Управление процессом тестирования ПО

Тест-менеджмент

2008, v.3.2

АВТОРСКИЙ КОЛЛЕКТИВ

Вячеслав Панкратов
http://pankratov.org.ua/
Дмитрий Лысенко
http://dmitrylysenko.info/

Расписание тренинга

- **День 1:**
  - Основы тестирования и обеспечения качества, обзор ролей и артефактов тестирования
  - Тест-дизайн
  - Автоматизация тестирования ПО
  - Тестирование производительности ПО
- **День 2:**
  - Организация основных процессов тестирования
  - Тест-менеджер: задачи, ответственность
  - Коллеги, подчинённые, руководство
  - Риски: как начать
  - Управление временем, организация работ

Содержание

- **Знакомимся и проясняем ожидания**
- **Определение тестирования ПО и качества информационных систем: чем мы всё-таки занимаемся?**
- **Цели и задачи тестирования ПО**
  - **Уровни и этапы тестирования**
  - **Организация работы с дефектами**
  - **Обзор задач, ролей и артефактов в тестировании**
- **Тест-дизайн**
- **Автоматизация тестирования ПО**
- **Тестирование производительности**

Содержание

- **Работа с рисками: как начать это дело?**
- **Успешное управление: эффективность**
  - Работа с подчиненными
  - Работа с руководством, коллегами и заказчиками
  - Управление задачами с использованием модели FACT и др.
  - Правильное управление своим временем

Базовые понятия

- Качество Продукта
- Процесс производства ПО
- Связь: Качество-Процесс-Продукт
- Тестирование ПО: как продать тестирование, миссия, фокус усилий

Качество Продукта

- **Что такое Качество Продукта**
- **Функциональность, Надежность, Производительность**
  - Функциональность — делает ли приложение то, что от него требуется
  - Надежность — работает ли приложение без сбоев, «зависаний» или вызова исключений
  - Производительность — работает ли приложение с приемлемой скоростью при доступе к нему многих пользователей

Корни и ветки дерева качества

- **Откуда берется Качество?**
  - Качество Продукта определяется только Качеством Процесса его разработки
  - Качество Процесса определяется только Уровнем Культуры разработки в Компании

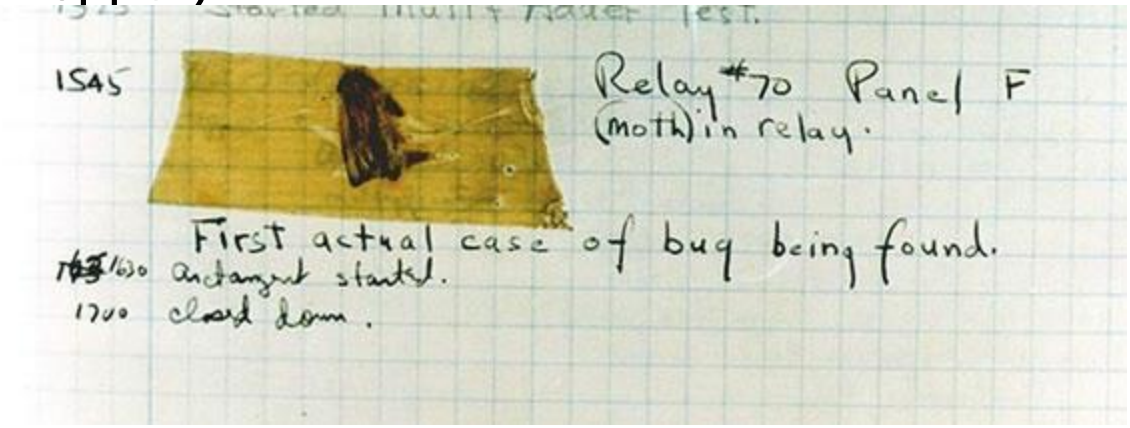
Как продать Качество?

- **Типичные проблемы с «продажей» качества:**
  - Неочевидность возврата инвестиций
  - Большой первоначальный объем работ: обзоры, инструменты, обучение
  - Хорошую работу не видно!

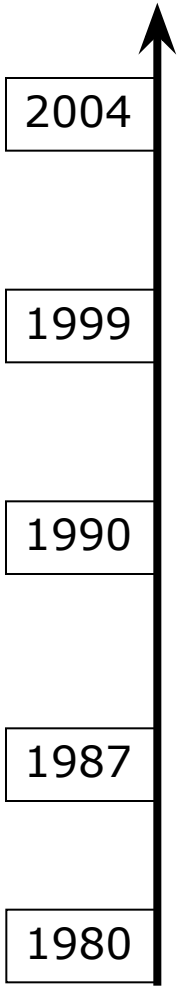
Тестирование ПО: истоки

Первый баг

09.09.1945г. Грейс Мюррей Хоппер (Grace Murray Hopper)



Эволюция представлений о тестировании



- Проверка соответствия между реальным поведением программы и ее ожидаемым поведением на конечном наборе тестов, выбранном определенным образом. *[IEEE Guide to Software Engineering Body of Knowledge, SWEBOK, 2004]*
- Техническое исследование программы для получения информации о ее качестве с точки зрения определенного круга заинтересованных лиц. *[C. Kaner, 1999]*
- Это не действие. Это интеллектуальная дисциплина, имеющая целью получение надежного программного обеспечения без излишних усилий на его проверку. *[B. Beizer. Software Testing Techniques, Second Edition. NY:van Nostrand Reinhold, 1990]*
- Процесс наблюдения за выполнением программы в специальных условиях и вынесения на этой основе оценки каких-либо ее аспектов. *[ANSI/IEEE standard 610.12-1990: Glossary of SE Terminology. NY:IEEE, 1987]*
- Процесс выполнения программы с намерением найти ошибки. *[Г.Майерс. Надежность программного обеспечения. М:Мир, 1980]*

Определение тестирования (классика)

- **Глен Маерс:**
  - Тестирование это процесс выполнения программ с намерением найти ошибки
- **Пол Йоргенсен:**
  - Тестирование сфокусировано на ошибках и сбоях. Тест – выполнение действий над ПО с целью найти ошибки или продемонстрировать работоспособность

Расширенное определение Тестирования

- **Обобщенное определение тестирования**
  - Тестирование – процесс проверки соответствия заявленных к продукту требований и реально разработанной функциональности
- **Преимущества**
  - Фокус процесса тестирования смещен в сторону проверки требований

Уточнённое определение Тестирования

Уточним определение

Тестирование – процесс проверки соответствия заявленных к продукту требований и реально реализованной функциональности, осуществляемый путем наблюдения за его работой в искусственно созданных ситуациях и на ограниченном наборе тестов, выбранных определенным образом

Задачи тестирования

- **Задачи тестирования намного шире чем поиск дефектов!**
- **Типичные вопросы, с которыми сталкиваются группы тестирования:**
  - Сколько ошибок мы нашли? N – это много или мало?..
  - Почему наши пользователи находят ошибки, если мы потратили на тестирование столько времени?!
  - Мы нашли N ошибок – мы можем остановить тестирование?
  - А как будет себя вести наше приложение в эксплуатации?..

Как продать Тестирование

- Тестирование ПО занимает от 30% до 50% от всей стоимости разработки, в случаях сложных инфраструктурных проектов до 80%
  - (!) Продукты Microsoft: на 1 разработчика 2 тестировщика
- Тестирование не только процесс **контроля**, но и мощный инструмент **разработки**, который позволяет достичь определенного уровня качества
- >>

Как ещё продавать Тестирование?

