SQA DAYS #29
sqadays.com

29-30 ОКТЯБРЯ
МОСКВА

SOFTWARE QUALITY ASSURANCE DAYS

Семь принципов тестирования:
почему они так важны и как ими пользоваться

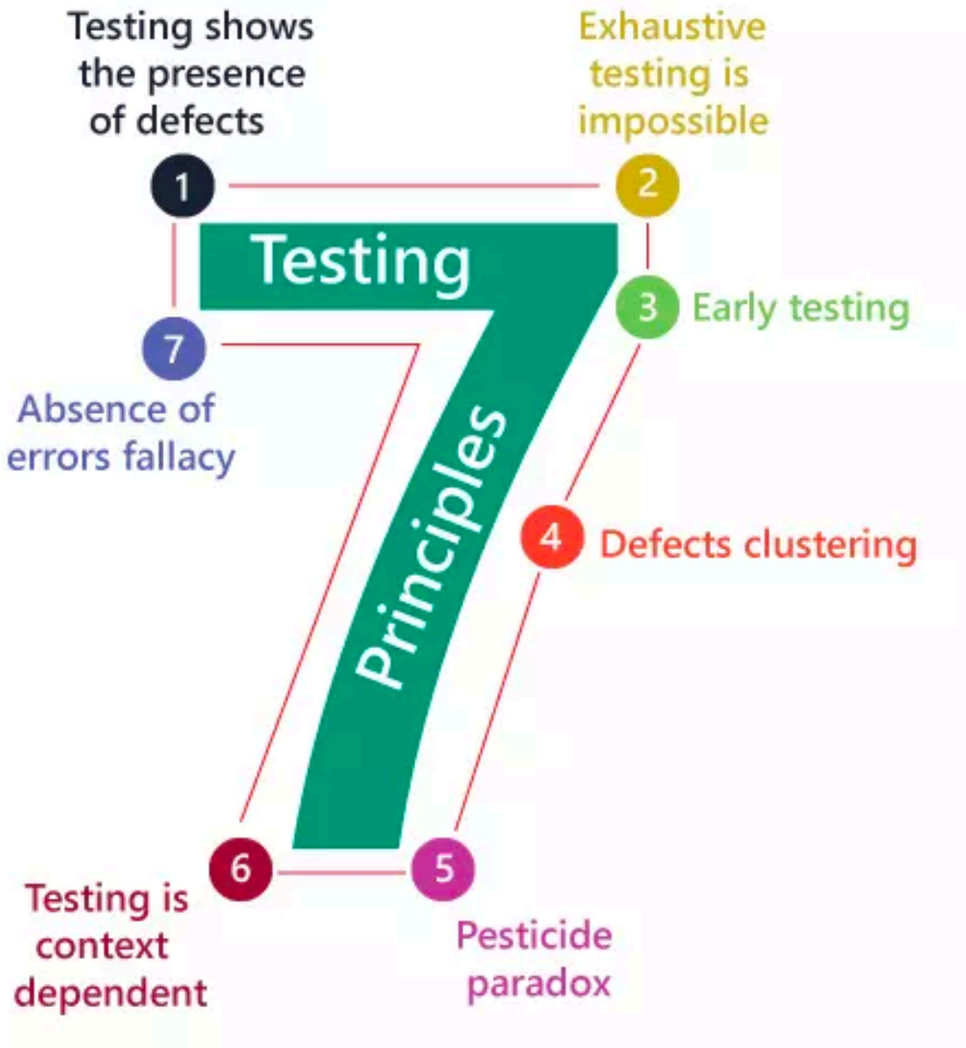
Сахаров Вячеслав

Customertimes, Киев, Украина

Обо мне:

