

# ТЕСТИРОВАНИЕ

ПРОГРАММНОГО  
ОБЕСПЕЧЕНИЯ

# ТЕСТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

- ✓ Поиск ошибок
- ✓ Проверка соответствия ПО требованиям и здравому смыслу
- ✓ Оценка работоспособности ПО
- ✓ Способ контролировать качество ПО

**Тестирование программного обеспечения** — процесс анализа программного средства и сопутствующей документации с целью выявления дефектов и повышения качества продукта.

В глоссарии **ISTQB\*** нет термина «тестирование ПО», который широко используется в русском языке. Там есть лишь термин «тестирование (**testing**)».



**Тестирование (testing)** — процесс, содержащий в себе все активности жизненного цикла, как динамические, так и статические, касающиеся планирования, подготовки и оценки программного продукта и связанных с этим результатов работ с целью определить, что они соответствуют описанным требованиям, показать, что они подходят для заявленных целей и для определения дефектов.

\* **ISTQB** — International Software Testing Qualifications Board Glossary

# ПРИЧИНЫ ОШИБОК В ПО

- ✓ Человеческий фактор
- ✓ Проблемы в описании требований к программному обеспечению
- ✓ Недостаток времени
- ✓ Недостаточно продуманная архитектура приложения
- ✓ Недостаточное знание бизнеса
- ✓ Нехватка профессиональных навыков и опыта
- ✓ Изменения «в последнюю минуту»

# ИСТОРИЯ ТЕСТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

50–60-е годы  
прошлого века

Первые программные системы разрабатывались в рамках программ научных исследований или программ для нужд министерств обороны.

Тестирование таких продуктов проводилось строго формализовано с записью всех тестовых процедур, тестовых данных, полученных результатов.

Тестирование выделялось в отдельный процесс, который начинался после завершения кодирования, но при этом, как правило, выполнялось тем же персоналом.

Фактически тестирование представляло собой скорее отладку программ (*debugging*).

# ИСТОРИЯ ТЕСТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

В 1960-х много внимания уделялось «исчерпывающему» тестированию, которое должно проводиться с использованием всех путей в коде или всех возможных входных данных.

Однако это невозможно:

- количество возможных входных данных очень велико;
- существует множество путей;
- сложно найти проблемы в архитектуре и спецификациях.

Итог: «исчерпывающее» тестирование было отклонено и признано теоретически невозможным.

# ИСТОРИЯ ТЕСТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

В 1970-х годах фактически родились две фундаментальные идеи тестирования:

- ✓ тестирование сначала рассматривалось как **процесс доказательства работоспособности** программы в некоторых заданных условиях (*positive testing*), а затем:
- ✓ как **процесс доказательства неработоспособности** программы в некоторых заданных условиях (*negative testing*)

Бытует неверное понимание того, что негативные тест-кейсы должны заканчиваться возникновением сбоев и отказов в приложении. Нет, это не так. Ожидаемым результатом негативных тест-кейсов является именно корректное поведение приложения, а сами **негативные тест-кейсы считаются пройденными успешно, если им не удалось «поломать» приложение.**

# ИСТОРИЯ ТЕСТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

В 80-х годах произошло ключевое изменение места тестирования в разработке ПО:

- ✓ вместо одной из финальных стадий создания проекта **тестирование стало применяться на протяжении всего цикла разработки**, что позволило не только быстро обнаруживать и устранять проблемы, но даже предсказывать и предотвращать их появление.

В этот же период времени отмечено бурное **развитие и формализация методологий тестирования** и появление первых элементарных попыток автоматизировать тестирование.

В ходе тестирования надо проверить не только собранную программу, но и требования, код, архитектуру, сами тесты.

# ИСТОРИЯ ТЕСТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

В 90-х годах произошёл переход от тестирования как такового к более всеобъемлющему процессу, который называется «обеспечение качества (quality assurance)». В понятие «тестирование» стали включать планирование, проектирование, создание, поддержку и выполнение тест-кейсов и тестовых окружений.

