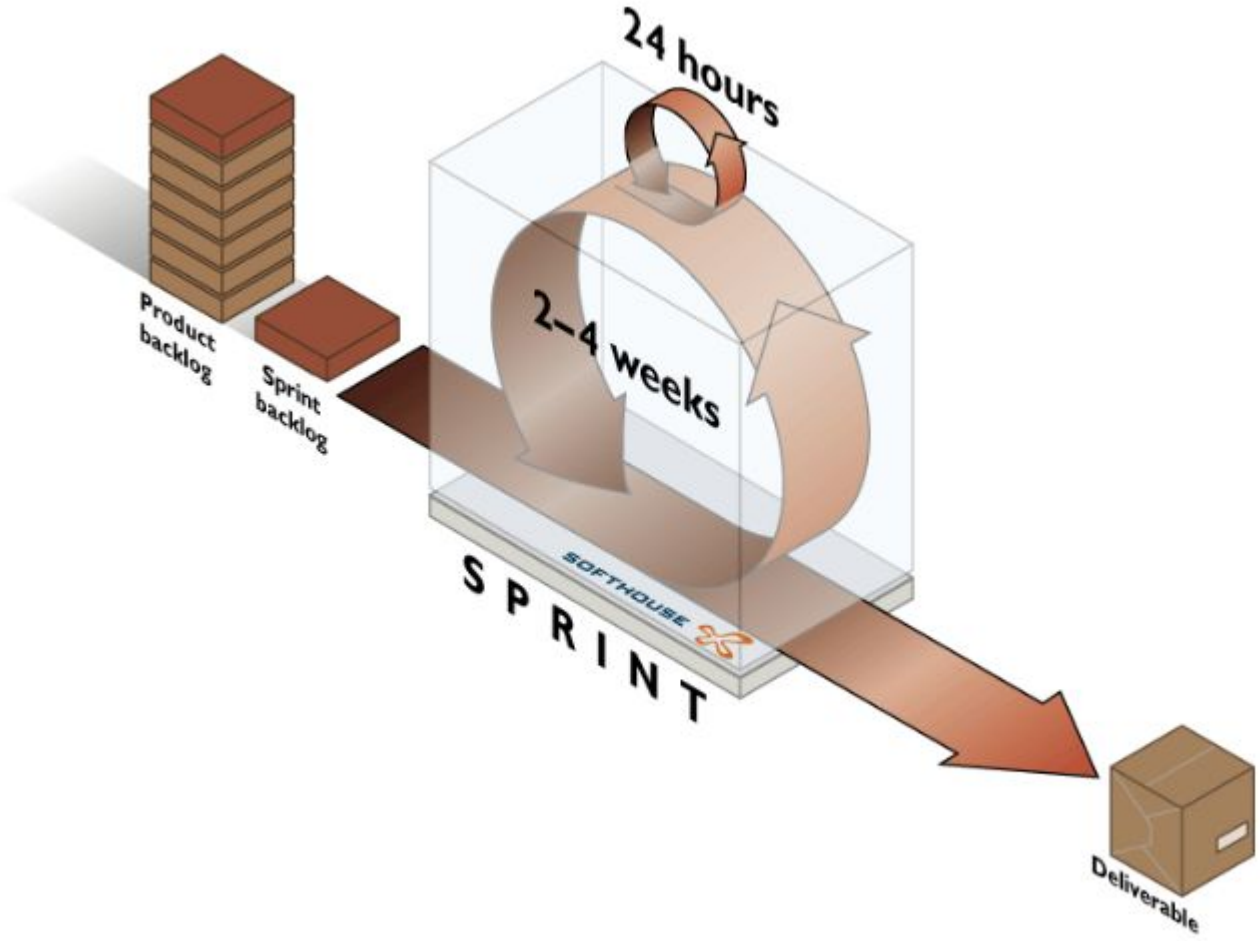
Идеи:

- 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.