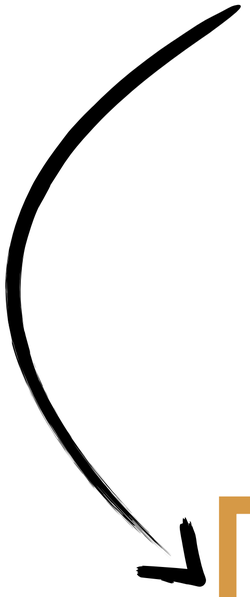
- Менеджер по релизам в Customertimes
- Более 9 лет опыта работы, первые шесть из которых — как QA специалист
- 5-летний опыт преподавания, коучинга и участия в общественных мероприятиях, таких как конференции и семинары
- Опыт управления и постановки процессов в больших корпоративных проектах с распределенными командами
- Сертифицированный тест-менеджер ISTQB

Краткая история появления понятия



Принцип 1 — Тестирование демонстрирует наличие дефектов

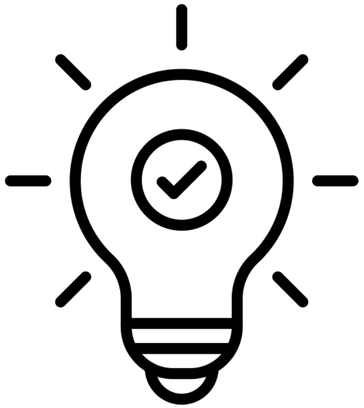
Принцип 2 — Исчерпывающее тестирование недостижимо



Грамотное управление ожиданиями

Кейс:
Владелец продуктовой компании регулярно отчитывал команду тестирования за “пропущенные” баги на продакшн, в духе: “как вы это умудрились пропустить?”

- **Вариации проблематики:**
  Любая риторика, направленна на то, что команда тестирования должна находить все ошибки.
- Отсутсвие четко сформированных критериев оценки результата тестирования (критерии оценки ожидаемого уровня качества).



Решение:
В какой-то момент я решил что настало время познакомить его с элементарными основами Тестирования.
Знакомство с двумя первыми принципами и беглая переоценка ресурсов сразу помогло владельцу компании прийти к мысли, что мы не могли и близко достигнуть состояния “проверить все”.

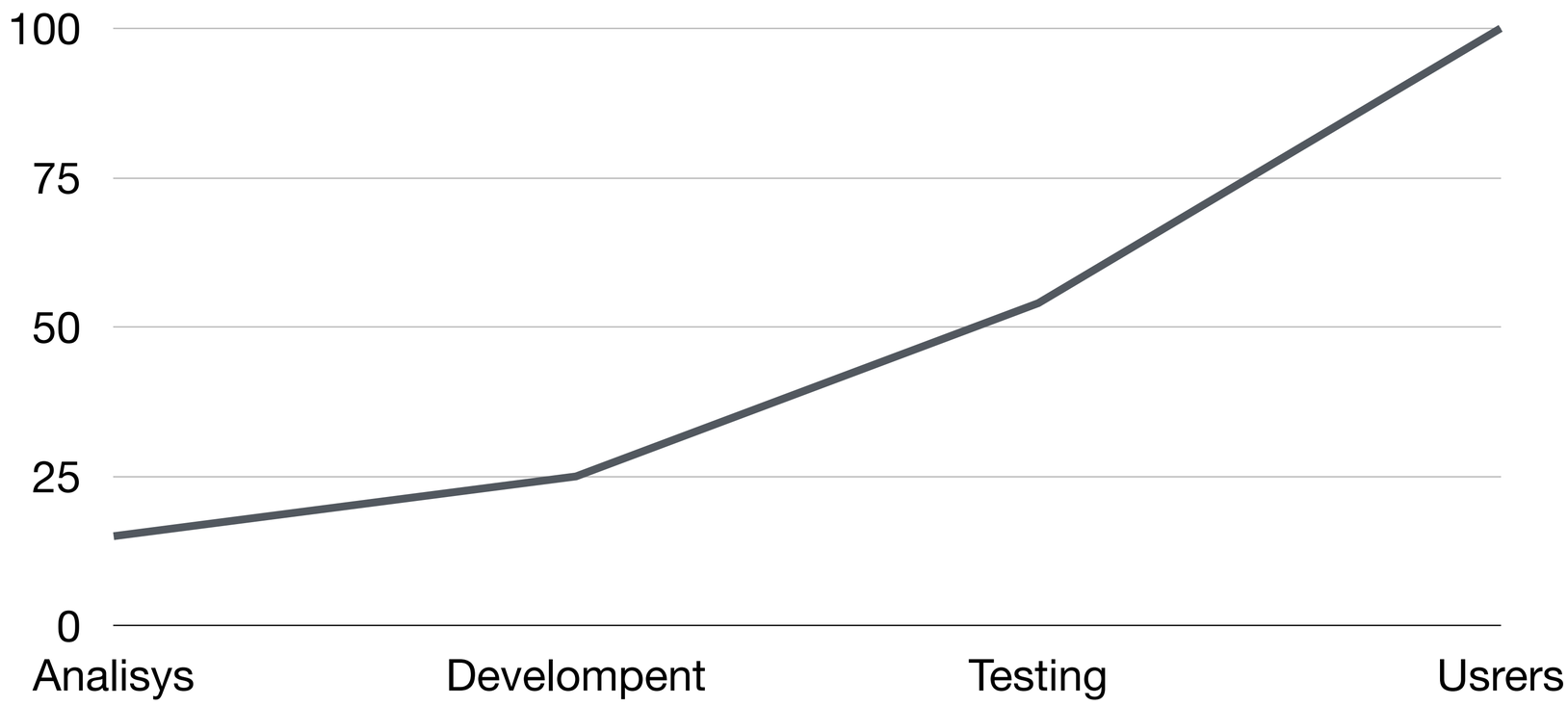
Кейс :
Старт-ап интегрировался с новой платежной системой, не хватало рук и интеграцию проверял один человек. По итогу, часть комбинаций вводных данных не была проверена и это вызвало недовольство проектного менеджера.

