Модуль 1

Введение в дисциплину. Требования

Agenda

- Правила поведения (урок, Slack) и знакомство
- Понятие тестирования и его принципы
- Разграничение понятий QA, QC
- Анализ требований: понятие, уровни и типы. Свойства качественных требований
- Домашнее задание

Понятие тестирования

Тестирование ПО (простым языком) - это процесс проверки программного обеспечения (продукта), который подтверждает, что система работает в соответствии с требованиями (то есть делает то, для чего он предназначен).

Testing (by ISTQB) - the process consisting of all lifecycle activities, both static and dynamic, concerned with planning, preparation and evaluation of a component or system and related work products to determine that they satisfy specified requirements, to demonstrate that they are fit for purpose and to detect defects.

Правильно протестированный программный продукт обеспечивает надежность, безопасность и высокую производительность, что в дальнейшем приводит к экономии времени, денег и удовлетворенности клиентов.

Seven Testing Principles (ISTQB):

1. Testing shows the presence of defects, not their absence
2. Exhaustive testing is impossible
3. Early testing saves time and money
4. Defects cluster together
5. Beware of the pesticide paradox
6. Testing is context dependent
7. Absence-of-errors is a fallacy


testing
QA vs QC
WORK

QA - QC- testing

Термины "обеспечение качества" (QA) и "контроль качества" (QC) и тестирование часто используются как взаимозаменяемые. Хотя все они тесно связаны, являются частью управления качеством и иногда их трудно различить, они далеко не одно и то же. QA фокусируется на предотвращении дефектов, в то время как QC фокусируется на выявлении дефектов.

QA - это процесс, который гарантирует, что процедуры и стандарты подходят для проекта и правильно выполняются. Работа команды QA заключается в улучшении процессов разработки и тестирования, чтобы дефекты не возникали в процессе разработки продукта. Создаются и разрабатываются артефакты для онбординга новых сотрудников.

QC - это процесс, который проверяет, что проект следует установленным стандартам, процессам и процедурам (анализ результатов тестирования). Работа команды QC заключается в выявлении дефектов после разработки продукта, но до его выпуска. Цель QC - выявить (и исправить) дефекты в готовом продукте.

Тестирование - проверка создаваемого программного продукта на соответствие требованиям к этому продукту. Основная задача тестирования – выявить и зафиксировать дефекты.


QUALITY ASSURANCE
Define and improve the quality related processes and procedures to ensure quality
QUALITY CONTROL
Evaluate the qualty, estimate if it meets customer's expectations
TESTING
Find defects in the product