Принципы:

- удовлетворение клиента за счёт ранней и бесперебойной поставки ценного ПО;
- приветствие изменений требований, даже в конце разработки;
- частая поставка рабочего ПО (каждый месяц или неделю или ещё чаще);
- тесное, ежедневное общение заказчик <-> разработчики;
- проектом занимаются мотивированные личности, которые обеспечены нужными условиями работы, поддержкой и доверием;
- рекомендуемый метод передачи информации — личный разговор;
- работающее ПО — лучший измеритель прогресса;
- спонсоры, разработчики и пользователи должны иметь возможность поддерживать постоянный темп на неопределенный срок;
- 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.

Гибкие методологии и модели разработки ПО: Agile, Scrum... и множество других.



Важность тестирования

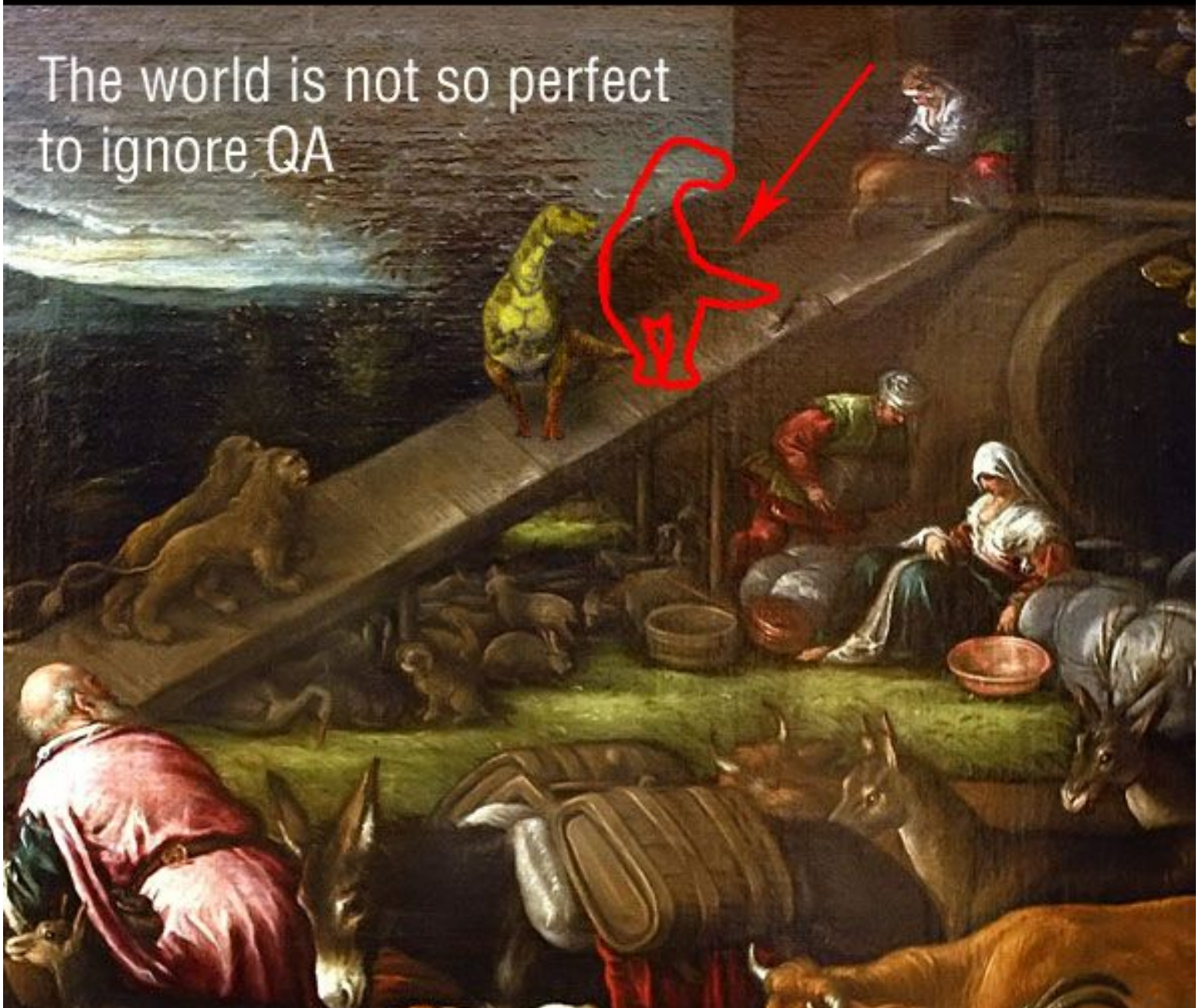
Тестирование приобрело особую важность в силу нескольких причин…