Решение:
Мы засекли сколько времени уходит на проверку одного сценария. Затем, оценили полный список из сценариев и посчитали чисто математически сколько времени на это должно уйти. Против математики ему возразить было нечего — сроки были огромные.

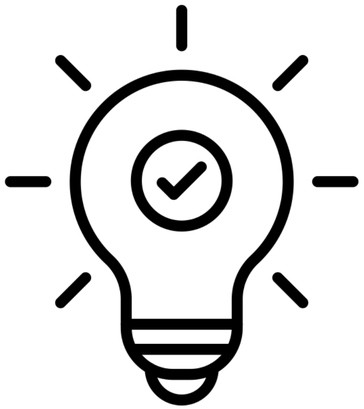
Совет:
Управление ожиданиями - очень важный элемент работы с другими участниками проекта.
Важно напоминать, что проверить все невозможно. В большинстве случаев - лучше имплементировать превентивные меры и добиться соответствия стандартам готовности функционала (definition of ready), чем думать что все ошибки будут найдены на стадии тестирования.

Принцип 3 — Раннее тестирование

<u>Основная польза принципа:</u> огромная экономия времени и средств



Кейс:
Тестировщики участвовали в процессе разработки только на этапе приемочного тестирования и жаловались на низкое качество продукта и непонятно сформулированные требования, которые не всегда правильно трактовались разработчиками.



Решение:
Было принято решение, что все User story будут проходить обязательное утверждение от QA. Было реализовано через статус в Jira, который могли редактировать только QA.

Как результат: очень выросло качество выполненной работы и значительно сократилось время, потраченное на переделку User story (в 4 раза меньше времени на доработку после первого тестирования).