Начинают появляться различные программные инструменты для поддержки процесса тестирования: более продвинутые среды для автоматизации с возможностью создания скриптов и генерации отчетов, системы управления тестами, ПО для проведения нагрузочного тестирования.



# ИСТОРИЯ ТЕСТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

С нулевых годов появляются гибкие методологии разработки и такие подходы, как «разработка под управлением тестированием (*test-driven development, TDD*)».

У тестировщиков появляются специализированные технологии и инструменты, а сам процесс тестирования интегрирован в цикл разработки программного обеспечения.

Популярны идеи о том, что во главу процесса тестирования следует ставить не соответствие программы требованиям, а её способность предоставить конечному пользователю возможность эффективно решать свои задачи.

# ТЕСТИРОВАНИЕМ ЗАНИМАЮТСЯ

- ✓ заказчики
- ✓ разработчики
- ✓ тестировщики
- ✓ бизнес аналитики
- ✓ команда технической поддержки продукта
- ✓ дизайнеры
- ✓ конечные пользователи

## Кто такой тестировщик

В ЕКСД\* РБ выделены 2 должности :

- ✓ специалист по тестированию программного обеспечения (**QA-тестировщик**);
- ✓ тестировщик программного обеспечения (**QA-инженер**);

**QA-Тестировщик** — это специалист, который находит ошибки (баги) в работе программного обеспечения (ПО) во время его тестирования, чтобы повысить качество продукта.

В его обязанности входит:

1. Проверяют соответствие ПО заданным требованиям
2. Фиксируют ошибки
3. Создают тестовые планы.

**QA-инженер** — это специалист, который тестирует и контролирует качество продукта на всех этапах его создания, который дает рекомендации по улучшению продукта и повышению качества разработки. Обязанности:

1. Изучают продукт
2. Вносят замечания на начальных этапах тестирования
3. Определяют, какие нужно провести тесты
4. Ставят сроки начала тестирования
5. Координируют взаимодействие тестировщиков и разработчиков
6. Следят, чтобы обо всех ошибках, которые обнаружились на разных этапах разработки и тестирования, узнали нужные люди
7. Контролируют сроки решения проблем
8. Ставят приоритеты в работе.

\* ЕКСД – единый классификационный справочник должностей



*Результатом работы тестировщика является **счастье** (удовлетворение) **конечного пользователя**. Причем "счастье" не в глобальном его значении, а та его часть, которая связана с качеством вашего продукта.*

Роман Савин  
Книга: тестирование dot com или Пособие по жестокому обращению с багами в интернет-стартапах

**ТЕСТИРОВАНИЕ**

---

**КОНТРОЛЬ И ОБЕСПЕЧЕНИЕ КАЧЕСТВА**

**Обеспечение качества (Quality Assurance - QA)** - это совокупность мероприятий, охватывающих все технологические этапы разработки, выпуска и эксплуатации программного обеспечения информационных систем, предпринимаемых на разных стадиях жизненного цикла ПО, для обеспечения требуемого уровня качества выпускаемого продукта.

**Контроль качества (Quality Control - QC)** - это совокупность действий, проводимых над продуктом в процессе разработки, для получения информации о его актуальном состоянии в разрезах: "готовность продукта к выпуску", "соответствие зафиксированным требованиям", "соответствие заявленному уровню качества продукта".

**Тестирование программного обеспечения (Software Testing)** - это одна из техник контроля качества, включающая в себя активности по планированию работ (Test Management), проектированию тестов (Test Design), выполнению тестирования (Test Execution) и анализу полученных результатов (Test Analysis).

# В чем отличия QA от тестирования

- **QA** — это забота о качестве в виде превентирования появления багов
- **Тестирование** — это забота о качестве в виде обнаружения багов до того, как их найдут пользователи.

Пример о воспитании из лекции

Общее в QA и тестировании заключается в том, что они призваны улучшить ПО, различие между ними — в том, что:

- QA призвано улучшить ПО через улучшение процесса разработки ПО;
- тестирование — через обнаружение багов.