Каких?

Бизнес:
«пользователи
склонны пользоваться
качественными
продуктами (даже
если они дороже)»

Пользователи: «лучше не рисковать личными данными, деньгами и т.п.»

Project: Noah's Ark
Smoke test: failed



Все: «мы не хотим рисковать»

Наконец, тестирование – это…

- относительно новая;
- стремительно развивающаяся;
- интересная;
- находящаяся на границе многих смежных дисциплин;

… область информационных технологий.

Brainstorming

Как вы думате, что делает тестировщик?

Чем занимается тестировщик?

Контроль качества

Обеспечение качества («профилактика» и «здоровый образ жизни»)

Качество продукта, и в частности тестирование, влияют на конечный результат – удовлетворенность заказчика.

Фактически, **«тестирование ПО»** – это **«диагностика» и «помощь в лечении»** программного средства как такового и всего проекта в целом.

Brainstorming

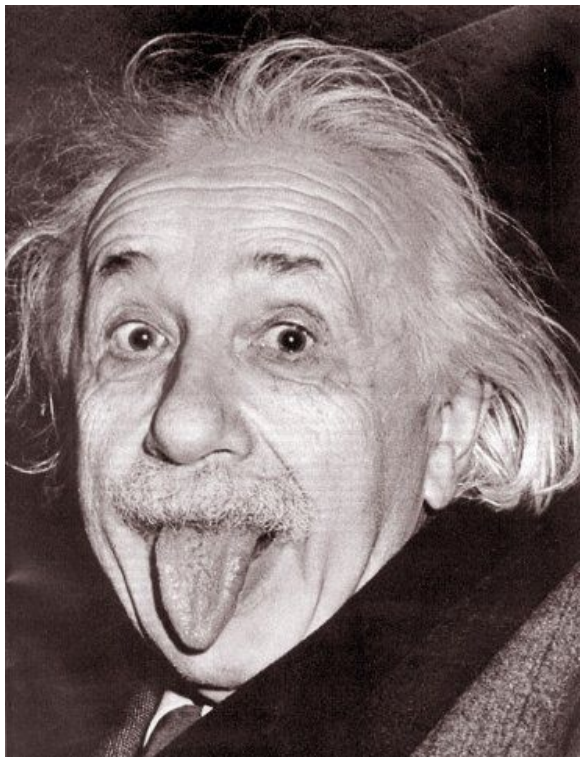
Как вы думаете, а что хороший тестировщик должен знать?

Знание иностранных языков.

Первоначально важно хотя бы "Умение излагать мысли и замечания на родном языке для тестировщиков", а потом уже "Английский для тестировщиков" !

Технические навыки:

- Программирование: C/C++/C#, Java, PHP, Obje Pascal, Visual Basic, JavaScript, HTML, .NET.
- Администрирование СУБД: Oracle, MS SQL, MySQL.
- Администрирование ОС: Windows, Sun Solaris, HP-UX, Free-BSD, Linux.
- Сетевое администрирование: TCP/IP, IPX/SPX, NetBIOS.
- Автоматизированное тестирование: Silk*, Rational*, Mercury Interactive *, JUnit, HTTP/HTML-Unit.