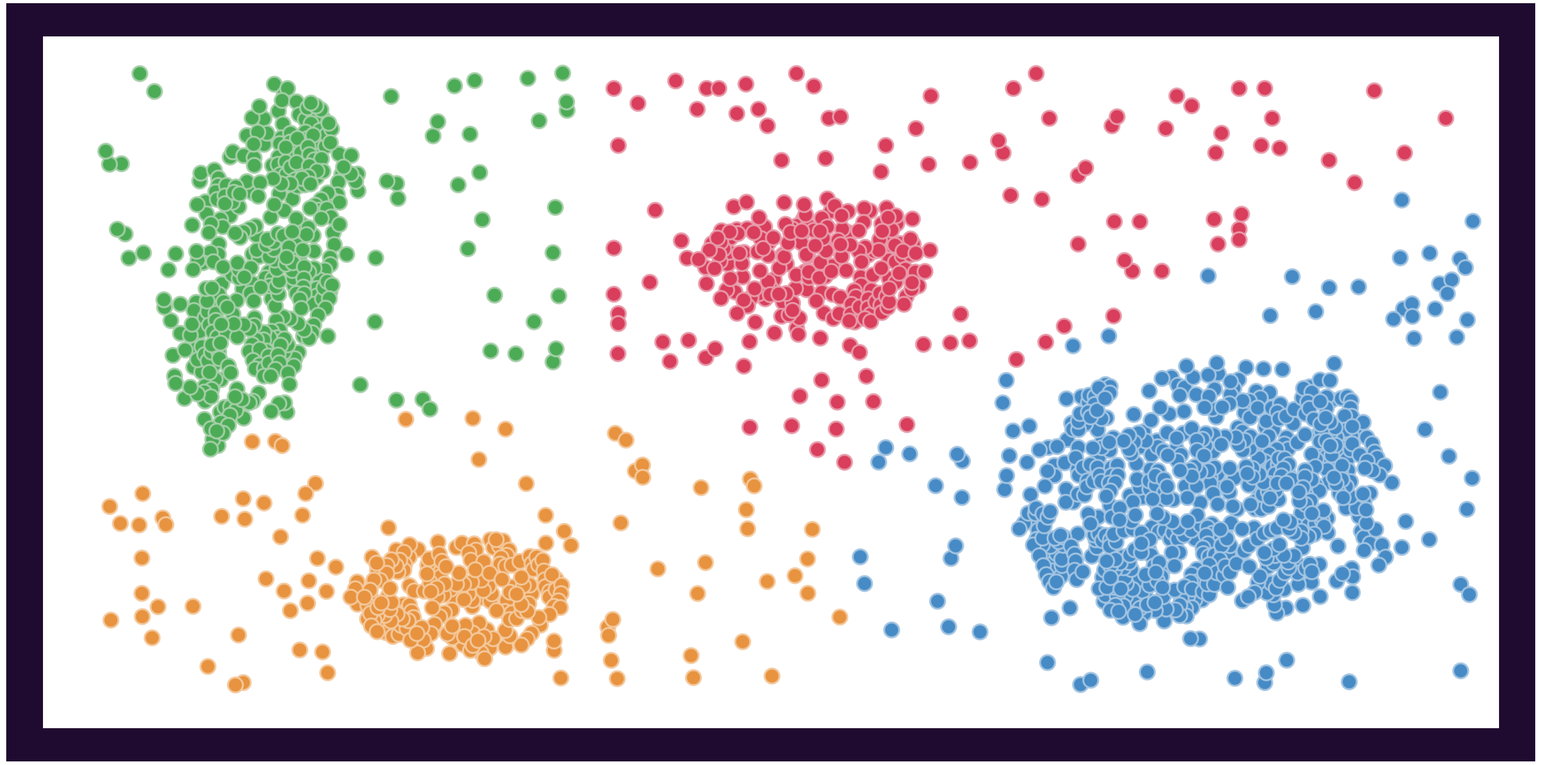
Кейс:
В одной компании, ввиду особенности продукта, релизы могли происходить только раз месяц, а UAT был организован уже на Production. Как результат: очень часто заказчик был недоволен реализацией и требовал небольших “доделок” в следущей поставке.

Решение:
Было организовано ранее “приемочное тестирование”, в формате Demo еще до стадии передачи на полноценное функциональное тестирование — таким образом была полностью исключена трата времени на проверку “не совсем того что надо”.

Совет:
Этот принцип самый мощный для бизнеса из всех семи — он демонстрирует огромную экономию на всех типах и этапах разработки ПО. Сам процесс раннего вовлечения многогранен и должен применяться как в общих, так и в частых случаях (на всех уровнях и типах тестирования).