- Основная цель тестирования – достижение заданного уровня качества продукта
- Тестирование может не только выявлять и определять дефекты, но и **предотвращать** их
- Тестирование становится процессом, эффективность которого кардинально влияет на стоимость Продукта и Сроки его выхода на рынок

Стоимость исправления ошибок

- Миссия тестирования – снизить стоимость разработки путем раннего обнаружения дефектов;
- Миссия тестирования – отправная точка при продаже качества

Классификация тестирования

- По степени детализации
- По подходу к тестированию

Практические соображения

- **Одна из задач менеджера – развитие компетенции персонала**
  - Даже если вы уверены, что все знают как правильно называется тот тип тестов, который они выполняют, занесите это в базу знаний ☺
  - Думая над задачей «следующей ступеньки» для подчинённых – думайте про «горизонтальное развитие»

Уровень готовности: модули, Unit

- **Модульный уровень**
  - Тестирование целостности кода на уровне логических модулей
  - Выполняется разработчиками
  - Контролируется группой тестирования с помощью инструментов анализа покрытия кода unit-тестами (unit test coverage tools)

Уровень готовности: отдельные части

- **Интеграционный уровень**
  - Тестирование промежуточных результатов интеграции системы
  - Выполняется разработчиками и тестировщиками
  - Возможны подходы «сверху вниз» и «снизу вверх»

Уровень готовности: собранная система

- **Системный уровень**
  - Проверка полностью построенной системы на соответствие сформулированным требованиям
  - Подуровни:
    - Альфа-тестирование
    - Бета-тестирование
    - Приемочное тестирование

Уровень готовности: система

- **System level уровни**
  - Альфа-тестирование
    - Выполняется группой тестирования внутри команды/организации разработки
  - Бета-тестирование
    - Выполняется группой тестирования в среде дружественно настроенных клиентов
  - Приемочное тестирование
    - Выполняется заказчиком с целью определить, будет ли система принята в эксплуатацию
    - (!) Выполняется группой тестирования с целью определить, будет ли система принята в тестирование (*smoke testing*)

Как тестируем: снаружи

Классификация по применяемому подходу

- **Black box (Functional) Testing**
  - Тестирование с точки зрения конечного пользователя
  - Часто комбинируется с методиками «белого ящика»
  - Наиболее распространенные методики тестирования для user-oriented систем и приложений

Как тестируем: изнутри

- **White box (Structural) Testing**
  - Анализ приложения на уровне кода
  - Подвиды:
    - Ручное (экспертное тестирование кода)
    - Автоматизированное тестирование инструментами статического анализа
  - Один из наиболее «дорогостоящих» методов тестирования, требующий квалификации высокого уровня

Как тестируем: «умные руки»

- **Grey box testing («серый» ящик)**
  - Смешанная методика, применяемая опытными тестировщиками или разработчиками при отладке кода
  - Тестирование с применением знаний о коде приложения и подробностей реализации функциональности

Тест-анализ и тест-дизайн

- Работа с требованиями – почему не все это умеют и могут делать
- Расчёт трудоёмкости – сила методов тест-дизайна
- Миф про «универсального тестировщика»

Что было написано в требовании

- SRS-01 (до изменения)
  - Форма регистрации нового пользователя в системе “InfoSecurityManagement” позволяет вводить в реестр пользователей данные о пользователе и его роли:
    - Имя,
    - Доменное имя,
    - Должность,
    - Полномочия в системе

Что изменилось в требовании

- SRS-01.1 (после изменения)
  - Форма регистрации нового пользователя в системе “InfoSecurityManagement” позволяет вводить в реестр пользователей данные о пользователе и его роли:
    - Имя,
    - Доменное имя,
    - Должность,
    - Полномочия в системе
  - Если такой пользователь уже существует в реестре системы “InfoSecurityManagement”, на форме ввода появляется его E-mail адрес.

Практический кейс

- Что должно произойти в тест-кейсах?
- Кто это должен сделать?
- Когда это может происходить?
- Вы уверены, что ваш рядовой тестер понимает глубину задачи?

Как часто бывает и что с этим делать

- **Пример нетестируемого требования производительности ПО**
  - Время отклика системы должно находиться в приемлемых рамках
  - Время отклика (Отклика на какой операции?) системы (что такое система в этом требовании: UI, DB, client + server + network?) должно находиться (Условия? Нагрузка?) в приемлемых рамках (Цифры?)

Первое уточнение[33]

- **Пример тестопригодного требования**
  - Время отклика системы с точки зрения конечного пользователя (end-to-end) во время продуктивной нагрузки (50 пользовательских сессий в режиме «менеджер» / 15 пользовательских сессий в режиме «аналитик») при загруженности пропускного канала от клиентской системы до сервера приложений в пределах 50% для сети 100 Mb/sec и утилизации ресурсов сервера приложений (CPU, RAM) в рамках 70-80%, а клиентской машины в рамках 40-60%, не должно превышать 1 секунды для операций создания записи (сущности) и 3