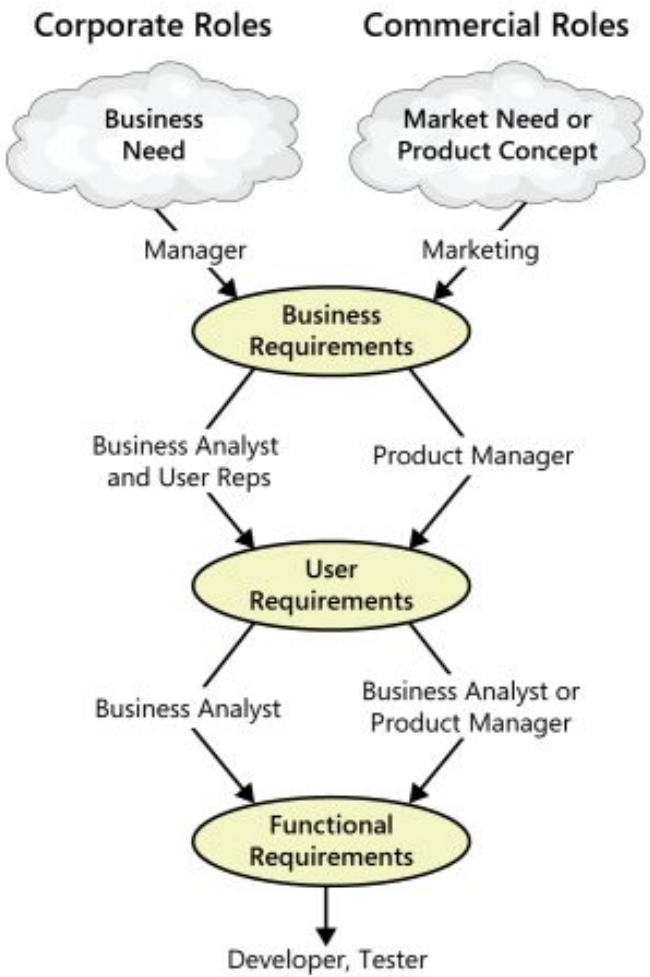
Анализ требований

Действия на этапе анализа требований:

- Анализ спецификаций системных требований с точки зрения тестирования
- Подготовка RTM, которая представляет собой матрицу прослеживаемости требований (traceability matrix)
- Определение методов тестирования и типов тестирования
- Приоритизация функциональности, требующей фокусировки тестирования
- Анализ возможности автоматизации
- Определение деталей среды тестирования, в которой будет проводиться фактическое тестирование.

Требования - это описание того, какие функции и с соблюдением каких условий должен выполнять программный продукт, чтобы пользователь мог решить свою проблему либо достичь цель.

Требования делятся на три категории: бизнес, пользовательские (User) и программные. Далее рассмотрим подробнее каждый уровень


Corporate Roles
Commercial Roles
Business Need
Market Need or Product Concept
Manager
Marketing
Business Requirements
Business Analyst and User Reps
Product Manager
User Requirements
Business Analyst
Business Analyst or Product Manager
Functional Requirements
Developer, Tester

Бизнес-требования. К ним относятся заявления высокого уровня о целях, задачах и потребностях. Бизнес-требования не включают никаких подробностей или конкретных функций. Они просто формулируют проблему и бизнес-цель, которую необходимо достичь, например:

- увеличение доходов/производительности/охвата клиентов,
- снижение затрат/ошибок,
- улучшение обслуживания клиентов и т. д.

Требования пользователя. Пользовательские требования отражают конкретные потребности или ожидания клиентов программного обеспечения. Они представляют собой расширенные функциональные возможности пользователя или сложные проблемы пользовательского интерфейса.

Требования к программному обеспечению. Спецификация, в которой указаны конкретные функции, нефункциональные требования и необходимые варианты использования программного обеспечения. По сути, SRS (software requirements specification) подробно описывает, что будет делать программное обеспечение.

Требования к программному обеспечению обычно делятся на:

- функциональные требования (FR)
- нефункциональные требования (NFR)

Функциональные требования.

Функциональные требования - свойства или функции продукта, которые разработчики должны реализовать, чтобы пользователи могли выполнять свои задачи. Поэтому важно сделать их понятными как для команды разработчиков, так и для заинтересованных сторон. Как правило, функциональные требования описывают поведение системы в определенных условиях.

Функциональное требование может выражать отношения "если/то", как в приведенном ниже примере:

“Если пользователь нажимает кнопку “Добавить товар в корзину”, то выбранный товар отображается в корзине”.

Функциональные требования могут детализировать конкретные типы вводимых данных, такие как имена, адреса, размеры и расстояния.

Функциональные требования относительно легко и просто тестировать, поскольку они определяют, как ведет себя система. Тест не проходит, если система не функционирует так, как ожидалось.

Нефункциональные требования.

Нефункциональные требования относятся к удобству использования программного обеспечения. Нефункциональные требования к программному обеспечению определяют, как система должна работать или выполнять свои функции. Система может соответствовать функциональным требованиям и не соответствовать нефункциональным требованиям.

NFRs определяют характеристики программного обеспечения и ожидаемый пользовательский опыт. Они охватывают, например:

- производительность
- удобство использования
- масштабируемость
- безопасность
- портативность

Пример нефункционального требования, связанного с производительностью:

“Страницы этого веб-сайта должны загружаться в течение 0,5 секунды”.

Виды требований:

*1) Спецификация требований к программному обеспечению.*Как функциональные, так и нефункциональные требования могут быть формализованы в документе спецификации требований к программному обеспечению (SRS). SRS содержит описания функций и возможностей, которые должен предоставлять продукт. Документ также определяет ограничения и допущения.