Принцип 4 — Скопление дефектов (кластеризация)

Основная польза принципа: приоритизация тестирования, планирование затрат, эффективность.



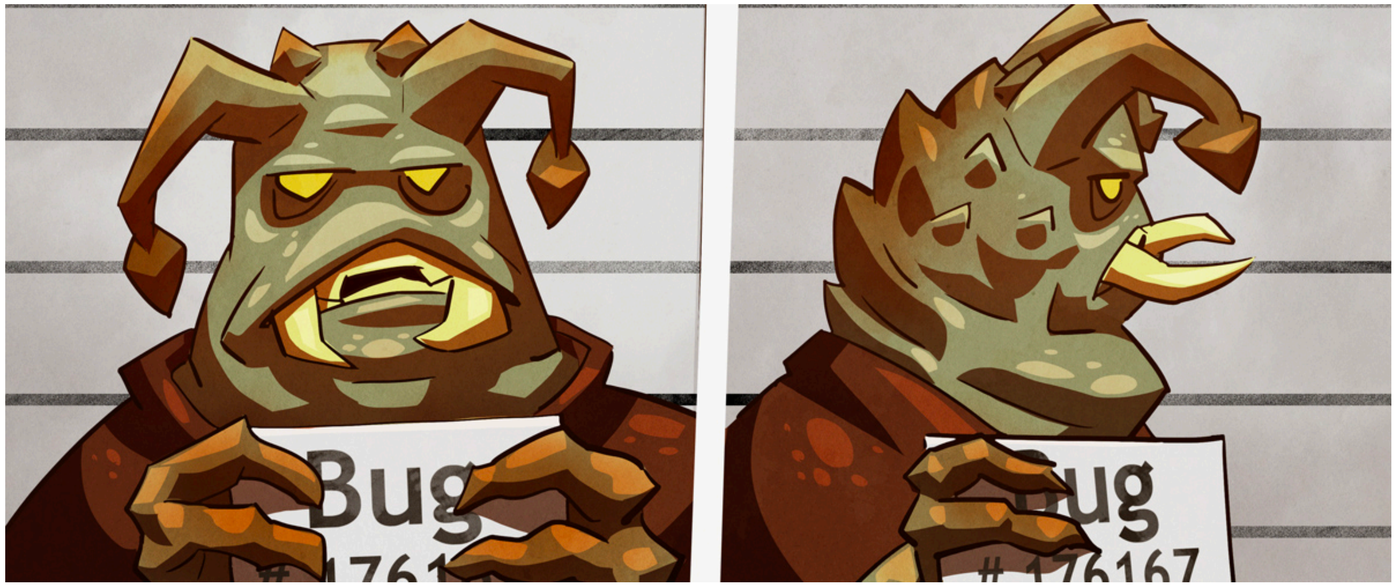
Кейс:
В медицинском приложении один из модулей системы (UI интерфейс), написанный на скорую руку, время от времени выдавал какие-то ошибки, делающие работу в нем невозможной.

Решение:
После подсчета с дев-лицом сколько времени уходит на то, чтобы фиксить и тестировать плохой модуль, было принято решение полностью его переписать и, таким образом, устранить кластеризацию дефектов.

Совет:
Проводите RCA для мест сильного скопления дефектов — иногда это может помочь спасти часы или дни работы. Ошибки также могут быть вызваны испорченными данными от прошлых багов или просто нестабильностью системы — важно исключать такое по максимуму и не тратить силы на проверку этого.

Принцип 5 — Парадокс пестицида

<u>Основная польза принципа:</u> динамичный тест дизайн обнаруживает намного больше ошибок и обеспечивает большую уверенность в качестве продукта



Кейс:
Тестировали приложение на скандинавских рынок. В рамках регрессии постоянно проходили регистрацию нового пользователя, для чего всегда использовался один и тот же тест. Спустя какое-то время уже реальные пользователи заметили что, невозможно зарегистрировать пользователя с умляутами (ä, ö и ü).