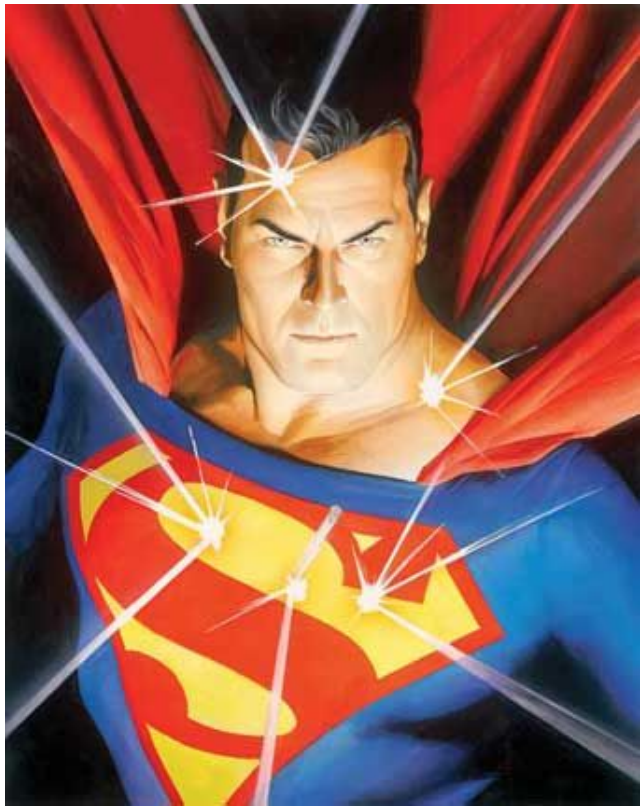


Тестировщику приходится выполнять ответственную работу и много общаться.

Каким он для этого должен быть?

Brainstorming

А какими психологическими навыками и особенностями должен обладать тестировщик?



Психологические навыки и особенности тестировщика таковы:

- Повышенная ответственность.
- Хорошие коммуникативные навыки.
- Способность ясно, быстро, чётко выражать свои мысли.
- Исполнительность.
- Терпение, усидчивость, внимательность к деталям, наблюдательность.
- Гибкое мышление, хорошая способность к обучению.
- Хорошее абстрактное и аналитическое мышление.
- Способность ставить нестандартные эксперименты.
- Склонность к исследовательской деятельности.

Тестировщик – полноправный участник проекта, но…

… какое место он занимает в команде? Кто он?

Не принципиально, как называется ваш труд и с кем вы сидите в одной комнате. Главное – сотрудничество, направленное на достижение общей цели!

Поговорим о терминологии

Тестирование программного обеспечения (software testing) – процесс анализа программного средства и сопутствующей документации с целью выявления дефектов и повышения качества продукта.

выявления дефектов и **повышения** качества продукта.

Дефект (баг, глюк; defect, bug) – любое несоответствие фактического и ожидаемого результата (согласно требованиям или здравому смыслу).

А что мы сразу же легко определяем как дефект в любой программе, даже никогда не видев требований к ней?

Ожидаемый результат (expected result) – такое поведение программного средства, которое мы ожидаем в ответ на наши действия.

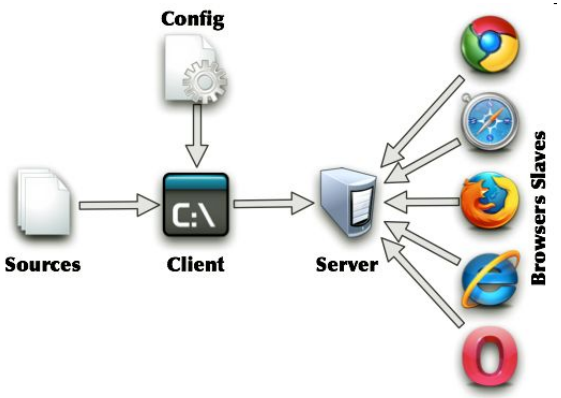
Откуда (как) мы можем узнать, что наши ожидания логичны и верны?