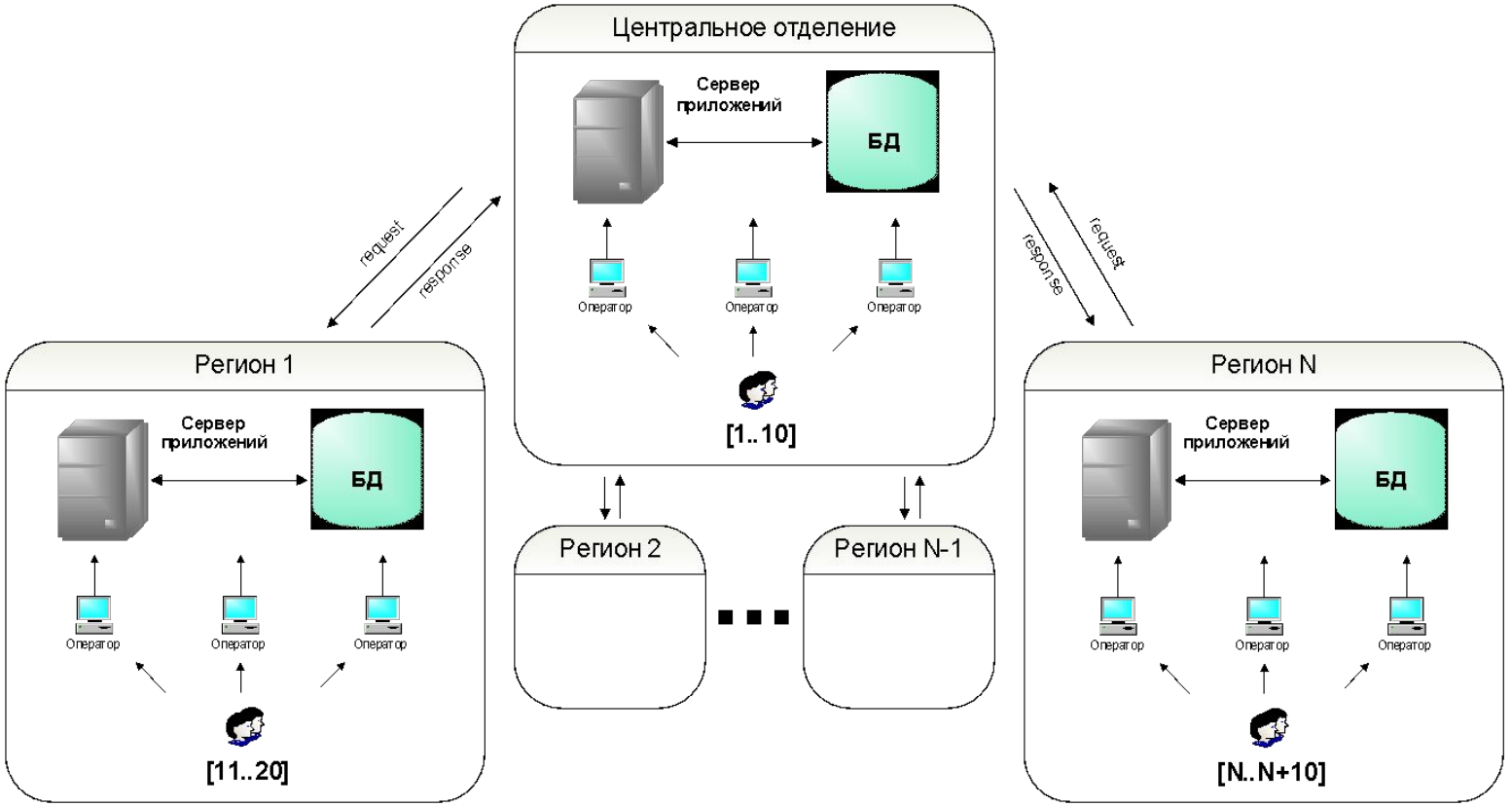
Ещё одно уточнение

- **Что мы упустили в требовании?**
  - Время отклика … при загруженности пропускного канала …, не должно превышать 1 секунды … время выполнения …
- **Что с ресурсами?.. Какими они должны быть?**

Что должно было получится

- **Пример тестопригодного требования**
  - Время отклика системы с точки зрения конечного пользователя (end-to-end) во время продуктивной нагрузки (50 пользовательских сессий в режиме «менеджер» / 15 пользовательских сессий в режиме «аналитик») при загруженности пропускного канала от клиентской системы до сервера приложений в пределах 50% для сети 100 Mb/sec, не должно превышать 1 секунды для операций создания записи и 3 секунд для операций поиска записи.
  - Время выполнения аналитических отчётов определяется отдельно для каждого отчёта.

Практический пример



Анализ архитектуры

Архитектура

- Сервер приложений
- Сервер БД
- «Толстые» клиенты, около 10 операторов

Первые выводы и вопросы

- Большинство операций происходит на стороне клиента
- Тестируем клиентскую часть и сервер приложений
- Сервер приложения может работать со своей БД и с БД центрального отделения
- БД не содержит никакой логики – только хранилище?

Анализ конфигурационных требований

Требования к конфигурациям

- Клиентская часть поддерживается на 4-х ОС
- Сервер приложения поддерживается на 2-х ОС
- Локализация – система поддерживает два языка
- На тестирование выносится 20 функциональных требований к клиентской части и 10 функциональных требований к серверной части

Пытаемся планировать

Вопросы к обсуждению

- Какие виды тестов будем проводить?
- Нагрузочного тестирования не будет, 10 операторов – это не та нагрузка, которую стоит проверять (или будет?)
- Что стоит автоматизировать, что нет?
- Какие окружения выделяем для тестирования?

Попробуем прикинуть трудоёмкость

Допущения

- Допустим, на одно функциональное требование мы предполагаем написать 5 тестовых сценариев
- Допустим, на прохождение 1-го тестового сценария мы предполагаем потратить 5 минут

Посчитайте сами и ответьте на следующие вопросы:

- Сколько всего окружений получается?
- Сколько всего тестовых сценариев будет в системе?
- Время затраченное на проведение 1-го

Считаем окружения

Окружений: 16

- 4 клиентские ОС * 2 языка = 8 клиентских конфигураций * 2 серверные ОС = **16 окружений**

Тестовых сценариев в системе: 150

- (20 клиентских требований + 10 серверных требований) * 5 тестовых сценариев на одно требование = **150**.

Сколько всего тестовых сценариев для проведения 1-го раунда тестирования?

Считаем время

Расчеты

Всего тестовых сценариев: 16 окружений * 150 тестовых сценариев = **2400**

Время на проведение 1-го раунда тестирования: (2400 тестовых сценарев * 5 минут) / 60 = **200 часов или 5 недель**

Давайте подумаем

Что мы не учли?

- Требования относятся к функциональности (логике приложения) или к окружению (системные функции, работа с ресурсами ОС и т.д.). Если к функциональности, то не надо проверять их в различных окружениях.

Разбор тестируемых функций

Что зависит *обычно* от окружения на клиенте и сервере?

- Вход, выход, печать форм, получение языка ОС, получение цветовой гаммы ОС, работа с протоколами общения между серверами приложений

Что не зависит от окружения?

- Получение информации из БД, запрос на сервер приложения, анализ полученных данных на клиенте и т.д.

Подбиваем баланс по группам требований

Получаем:

- 5 требований зависят от окружений на клиенте
- 5 требований зависят от всех окружений
- 5 требований зависят только от окружений сервера приложений
- 15 требований относятся к функциональности и не зависят от окружений

Пересчитываем

Итого:

- 25 тестовых сценариев * 8 = 200 тестовых сценариев зависящих от окружения на клиенте
- 25 тестовых сценариев * 16 = 400 тестовых сценариев зависящих от всех конфигураций
- 25 тестовых сценариев * 2 = 50 тестовых сценариев зависящих от окружения на сервере приложений
- 75 тестовых сценариев относятся к функциональным тестам

200 + 400 + 50 + 75 = **725 тестовых сценариев**

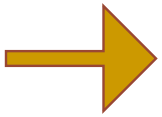
Сила тест-дизайна

Расчеты

- Всего тестовых сценариев: 725
- Время на проведение 1-го раунда тестирования: (725 тестовых сценариев * 5 минут) / 60 = 60,5 часов или 1,5 недели

Было: 200 часов или 5 недель

Стало: 60,5 часов или 1,5 недели



Экономия достигается за счёт принимаемых допущений и связанных с ними рисков

Автоматизация тестирования ПО

- Практические соображения
- Принимаем решение об автоматизации тестирования

Когда и как начинать?

Принятие решение об автоматизации

- Решение об автоматизации тестирования на проекте по разработке должно приниматься с учетом роста затрат на тестирование
- Внедрять Автоматизированное тестирование ПО стоит только при наличии полноценного рабочего процесса тестирования и нескольких успешных циклов (выпущенных версий) тестирования в проекте

О чём нужно помнить?

Принятие решение об автоматизации

- Нужно помнить, что первоначальные затраты на автоматизацию тестирования могут в 8-12 раз превосходить затраты на полноценное ручное тестирование
- При расчете эффективности автоматизации тестирования, требуется учитывать затраты на поддержку автоматизированных тестов (рекомендуется закладывать от 25% до 40% рабочего времени инженера автоматизации на поддержку и актуализацию тестовых сценариев и

Когда это полезно?

Критерии пригодности к автоматизации

- Внедрение автоматизации экономически оправдано
  - Проект по разработке долгосрочный (1-2 года) или находится в области mission/business critical проектов – потери от сбоев влекут невосполнимые финансовые потери или угрозу жизни.
  - Формальные методы расчета возврата вложений показывают положительный возврат средств

Когда это необходимо?

Критерии пригодности к автоматизации

- Внедрение автоматизации критично для успешной реализации проекта
  - Объем регрессионного тестирования в 4-5 раз превышает объемы тестирования новой функциональности (80/20 - критичное соотношение)
  - Ресурсов по тестированию, при правильном соблюдении баланса сил Dev/Test в проекте в обозримом будущем недостаточно, чтобы удерживать контроль над качеством системы

С чего начать? Цели и задачи

Что стоит автоматизировать вначале

- Автоматизация приёмочного тестирования билда или набора Smoke-тестов
- Автоматизация базовых операций системы, основных алгоритмов и устоявшейся функциональности
- Автоматизация инсталляции-развёртывания системы и создания базового набора тестовых данных

Идём или не идём?