# Что такое баг?



Баг (*bug*) — это отклонение фактического результата (*actual result*) от ожидаемого результата (*expected result*).

Баг живет лишь при **одновременном** выполнении всех трех условий:

1. **Известен** фактический результат;
2. **Известен** ожидаемый результат;
3. **Известно**, что результат из пункта 1 не равен результату из пункта 2.



# Источники ожидаемого результата

- ✓ **Спецификация ! ! !**
- ✓ жизненный опыт
- ✓ здравый смысл
- ✓ устоявшиеся стандарты
- ✓ статистические данные
- ✓ авторитетное мнение и др.

## **Статистические данные:**

Время ожидания открытия веб-страницы, после которого пользователь уходит

!!! В большинстве случаев баг — это отклонение от спецификации

## **Здравый смысл**

Пример: по спецификации пользователь может загрузить лишь одну фотографию за раз. А что, если у него таких фотографий 200? Будет он счастлив? Что делаем? Пишем *Feature Request* ("запрос об улучшении").

## **УСТОЯВШИЕСЯ СТАНДАРТЫ**

Как правило, после регистрации, пользователь должен получить е-мейл с подтверждением. Если спек не упоминает о таком е-мейле, вы можете потребовать дополнить его на основании сложившейся практики.

## Пример

Пункт 19.а спецификации #8724 "О регистрации нового пользователя" устанавливает:

«Поле "Имя" должно быть обязательным. Страница с ошибкой должна быть показана, если пользователь посылает регистрационную форму без заполнения указанного поля».

Функциональный баг



В общем все просто:

- тестировщик идет на страничку с регистрационной формой;
- кликает линк "Регистрация";
- заполняет все обязательные поля, **кроме поля "Имя"**;
- нажимает на кнопку "Зарегистрироваться".

Если ошибка не показана и регистрация подтверждается, то нужно рапортовать баг (file a bug).

Баг в спецификации

**! НО.** Непонятно, каким должно быть сообщение об ошибке **!**  
Программист может написать любое на своё усмотрение.

# КАЧЕСТВО СИСТЕМЫ

Это степень удовлетворения системой заявленных и подразумеваемых потребностей различных заинтересованных сторон, которая позволяет, таким образом, оценить достоинства.

Эти заявленные и подразумеваемые потребности представлены в международных стандартах серии SQuaRE посредством моделей качества, которые представляют качество продукта в виде разбивки на классы характеристик.

Стандарт качества ПО ISO/IEC 25010:2011



НАЦИОНАЛЬНЫЙ  
СТАНДАРТ  
РОССИЙСКОЙ  
ФЕДЕРАЦИИ

ГОСТ Р ИСО/МЭК  
25010—  
2015

**Информационные технологии**

**СИСТЕМНАЯ И ПРОГРАММНАЯ ИНЖЕНЕРИЯ**

**Требования и оценка качества систем  
и программного обеспечения (SQuaRE).**

**Модели качества систем и программных продуктов**

ISO/IEC 25010:2011

Systems and software engineering — Systems and software  
Quality Requirements and Evaluation (SQuaRE) — System and  
software quality models  
(IDT)

**Содержание**

1	Область применения .....
2	Соответствие .....
3	Основы модели качества .....
3.1	Модели качества .....
3.2	Модель качества при использовании .....
3.3	Модель качества продукта .....
3.4	Цели моделей качества .....
3.5	Применение модели качества .....
3.6	Качество с точки зрения различных заинтересованных .....
3.7	Взаимосвязь моделей .....
4	Термины и определения .....
4.1	Термины к модели качества при использовании .....
4.2	Термины к модели качества продукта .....
4.3	Общие определения .....
4.4	Термины и определения из ИСО/МЭК 25000 .....
	Приложение А (справочное) Сравнение с моделью качества .....
	Приложение В (справочное) Пример отображения функциона .....
	Приложение С (справочное) Использование модели качества .....
	Библиография .....