Чек-лист (check-list) – набор идей тестов.

Почему мы не сразу приступаем к разработке тестов?

Приведите пример чек-листа из Вашей жизни

Тест-кейс (test case) – набор входных данных, условий выполнения и ожидаемых результатов, разработанный с целью проверки того или иного свойства или поведения программного средства.



Что случится, если у нас не будет:
а) входных данных;
б) условий выполнения;
в) ожидаемых результатов;
г) цели.

Тестовый сценарий, тест-сьют (test scenario, test-suite) – набор тест-кейсов, собранных в группу (последовательность) для достижения некоторой цели.

И снова: а в жизни бывают примеры тест-сьютов?

Тест-план (test plan) – часть проектной документации, описывающая и регламентирующая процесс тестирования.

Что, на ваш взгляд, нужно отразить в таком плане?

Может быть, кто-то приведёт пример похожего плана из жизни?

Определение теста и тестового набора

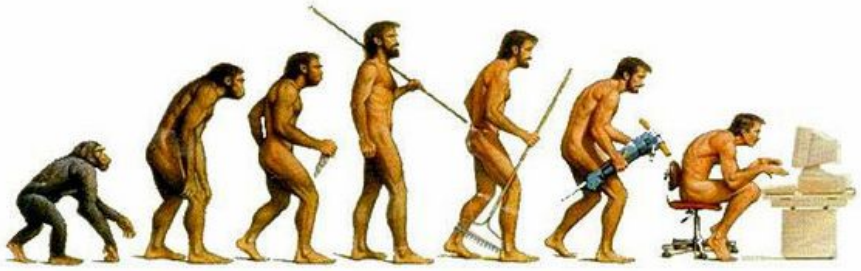
Тестовый набор – практические соображения:

Почему?

- После объединения тестов в наборы не должно оставаться незадействованных тестов
- При проектировании тестов «сверху вниз», тест кейсы будут являться частями тестовых наборов
- Рекомендую именно проектирование тестов сверху вниз

Почему?

Билд («сборка») (build) – промежуточная версия программного средства (финальный билд часто называют релизом (release)).



Можем назвать пример билдов в повседневной жизни?

Качество (quality)

- Качество (quality) – показатель степени соответствия продукта его требованиям.

Как мы в повседневной жизни определяем, что какая-то вещь, какая-то работа и т.д. могут быть названы «качественными», например – обувь?

Качество продукта определяется качеством процесса его разработки

- Некоторые рассуждения о качестве:
  - Если заказчик доволен продуктом – продукт качественный.
  - Если продукт соответствует требованиям – продукт качественный.
  - У качественного продукта всегда есть преимущества и нет серьёзных недостатков

Заказчик должен быть **отсатисфачен!**

Как посчитать?

- Оговорить критерии
- Выяснить, какие **показатели** будут критичны Заказчику или Компании

Как они называются?

Выпуск хорошего продукта это цель не только тестировщика, а всех вместе взятых!

Метрики качества (quality metrics)

- **Метрика качества** (quality metric) – числовое значение некоторого показателя качества.
  Может определяться расчётным способом или по некоторой формуле.

$$J_\lambda(x_2, y_2, s_2) = \iint I_\lambda(x_2, y_2) \cdot \left| m_\lambda \left(\frac{x_2 - x_0}{\lambda \cdot s_2}, \frac{y_2 - y_0}{\lambda \cdot s_2} \right) \right|^2 dx_0\, dy_0 =$$

$$= I_\lambda(x_2, y_2) \otimes \left| m_\lambda \left(\frac{x_2}{\lambda \cdot s_2}, \frac{y_2}{\lambda \cdot s_2} \right) \right|^2$$

Сделайте мне удобный интерфейс …

Варианты метрик

- **Покрытие требований** тестами – не менее 80%
- **Плотность покрытия** – не менее 3
- **Закрыто 100%** известных **критических** дефектов, **90%** дефектов **средней** критичности, **50% остальных** дефектов.
- **Общий показатель прохождения тестов** – не менее некоторого значения:

  X = (Passed/Executed)*100%

Есть вопросы? Давайте обсудим!