Принятие решение об автоматизации

- Непосредственные выгоды от внедрения автоматизации достигаются за счет экономии времени при многократном прогоне тестов
- Решение внедрении инструментов автоматизации стоит принимать только на основе формальных оценок: метод ROI (возврат на вложения)

Тестирование производительности ПО

- Что понимают под нагрузочным тестированием
- Алгоритм создания скрипта

На что обычно наступают

- Непонимание терминологии заказчиком и проектной командой
  - Определите понятия «нагрузочное тестирование»
- Неготовность заказчика назвать «рамки успешности»:
  - Идите от бизнес-задач вместе с Заказчиком
  - Попробуйте использовать алгоритм «эксперимент-анализ-эксперимент» и думать в разрезе «стало лучше или хуже?»

На что обычно наступают-2

- Технологическая невозможность проектирования и воспроизведения адекватной модели нагрузки
  - Собирайте данные из эксплуатации
  - Ставьте эксперименты с разными моделями нагрузки
  - Пробуйте делать свои «пушки»
- Несогласованность ожиданий
  - Проговорите и зафиксируйте формат отчётов
  - Проговорите и зафиксируйте вероятность доверия к результатам

Слова: производительность и нагрузка

Тестирование производительности (performance testing)

- Нагрузочное тестирование, целью которого является определение показателей производительности системы при типовой нагрузке.

Тестирование производительности при максимальной нагрузке (load testing)

- Нагрузочное тестирование, целью которого является определение параметров производительности при **максимально прогнозируемой нагрузке** на систему, а также определение пределов производительности, т.е. максимального количества пользователей, которые одновременно смогут работать с системой без ошибок и сбоев.

Слова: стрессы и перегруз

Стрессовое тестирование (stress testing)

- Нагрузочное тестирование, целью которого является определение параметров производительности системы в случае ограниченности или недоступности ресурсов:
  - Оперативная память
  - Место на диске
  - Пропускная способность канала связи
  - Процессор

Слова: объёмы данных

Объемное тестирование (volume testing)

Нагрузочное тестирование, целью которого является определение параметров производительности системы в случае значительного изменения объемов данных.

Например:

- Для принтера – большое количество печатаемых страниц
- Для грамматического анализатора текста – большой объем текста, загруженного сразу

Слова: наработка на отказ

Тестирование надежности (reliability testing)

- Нагрузочное тестирование, целью которого является определение параметров производительности в условиях длительной эксплуатации, в том числе с большими объемами данных и при большой нагрузке на систему.

Слова: расширяемость и масштаб

Тестирование масштабируемости (scalability testing)

- Нагрузочное тестирование, целью которого является определение параметров производительности системы на различных аппаратно-программных платформах.
- Система называется масштабируемой, если она способна увеличивать производительность пропорционально дополнительным ресурсам

Алгоритм работы

Шаги

- Запускаем тул на запись и кликаем нашу последовательность.
- Логин и логофф части записываем в «vuser_init» и «vuser_end» части (при прогоне с различными параметрами, операции описанные в инит и енд части не проигрываются).
- Действия пользователя записываем в action часть, или создаём несколько actions, если по-логике выполняется более чем одно действие с осязаемым результатом.
- Смотрим что получилось и проигрываем: получаем воспроизведение действий от лица одного пользователя (*пока создания нагрузки нет*).

Алгоритм работы-2

Шаги

- При необходимости расставляем точки рандеву
- При необходимости определяем транзакции (что интересно померять)
- При необходимости/возможности параметризируем скрипт
- Выполняем калибровочный прогон на 3-5 пользователей
- Создаём модель нагрузки
- Выполняем прогон
- Коллекционируем и анализируем данные

Планирование и этапы тестирования

- Этапы тестирования
  - Что происходит во время планирования
- Практические соображения

Этапы тестирования

- Планирование тестирования
- Проектирование тестирования
- Реализация тестирования
- Выполнение тестирования
- Оценка тестирования

Цели планов

- **Планирование тестирования**
  - Выясняется
    - объем работ,
    - сроки выполнения,
    - пути решения задач по выполнению работ,
    - ресурсы,
    - календарный план
  - Основной артефакт – **план тестирования** >>

Задачи и содержание планов

- **Назначение плана тестирования**
  - План тестирования ПО является **основным согласующим документом**
  - Описывает:
    - Существующие проектные ограничения и артефакты, необходимые для анализа готовности к тестированию ПО
    - Основные подходы к решению поставленной задачи и необходимые ресурсы для ее выполнения

Что внутри плана

- **Что происходит при планировании**
  - Анализ имеющейся информации о проекте
  - Выявление компонент ПО, которые будут тестироваться
  - Проверка архитектуры системы на полноту и тестопригодность
  - Описание модели нагрузки, включая список эмулируемых операций и измеряемых параметров
  - Составление перечня инструментов и ресурсов, используемых в проекте
  - Составление перечня документов,

Из чего состоит план

- **Основные разделы плана тестирования**
  - Название, Проект
  - Журнал изменений
  - Введение
    - Назначение
    - Исходные данные
    - Область применения
    - Идентификация проекта
  - Тестовые требования
  - Стратегия тестирования
  - Ресурсы и расписание
  - Материалы, подлежащие сдаче

Стратегия тестирования

- **Стратегия тестирования**
  - Типы тестирования
    - Тестирование данных и целостности базы данных
    - Функциональное тестирование
    - Тестирование бизнес-циклов
    - Тестирование пользовательского интерфейса
    - Нагрузочное тестирование
    - Тестирование безопасности и управления доступом
    - Конфигурационное тестирование
    - Инсталляционное тестирование
  - Критерии завершённости
  - Инструментальные средства

Этапы тестирования 72

- **Практические соображения**
  - Не старайтесь создать тест-план в точности по шаблону
  - Попробуйте применить концепцию «живых документов»
  - Здравый смысл…
    - Здравый смысл!
      - Здравый смысл ☺

Работа с дефектами

- Алгоритм работы с дефектами
- Трекинг по статусам и ролям
- Типичные проблемы

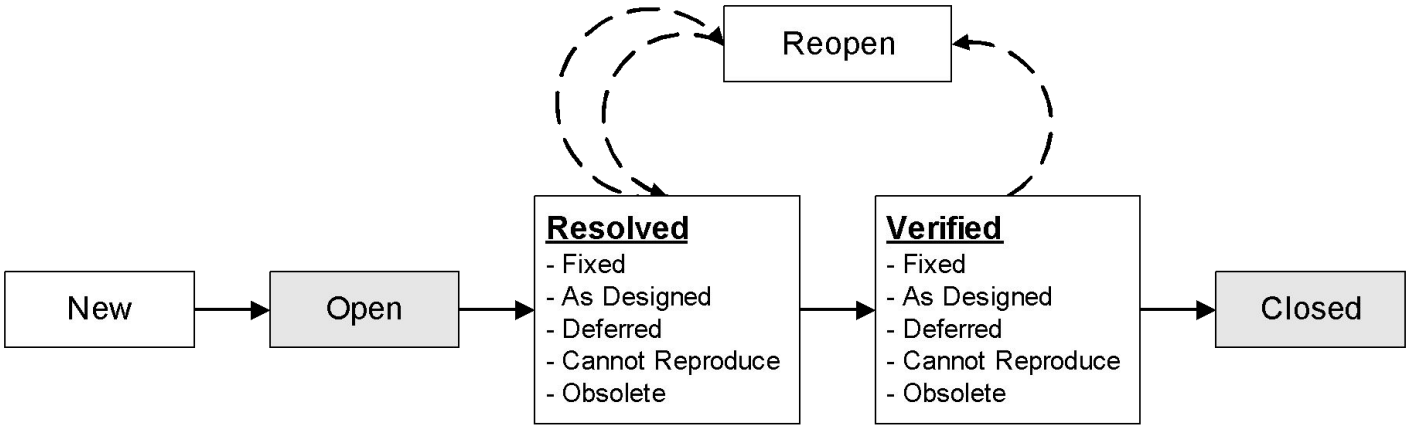
Универсальный алгоритм? Хм ☺️

Алгоритм работы с дефектами

- Алгоритм трекинга дефектов может отличаться от проекта к проекту
- Я предлагаю общую большую схему
- Если у вас это не так – посмотрите на что вы наступаете работая «как сейчас» и на что не будете если настроите как предлагается

Куколка-бабочка?