Решение:
Как оказалось, большинство тестов выполнялись с использованием очень стандартного “Test” в тестовых полях. Было сформировано и прикреплено к каждому кейсу несколько вариантов данных, которые стоило использовать в том или ином кейсе при прохождении регрессии так, чтобы покрытие данными задевало все 100% классов эквивалентности.

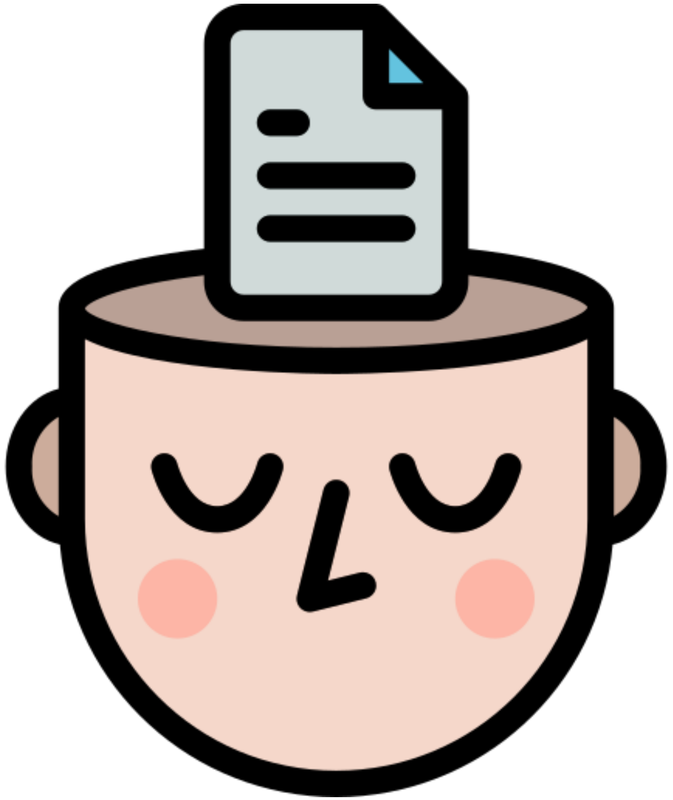
Кейс:
Автоматизированное тестирование. В E2E сценарии был шаг регистрации компании и заполнения формы с различными данными. Форма заполнялась сверху вниз. В какой-то момент один из пользователей пожаловался, что если заполнить сначала дату регистрации фирмы, то другие поля становятся недоступны (был баг вызванный javascript для дата-спикера)

Решение:
Был написан “райндомайзер” не только для этого блока, но и для всех подобных, который каждый раз менял последовательность шагов авто-тест кейсов, где они не должны были быть последовательными.

Совет:
Вообще любое изменение и расширение покрытия при помощи использования новых входящих данных это отличная идея, даже если сам тест кейс остался низменным. К тому же, не стоит забывать что Исследовательское тестирование как раз и было придумано для борьбы с этим парадоксом и стоит всегда отводить немного времени команды на него.

Принцип 6 — Тестирование зависит от контекста

Основная польза принципа:
контекстуальное тестирование помогает подобрать под проект оптимальные техники и подходы, чтобы максимально эффективно расходовать ресурс и получить наилучшие возможные результаты.