Составляющие качества

Наука простым языком

Функциональные возможности

Что я могу с помощью этого сделать?

Функциональная пригодность

Могу ли я с помощью этого сделать … ?

Правильность (корректность)

Правильно ли это сделано (работает)?

Способность к взаимодействию

Могу ли я соединить это с …?

Защищённость

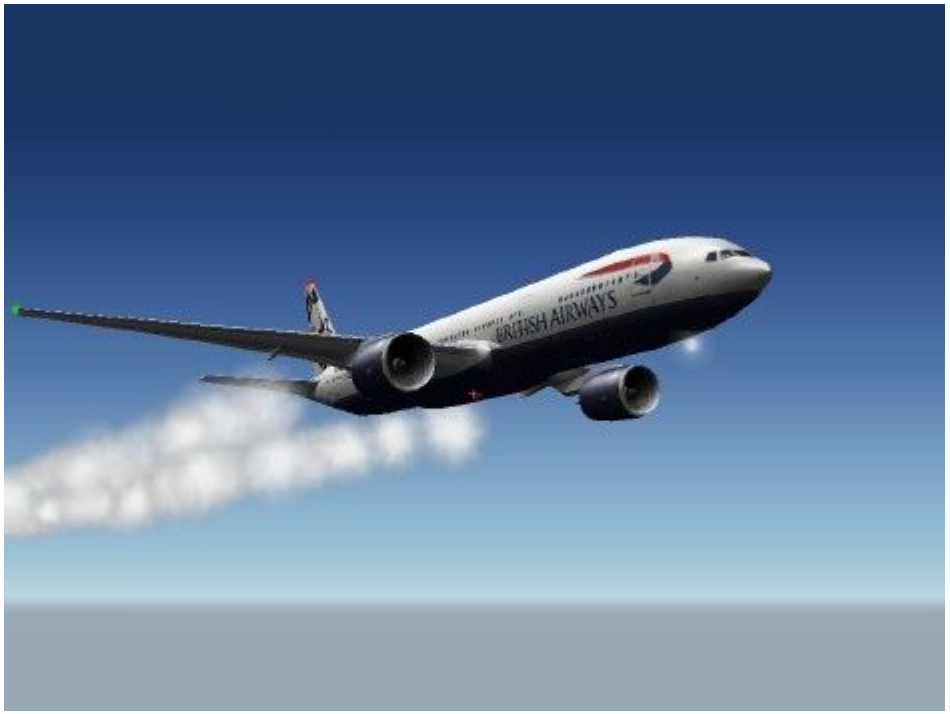
А если придут злоумышленники?

Надёжность

А оно не сломается?

Эффективность

Ему много ресурсов понадобится?



Практичность (применимость)