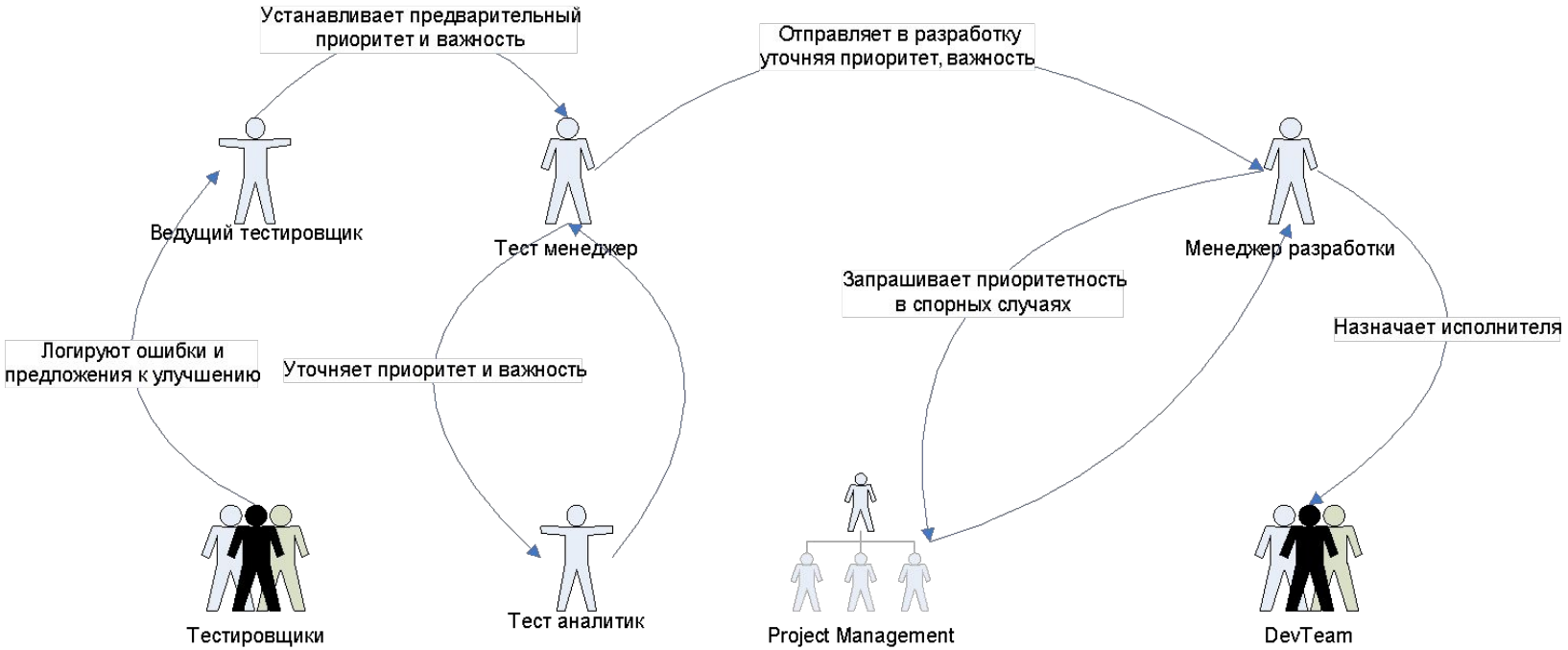
Трекинг по статусам



- Статусы ошибок, набор полей, их типы и свойства, возможные переходы между статусами определяются отдельно для каждого проекта

Кто кому: пас на тренера!

Трекинг по ролям



Что бывает не так?

Типичные проблемы

- Сокращение шагов по воспроизведению ошибки
- Отсутствие описания ошибочного поведения
- Отсутствие описания правильного поведения
- Неправильное использование статуса Reopen (для новых ошибок с похожими признаками)
- Неправильное использование статуса Duplicate

А ещё бывает и так...

- **Типичные проблемы**
  - …
  - Тестирование устаревшего билда
  - Изобретение собственных требований
  - Использование нечетких формулировок
  - Попытка определить причину ошибки (поставить диагноз)
  - Искусственное завышение приоритета ошибки
  - Самовольное сужение тестового

Работа с рисками

- Учет рисков, меры по их минимизации
- Процесс управления рисками
- Типичные риски
- Практические соображения

Работа с рисками: как начать?

- **Учет рисков, меры по их минимизации**
  - Риск – действующий или развивающийся фактор процесса, обладающий потенциалом негативного влияния на ход процесса
  - Риск – лишь потенциальный, а не обязательный результат
  - Опасен не сам риск, а его последствия

Работа с рисками: одной картинкой

Работа с рисками: зачем это всё?

Извлечение уроков

- Извлечение уроков формализует усвоение накопленного опыта в форме, доступной для использования как внутри группы, так и на уровне всей компании
- Даже если вы уверены, что все помнят усвоенные уроки, занесите информацию в базу знаний

Работа с рисками: главное начать

Процесс управления рисками

- Описанные фазы являются логическими шагами, и они не обязательно должны следовать в строгом хронологическом порядке
- Повторяйте шаги выявления-анализа-планирования по мере обнаружения дополнительных факторов, влияющих на проект
- Не все риски должны проходить через все описанные шаги

Работа с рисками: и ещё немного

Типичные риски

- Технические:
  - Нахождение критичных ошибок после ввода системы в эксплуатацию
  - Нахождение «дыр» в системе безопасности
  - Возникновение проблем при поставке и развертывании системы
  - Использование новых технологий
  - Использование продуктов третьих производителей
- >>

Работа с рисками: и ещё чуть-чуть!

Типичные риски

- …
- Планирование
  - Неправильное определение границ работ
  - Неправильный выбор архитектуры
  - Неправильная оценка ресурсов
- Организационные
  - Частое и противоречивое изменение требований заказчиком
  - Текучесть кадров

Работа с рисками в тестировании

Типичные риски в тестировании

- Нехватка ресурсов
- Требования будут формализованы поздно для полноценного тест-дизайна (не хватит времени)
- Критичные ошибки будут найдены поздно

Практические соображения

Типичные риски в тестировании

- Работа с рисками в тестировании не отличается от работы с рисками как таковой
- Отличаются типичные риски
  - Тестирование более зависимый процесс
- Б**о**льшая зависимость от заказчика

Практические соображения

Типичные риски в тестировании (пример)

- Большие и **НЕУЧТЁННЫЕ** затраты на перетестирование: проблемы планирования тестирования
- Использование группы тестирования как отладочного механизма: проблема с пониманием роли процесса тестирования в компании или проекте
- Поставка сырых или устаревших модулей: непоставленный процесс версионного контроля и конфигурационного менеджмента

Обзор ролей и артефактов в тестировании

ОСНОВНЫЕ РОЛИ

- Тест-менеджер, менеджер проекта по тестированию
- Тест-аналитик
- Тест-дизайнер
- Тестировщик, Инженер по тестированию

ВСПОМОГАТЕЛЬНЫЕ РОЛИ

- Администратор тестовой системы, приложений поддерживающих жизненный цикл тестирования
- Администратор баз данных, менеджер баз данных
- Разработчик тестов

Менеджер по тестированию не уборщица

Зачем нужно понимать какие роли куда относятся?