# МОДЕЛИ КАЧЕСТВА

К настоящему времени в серии SQuaRE имеются три модели качества:

- ✓ Модель качества при использовании
- ✓ Модель качества продукта
- ✓ Модель качества данных, определенная в ИСО/МЭК 25012

Совместное использование моделей качества дает основание считать, что учтены все характеристики качества.

# МОДЕЛЬ КАЧЕСТВА ПРИ ИСПОЛЬЗОВАНИИ

- ✓ Эффективность (результативность)
- ✓ Производительность
- ✓ Удовлетворенность (полноценность, доверие, удовольствие, комфорт)
- ✓ Свобода от риска (смягчение отрицательных последствий):
  - экономического риска;
  - риска для здоровья и безопасности;
  - экологического риска.
- ✓ Покрытие контекста (полнота контекста, гибкость)

# Определение характеристик моделей качества подробно дано в ГОСТ Р ИСО /МЭК 25010–2015

Например:

**4.1.1 эффективность, результативность** (effectiveness): Точность и полнота, с которой пользователи достигают определенных целей (ИСО 9241-11).

**4.1.2 эффективность, производительность** (efficiency): Связь точности и полноты достижения пользователями целей с израсходованными ресурсами (ИСО 9241-11).

Примечание - Соответствующие ресурсы могут включать в себя время выполнения задачи (человеческие ресурсы), материалы или финансовые затраты на использование.

**4.1.3 удовлетворенность** (satisfaction): Способность продукта или системы удовлетворить требованиям пользователя в заданном контексте использования.

**4.1.3.1 полноценность** (usefulness): Степень удовлетворенности пользователя достижением прагматических целей, включая результаты использования и последствия использования.

**4.1.3.2 доверие** (trust): Степень уверенности пользователя или другого заинтересованного лица в том, что продукт или система будут выполнять свои функции так, как это предполагалось.

**4.1.3.3 удовольствие** (pleasure): Степень удовольствия пользователя от удовлетворения персональных требований.

**4.1.3.4 комфорт** (comfort): Степень удовлетворенности пользователя физическим комфортом.

И т.д. ....

# МОДЕЛЬ КАЧЕСТВА ПРОДУКТА

## ✓ Функциональная пригодность:

- функциональная полнота;
- функциональная корректность;
- функциональная целесообразность.

## ✓ Уровень производительности:

- временные характеристики;
- использование ресурсов;
- потенциальные возможности.

## ✓ Совместимость:

- сосуществование;
- интероперабельность.



# МОДЕЛЬ КАЧЕСТВА ПРОДУКТА

## ✓ Удобство использования:

- определимость пригодности;
- изучаемость;
- управляемость;
- защищенность от ошибки пользователя;
- эстетика пользовательского интерфейса
- доступность.

# МОДЕЛЬ КАЧЕСТВА ПРОДУКТА

## ✓ Надежность:

- завершенность;
- готовность;
- отказоустойчивость;
- восстанавливаемость.

## ✓ Защищенность:

- конфиденциальность;
- целостность;
- неподдельность;
- отслеживаемость;
- подлинность.

# МОДЕЛЬ КАЧЕСТВА ПРОДУКТА

## ✓ Сопровождаемость:

- модульность;
- возможность многократного использования;
- анализируемость;
- модифицируемость;
- тестируемость.

## ✓ Переносимость:

- адаптируемость;
- устанавливаемость;
- взаимозаменяемость.

# Показатели надёжности ПО

На практике различают несколько типов показателей надёжности ПО:

- ✓ количество ошибок перед отладкой и после неё;
- ✓ наработка часов на отказ;
- ✓ интенсивность отказов;
- ✓ вероятность безотказного действия в течение заданного отрезка времени.

Показатели оцениваются как количественные, качественные и порядковые. Наиболее ценную информацию дают количественные. Их получают путем непосредственных наблюдений и обработки результатов испытаний систем.