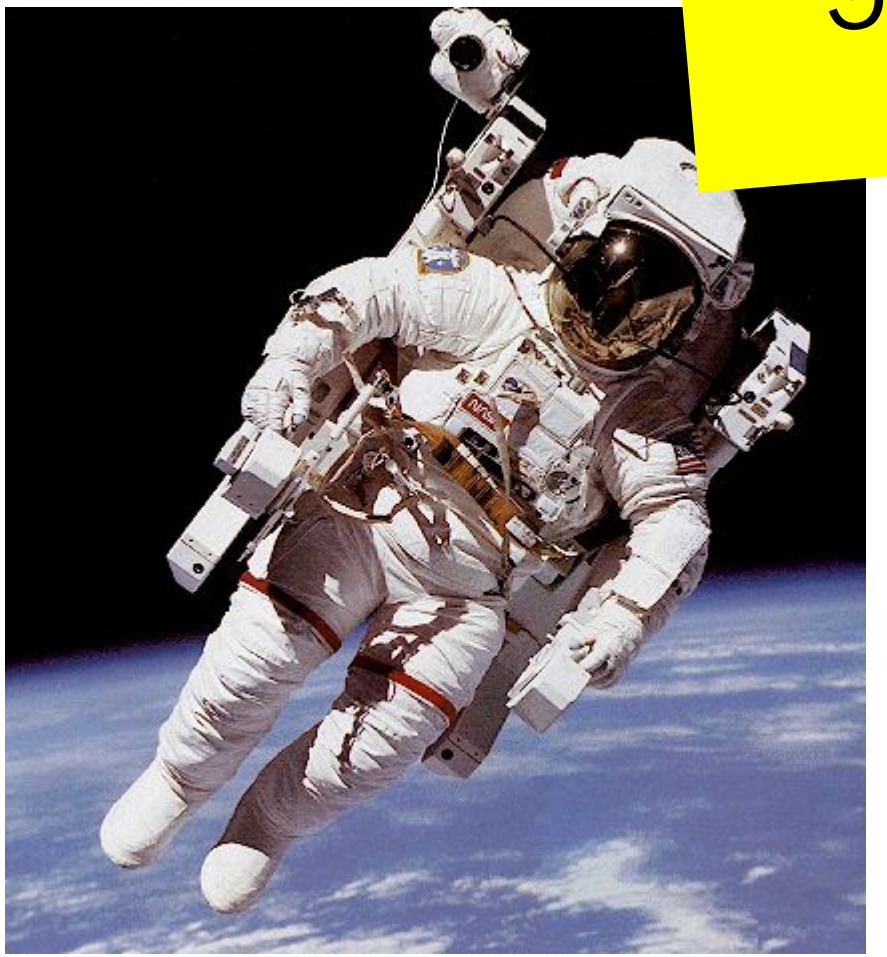
А я точно смогу с помощью этого ... ?

Сопровождаемость

Я смогу это доработать, улучшить?

Мобильность

Это сможет работать где-то ещё?



Основная сложность тестирования программ – это...

И что же это? Как вы думаете?

Основная сложность тестирования программ – это...

невозможность всё предусмотреть
в силу концептуальности ПО

Семь шагов к успеху

Анализ требований

Планирование тестовых испытаний

Проектирование тестов

Реализация и отладка тестов

Системное тестирование

Приёмочные испытания

Сопровождение

В чём заключается тестирование на каждом из этапов?

Что мы можем тестировать

А и вправду – что?

Программы при их непосредственном запуске и исполнении (software)

Код программ без запуска и исполнения (code)

Прототип программного продукта (product prototype)

- Что может служить прототипом:
  - Исследование имеющегося у заказчика продукта, который следует улучшить.
  - Исследование продуктов конкурентов.

А что еще может служить прототипом?
(подсказка: особенно в Agile разработке)

Проектную документацию (project documentation):

- Требования к программному продукту (product requirements).
- Функциональные спецификации к программному продукту (functional specifications).
- Архитектуру (architecture) и дизайн (design).
- План проекта (project plan) и тестовый план (test plan).
- Тестовые случаи и сценарии (test cases, test scenarios).

Сопроводительную документацию (и документацию для пользователей):

- Интерактивную помощь (on-line help).
- Руководства по установке (Installation guide) и использованию программного продукта (user manual).

Давайте что-нибудь протестируем!

Итого, что мы узнали сегодня?

Есть вопросы? Давайте обсудим!

Роман Савин

«Тестирование .com или пособие по жестокому обращению с багами в интернет-стартапах»

Рекс Блэк

«Ключевые процессы тестирования»