- Менеджеру нельзя заниматься вспомогательной деятельностью
- Специализация даёт прирост в эффективности: выделяйте роли в команде
- Специализаций и роли – основа перехода к процессному управлению
- Роли – первые «горизонтальные» ступеньки для роста ваших людей

Обзор артефактов тестирования (RUP)

- **Тестировщик ПО**
  - Test script
  - Test log
- **Аналитик**
  - Test Case
  - Test-Ideas List
  - Workload Analysis Model
  - Test Data
  - Test results

Обзор артефактов тестирования (RUP)

- **Дизайнер**
  - Test Strategy
  - Test Automation Architecture
  - Test Environment Configuration
  - Test Suite
- **Тест-менеджер**
  - Test Plan
  - Test Evaluation Summary

Мои артефакты: магия проста

- **Что я использую на практике:**
  - **Тест план:** что есть из артефактов, что будем тестировать и что НЕ будем тестировать, кто будет тестировать, какие типы тестов будем проводить, «тест лаб» или на чём будем тестировать, какими инструментами будем пользоваться, какие отчёты будем предоставлять, что будем сдавать заказчику
  - **Тест кейс:** «где я», что я делаю, что я ожидаю
  - **Чек-лист:** «когда много контрольных точек»
  - **Тестовая процедура:** «когда много шагов»

Активности тест-менеджера

- Круг задач
- Активности
- Список ответственности
- Совмещение ролей

Ваш круг задач

- Контроль объема проекта
- Планирование и контроль выполнения задач по тестированию
- Управление персоналом
- Контроль рисков, связанных с тестированием
- Мотивация персонала

Ваши активности

- Согласование миссии тестирования
- Получение обязательств по тестируемости продукта
- Оценка и отстаивание уровня качества
  - План развития качества продукта
- Оценка и улучшение производительности тестирования

Совмещение ролей

- Как правило, в небольших проектах совмещение ролей неизбежно. Это нормально
- Рассмотрим возможные сочетания ролей в проекте (следующий слайд)
  - ❑+ - сочетание допустимо
  - ❑± - сочетание нежелательно
  - ❑- - сочетание недопустимо
- >>

«Огурцы с молоком»

- Рассмотрим наиболее «порочное» сочетание ролей: разработчик-тестировщик
  - Разработчик проводит тестирование по остаточному принципу и не может найти неожиданные ошибки за счет «замыленности» взгляда.
  - Отсутствие или низкое качество регрессионного тестирования
  - Тестирование не является процессом с заданными входными и выходными данными и критериями завершения
  - Повышается вероятность пропуска или не нахождения критических дефектов

Исключения и связи

	Руководитель проекта	Разработка	Тестирование	Бизнес-анализ
Руководитель проекта		-	+	+
Разработка	-		-	-
Тестирование	+	-		±
Бизнес-анализ	+	-	±	

Успешное управление

100

- Работа с подчиненными
- Работа с руководством, коллегами, заказчиками
- Управление задачами
- Управление временем, тайм-менеджмент

С кем мы работаем?

Подбор исполнителей

- Именно руководитель составляет план и определяет количество нужных ресурсов
- Не соглашайтесь с урезанием ресурсов «безвозмездно», потребуйте либо увеличить бюджет, либо отпущенное время, либо усилить команду
- Согласие с «безвозмездным» урезанием ресурсов подрывает ваш авторитет
- >>

Плюс на минус даёт…

Подбор исполнителей

- Персональный состав команды согласовывается с руководством
- Руководитель может обоснованно отказаться работать с кем-либо из подчиненных
- Но если руководитель работает с подчиненными и не сообщает о проблемах, то это автоматически означает, что:
  - ❑ Это не отразится на качестве работы данного работника
  - ❑ Это не отразится на сроках выполнения работ
  - ❑ Это не отразится на качестве работы самого руководителя

Кто даёт оценки по времени?

Оценка задач

- Задачи должны оцениваться теми, кто их будет выполнять
- Проводите оценку задач совместно с подчиненными. При этом помните, что:
  - Исполнители всегда называют неверные сроки
  - Понять, насколько ошибается конкретный исполнитель можно лишь методом проб и ошибок
  - «Силовые» методы как средство борьбы с неправильными оценками не работают

Трудоёмкость задач: keywords

Оценка трудоемкости задач: способы

- **Evidence based**
- **Дельфи**
- **COCOMO**

Мысли вслух

Практические соображения

- Помните о правиле «треугольника» **Время-Деньги-Ресурсы**. Фиксировать можно не более 2 параметров
- Если какой-либо параметр меняется более, чем на 10%, то воздействие на другие два будет **квадратичным**
- Менеджер вносит на порядок б*о*льшие ошибки в оценки проекта, чем инженеры при оценке индивидуальных задач

О чём часто забывают

Оценка задач. Практические соображения

- При оценке не забывайте о «скрытых» задачах и «запасе прочности»
- При составлении планов и оценке учитывайте риски
  - *...слабо развиты наши методы оценок. В сущности, они отражают молчаливое и совершенно неверное предположение, что все будет идти хорошо.*
    Фредерик Брукс
- **Как правило, план не выполняется из-за неучтенных задач или рисков**

Все люди – разные

Индивидуальный подход

- Существует множество способов классификации психологических типов.
- Предлагаем вам следующую классификацию (с точки зрения отношения к работе):
  - «Пунктуальный». Не пренебрегает деталями, но всегда занят и не способен расставить приоритеты
  - «Хочу понравиться всем». Работает старательно и добросовестно, но не умеет говорить «нет» и часто перегружен чужой работой
  - >>

Все люди – очень разные ☺️

Индивидуальный подход

- ❑ …
- ❑ «Всегда опаздываю». Способен справиться с большим объемом работ в краткие сроки, но всегда откладывает ее до последнего – иначе ему неинтересно
- ❑ «Самодостаточный». Ответственен, но избегает обращаться за помощью и советами, из-за чего часто затягивает даже простые задачи
- ❑ «Творец». Способен найти нетривиальный подход, эффективно решать сложные задачи, но часто игнорирует планы и бизнес-потребности, избегает рутинной работы

- Учитывайте индивидуальность при постановке задач!

Что мы забываем при планировании?

Практические соображения

- Подстраивайте график работы группы под нужды проекта
- Учитывайте разницу во времени между офисами и заказчиком
- Учитывайте культурные различия

Коммуникации и встречи

Проводите ежедневные собрания группы

- Выявляйте текущие проблемы, но не пытайтесь их решать
- Не допускайте длинных неконструктивных дискуссий и обсуждения личностей
- Помните, ежедневные собрания имеют также и мотивационную функцию (о твоих проблемах знают, твои проблемы решаются)

Контроль и информация

Контроль

- Будьте в курсе, чем занят каждый подчиненный и когда он окончит свое задание
  - Планируйте загрузку с избытком («горизонт задач»)
- Если вы узнаете о плохой работе подчиненных от руководства – это тревожный симптом!