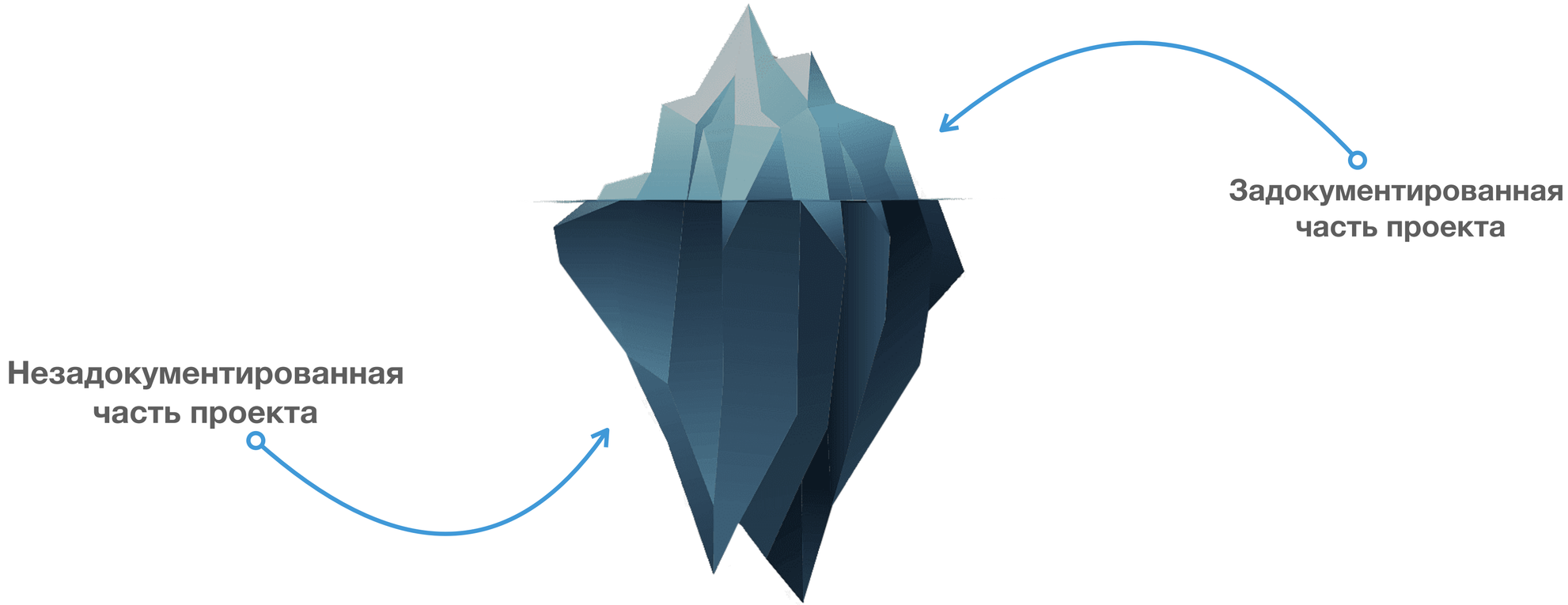
Кейс:
История произошла в компании где я работал, в “соседней” команде. Был нанят опытный QA lead для того чтобы наладить процессы тестирования на одном из проектов компании. Он с завидным энтузиазмом начал менять все процессы на то, как было в прошлой компании. По итогу процесс тестирования и коммуникаций был полностью развален. QA lead уволился не проработав и 4х месяцев.

Решение:
Каждый проект, каждая команда и каждый человек — уникальные переменные требующие отдельного анализа и стратегии взаимодействия.

Совет:
Ни одни курсы, ни одна книга или конференция не даст вам 100% рецепта успеха в вашем проекте и компании. Нужно научиться отделять хорошие идеи и грамотно их адаптировать - это единственный путь применять чужие знания.

Принцип 7 — Заблуждение об отсутствии ошибок

<u>Основная ценность принципа:</u> помогает создать действительно полезный для пользователей продукт, а следовательно успешный с точки зрения бизнеса



Кейс:
В компании, которую я консультировал, тест кейсы по которым надо проверять продукт создавал сам владелец продукта. Сам он был хорошим математиком, но увы, плохим тестировщиком. Как следствие: после выпуска нескольких версий, пользователи жаловались на наличие ошибок и неудобный интерфейс (UX).