Информация

- Обеспечьте совместное использование (sharing) имеющегося опыта
- Избегайте возникновения «Тайных Знаний» (Secret Knowledge)
- Поддерживайте создание и развитие Базы Знаний

Мотивация

- Кто может мотивировать
- Виды мотивации
- Иерархия потребностей
- Практические соображения

Кто мотивирует?

Кто должен мотивировать

- Как правило, в тестировании задачи по мотивации исполняет тест-менеджер
- **В этом контексте** его обязательные признаки:
  - Право подбора исполнителей – я тебя выбрал, а не тебя назначили в мой проект
  - Право распределения премий – я определяю размер твоей премии
  - Право запрета выпуска версии – я не просто смотрю, я управляю проектом и качеством Продукта

Как мотивировать?

Типичная схема мотивации

- **Материальная**
  - Ставка
  - Премия по результатам личной деятельности
  - Премия по результатам деятельности отдела
  - …
- **Нематериальная**
  - Личное общение
    - Внимание и защита
    - Сопричастность
    - Влияние

Иерархия потребностей (по Маслоу)

Что это означает на практике?

Мотивирующие факторы для различных уровней (по Маслоу)

5. Клубы (кружки) качества, SEPG, подарки при значимых событиях...

4. Название должности, представительские обязанности, имидж компании, аксессуары...

3. Участие в управлении, обратная связь с руководством, обучение, информация о долгосрочных перспективах...

2. Конкурентная з/п, комфортное рабочее место и т.п.

1. Выполнение 80% требуемого объема работы оплачивается на уровне средней з/п на рынке

Чего от нас ждут, как от менеджеров?

Исходя из того, что мы люди здравые ☺️

- Что нужно руководству компании?
- Что нужно вашему заказчику?
- Что нужно вашему ПМ-у?
- Что нужно вашему ведущему разработчику?

Управление задачами и временем

- Модель F.A.C.T.
- Разгребая завалы: «GTD»

Модель FACT

Распределите все задачи по срочности и важности

	Менее важно	Важно
Менее срочно	**T**	**C**
Срочно	**F**	**A**

Выполняйте задачи в такой последовательности:

- **A** – срочные и важные задачи
- **C** – несрочные, но важные задачи
- **F** – неважные, но срочные задачи
- **T** – неважные и несрочные задачи

Когда её делать… или не делать?

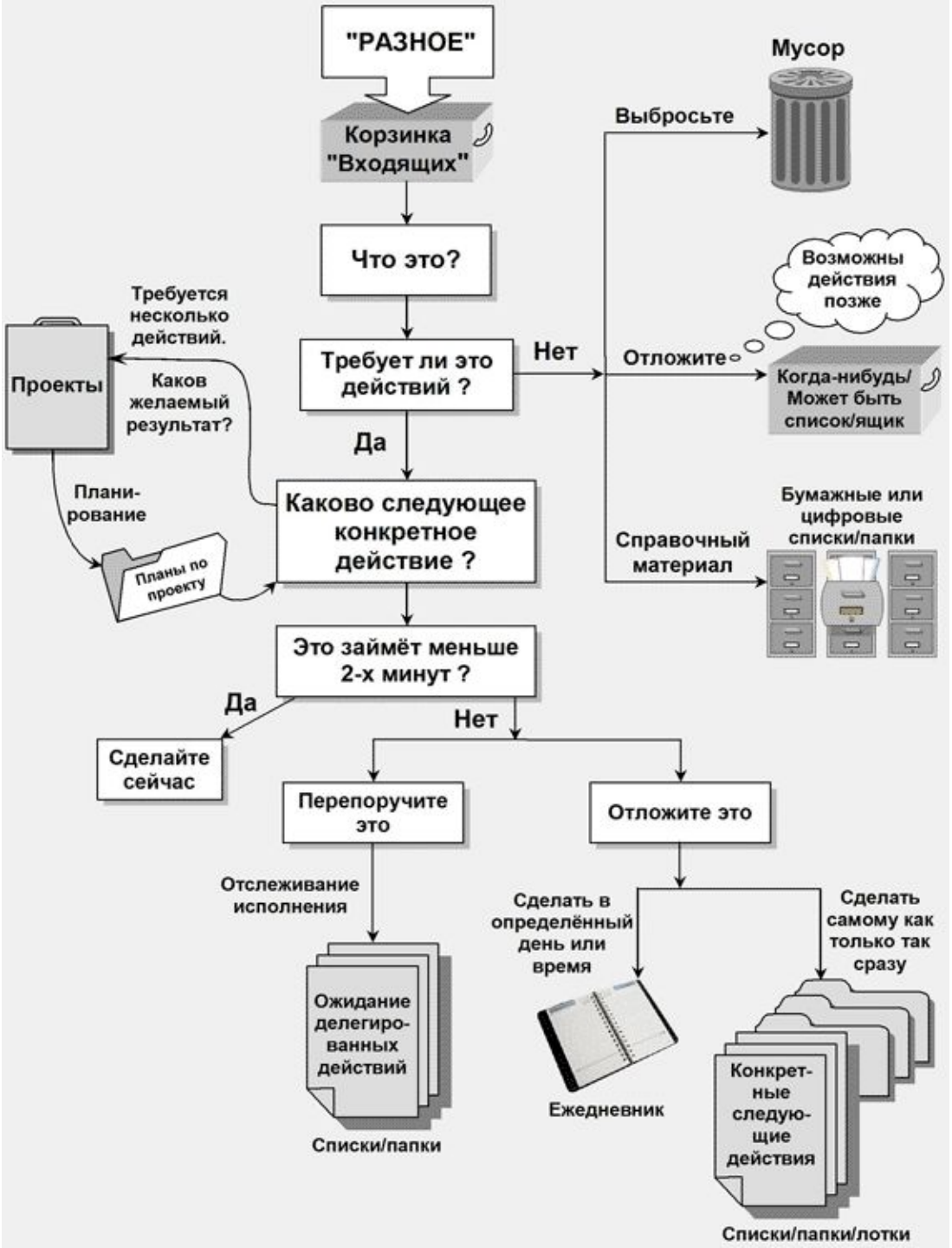
Принципы планирования рабочего дня

- При классификации задач задайте себе вопросы:
  - Это связано с моими целями? *(при ответе «да» - тип A,C)*
  - Только ли я смогу это сделать? *(при ответе «да» - тип A,C)*
  - Это должно быть сделано сегодня? *(при ответе «да» - тип F,C)*
- Прежде, чем отнести задачу к типу T, подумайте, не стоит ли ее делегировать

Какие задачи и когда делать?