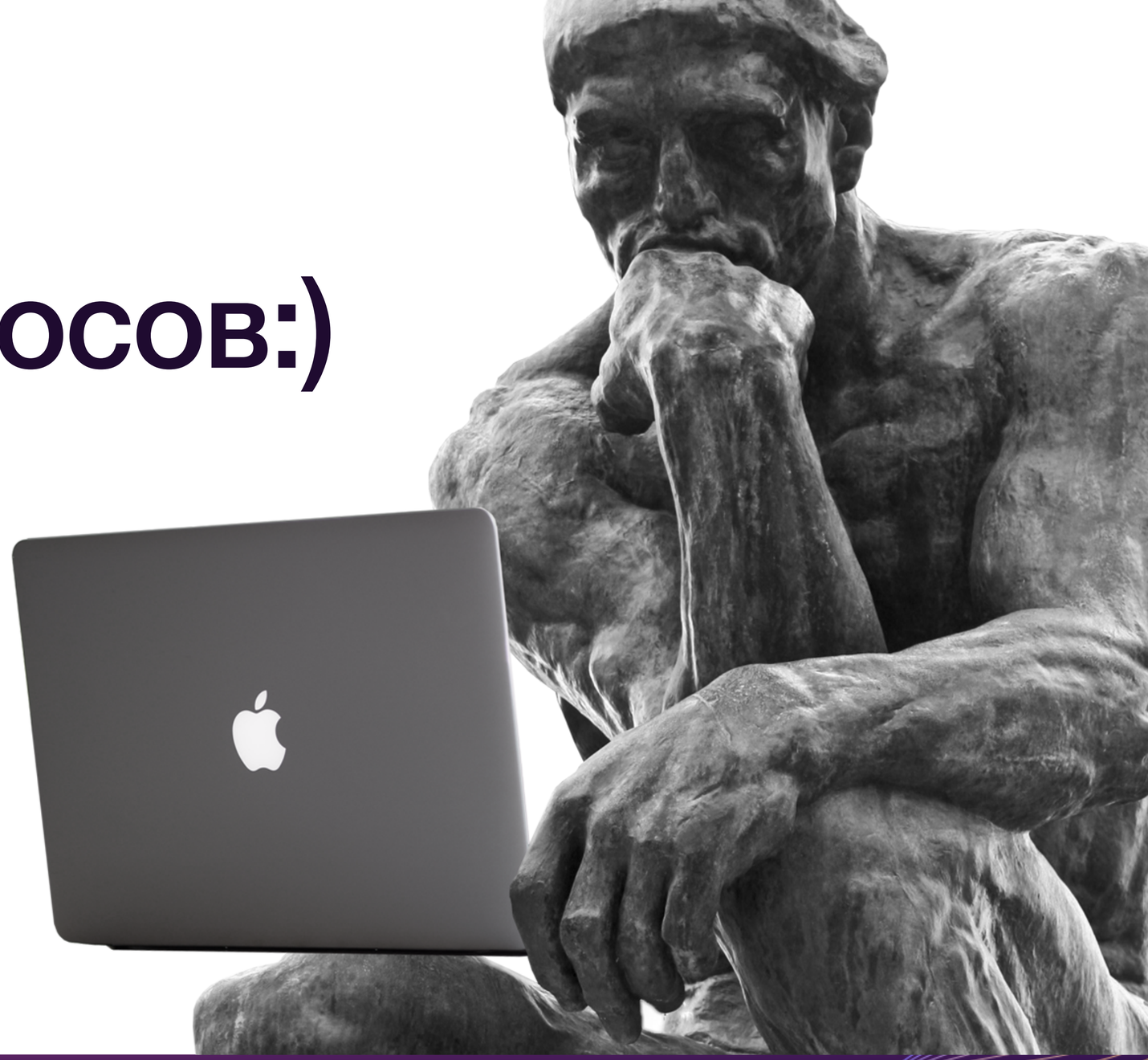
Решение:
Было принято решение в два раза уменьшить кейсы, генерируемые самим владельцем (оставить только критические), а остальные вверить самой команде тестирования. В тоже время команде тестирования был дан совет подойти к приложению как пользователи и найти почему им неудобно пользоваться.

Совет:
Тестирование - супер комплексный процесс и важно не забывать, что по итогу любым продуктом пользуются люди и именно их удобство во многом определит успех программного обеспечения. Тут пригодятся любые техники взаимодействия с пользователями.

Заключение

- Простой способ обьяснить сложное
- Управление ожиданиями от тестирования
- Грамотное планирование процессов
- Оценка трудозатрат их контроль
- Нашел проблему — посвяти своего менеджера!

Время вопросов:)



Спасибо за Внимание!