Принципы планирования рабочего дня

- Установите «коммуникационные окна»
- Установите время, когда вы отвечаете лишь на экстренные вопросы
- Попытайтесь не выделять специально времени для задач типа Т, а использовать их как «песок»


"РАЗНОЕ"
Корзинка "Входящих"
Что это?
Требует ли это действий ?
Нет
Выбросьте
Мусор
Отложите
Возможны действия позже
Когда-нибудь/ Может быть список/ящик
Да
Требуется несколько действий.
Проекты
Каков желаемый результат?
Плани-рование
Планы по проекту
Каково следующее конкретное действие ?
Справочный материал
Бумажные или цифровые списки/папки
Это займёт меньше 2-х минут ?
Да
Сделайте сейчас
Нет
Перепоручите это
Отложите это
Отслеживание исполнения
Ожидание делегиро-ванных действий
Списки/папки
Сделать в определённый день или время
Ежедневник
Сделать самому как только так сразу
Конкрет-ные следую-щие действия
Списки/папки/лотки

Как заставить себя умываться по утрам

123

Основные принципы

- Чтобы начать управлять временем требуются первоначальные затраты
- Ведите хронометраж рабочего дня. Попробуйте продержаться хотя бы 1-2 недели ☺️
- Фиксируйте запланированные и незапланированные дела, помехи, эмоциональное и физическое состояние
- Не старайтесь угодить всем. Берегитесь «хронофагов»
  - Пожиратели времени, люди которые постоянно спрашивают тебя, не затрудняясь поискать ответ самостоятельно
- Имейте список целей. Без этого невозможно составлять планы, а без плана невозможно управлять временем
- Научитесь говорить «нет» и делегировать задачи

Помогать не значит делать за кого-то

Делегирование задач

- Почему зачастую руководитель намного более занят, чем подчиненный?
- Если вы помогаете решать проблему вашего подчиненного, она не становится вашей
- Избегайте «размывания» ответственности
- Избегайте коллективной ответственности
- Учитывайте индивидуальность

Рекомендуемая литература

- **Автоматизированное тестирование программного обеспечения**
- **Элфрид Дастин, Джефф Рэшка, Джон Пол**
- Automated Software Testing
- Издательство: Лори, 2003 г.
- Исчерпывающее пошаговое руководство по использованию наиболее эффективных инструментальных средств, приемов и методов автоматизированного тестирования. Основываясь на анализе многочисленных примеров успешных отраслевых реализаций, эта книга предоставляет все, что необходимо знать для успешного внедрения автоматизированного тестирования в процесс разработки.

Рекомендуемая литература

- **Ключевые процессы тестирования. Планирование, подготовка, проведение, совершенствование**
- Рекс Блэк
- Издательство: Лори, 2006 г.
- В этой книге Рекс Блэк выделяет двенадцать процессов тестирования, являющихся ключевыми для достижения успеха. За описанием каждого из этих процессов следует пример использования процесса. Вместо громоздких правил представлены списки контрольных вопросов - легкие, гибкие инструменты для внедрения тестирования, ориентированного на процесс, для сбора измерений и внесения последовательных изменений.

Рекомендуемая литература

- **Тестирование производительности Web-приложений Microsoft. NET (+ CD-ROM)**
- Performance Testing Microsoft. NET Web Application
- Издательство: Русская Редакция, 2003 г.
- Эта книга написана группой специалистов Microsoft, протестировавших и настроивших сотни Web-сайтов и Web-приложений, - Microsoft Application Consulting and Engineering (ACE) Team (группа консалтинга и проектирования приложений Microsoft).

Рекомендуемая литература

- **Путь камикадзе**
- Эдвард Йордон
- Издательство: Лори, 2003 г.
- Книга Эдварда Йордона `Путь камикадзе` представляет собой полное руководство по выживанию в безнадежных проектах, предназначенное для разработчиков программного обеспечения

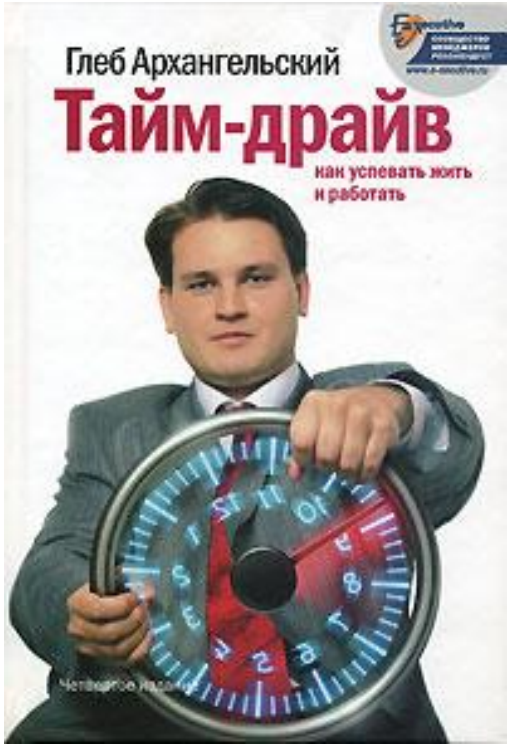
Рекомендуемая литература

- **Мифический человеко-месяц**
- Фредерик Брукс
- Издательство: Символ-Плюс, 2006 г.
- Эта книга – своего рода библия для разработчиков программного обеспечения во всем мире, написанная Бруксом еще в 1975 году. Полагают, что без прочтения книги Брукса не может состояться ни один крупный руководитель программного проекта

Рекомендуемая литература

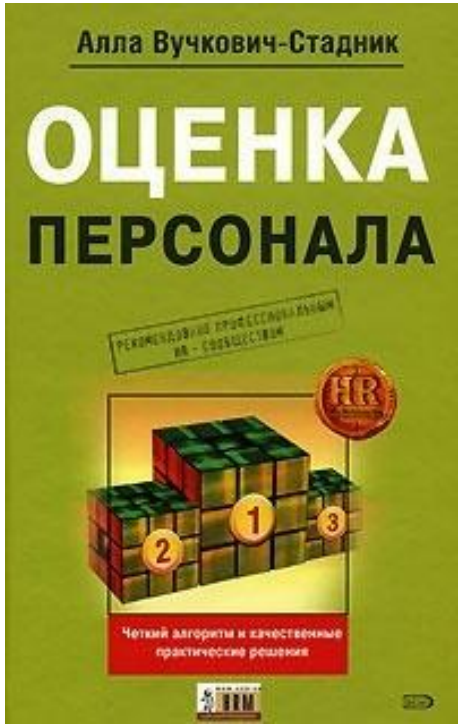
- **Тестирование dot com**
- Роман Савин
- Издательство: Дело, 2007 г.
- Этот курс лекций создан для тех, кто хочет обучиться тестированию, понять, как вести себя в корпоративном окружении, и добиться профессионального и личностного роста. Он будет интересен и участникам процесса разработки программного обеспечения, рекрутерам, людям, связанным с интернетом или пишущим о нем, и просто всем желающим понять кухню интернет-стартапов

Рекомендуемая литература



- **Тайм-драйв**
- Глеб Архангельский
- Издательство: Манн, 2007 г.
- Самая полезная и увлекательная книга об управлении временем. Книга отличается популярным изложением. В максимально простой и пошаговой форме, на реальных российских примерах, она дает ответ на главный вопрос современного менеджера: как успевать больше?

Рекомендуемая литература



- **Оценка персонала. Четкий алгоритм действий и качественные практические решения**
- Алла Вучкович-Стадник
- Издательство: Эксмо, 2008 г.
- Вы сможете грамотно выстроить систему оценки своего персонала, а ценный практический опыт автора позволит избежать многих ошибок в процессе ее внедрения.

Рекомендуемая литература

- **Грейдинг. Технология построения системы управление персоналом**
- Валерий Чемеков
- Издательство: Вершина, 2007 г.
- В предлагаемой вниманию читателей книге автор рассматривает системный подход к кадровым технологиям (грейдингу, оценке результативности, аттестации), который основан на измерении и описании должностей. Полученные данные позволяют формулировать требованиядля поиска кандидата, проведения аттестации, обучения, построения системы тарификации.

Рекомендуемые онлайн-источники

- QAforums.com – Software Testing and Quality Assurance Online Forums. English.
- StickyMinds.com – brain food for building better software. English.
- Braidy TesterBraidy Tester (http://blogs.msdn.com/micahel)Braidy Tester (http://blogs.msdn.com/micahel) – Making developers cry since 1995 ☺
- it4business.ru – информационный канал для IT-менеджера.