SRS должна включать следующие разделы:

- Цель. Определения, обзор системы и предыстория.
- Общее описание. Предположения, ограничения, бизнес-правила и видение продукта.
- Особые требования. Системные атрибуты, функциональные требования и требования к базе данных.

2) Сценарии использования (Use case). Варианты использования описывают взаимодействие между системой и внешними пользователями, которое приводит к достижению определенных целей.

Каждый вариант использования включает в себя три основных элемента:

Действующие лица. Это внешние/внутренние пользователи, другая система, взаимодействующие с системой.

Система. Непосредственно продукт, сервис, программное обеспечение.

Цели/Сценарии. Взаимодействия между действующими лицами и системой (основные, альтернативные)

https://www.usability.gov/how-to-and-tools/methods/use-cases.html

3) Пользовательские истории (User story).

Пользовательская история — это задокументированное описание функции программного обеспечения, рассматриваемое с точки зрения конечного пользователя. Пользовательская история описывает, что именно пользователь хочет, чтобы система делала. В Agile-проектах пользовательские истории организованы в бэклог, который представляет собой упорядоченный список функций продукта.

Типичная история пользователя записывается так:

As a <type of user>, I want <some goal> so that <some reason>.

As a small business owner, I want to create an invoice so that I can bill a customer.

Пользовательские истории должны сопровождаться критериями приемки (acceptance criteria). Это условия, которым должен удовлетворять продукт, чтобы быть принятым пользователем, заинтересованными сторонами или владельцем продукта. Каждая пользовательская история должна иметь хотя бы один такой критерий. Эффективные критерии приемки должны быть проверяемыми, краткими и полностью понятными всем членам команды и заинтересованным сторонам.

4) Прототипы. Прототип программного обеспечения — это общий термин для различных форм результатов ранней стадии, созданных для демонстрации того, как должны быть реализованы требования. Прототипы помогают преодолеть пробелы в видении и позволяют заинтересованным сторонам и командам прояснить сложные области разрабатываемых продуктов. Традиционно прототипы показывают, как будет работать решение, и дают примеры того, как пользователи будут взаимодействовать с ним для выполнения своих задач.

5) Модели и схемы. Графическое представление будущего программного продукта

Свойства хороших требований: в идеальном мире, каждое требование должно соответствовать следующим свойствам

Завершенное (complete). В документе должны быть точно прописаны все требования (подробно описаны бизнес-цели и преимущества, описаны все функции, ожидаемые от системы).

Атомарное (atomic). Требование является атомарным, если его нельзя разбить на отдельные требования без потери завершенности (описывается только одна ситуация). Сравним:

“The system shall interpret the keystroke combination Ctrl+S as File Save”.

“The system must permit search by order number, invoice number, and/or customer purchase

order number”.

Последовательное (consistency). Документы с требованиями к ПО часто бывают длинными и разделены на несколько частей, каждая из которых имеет свои специфические требования. Последовательные требования не имеют конфликтов, таких как различия во терминологии. Требование должно быть указано только один раз.

Понятное и недвусмысленное (unambiguous, clear). Требования к ПО не должны быть двусмысленными (подвержены множеству интерпретаций). Требования к ПО должны обеспечивать максимальную ясность, написаны простым языком, без терминов и жаргона, характерных для предметной области.

“The system shall provide automated information collection of license key data for a mass release from the Delivery/Fulfillment Team”.

“The Text Editor ideally shall be able to parse ASCII documents that define federal laws”.

Выполнимое и проверяемое (feasible and verifiable). Цель документа с требованиями — предоставить дорожную карту для реализации. Например, такое требование, как «должен начаться быстро», не поддается измерению.

Прослеживаемое (traceable). Трудно понять, когда разработчики фактически закончили программный проект. В идеале должна быть прямая связь между документами требований и готовым кодом. Код, присутствующий без соответствующего требования, может быть излишним или даже вредоносным.

Актуальное и корректное (up-to-date, correct).

Домашнее задание

Some examples of requirements. How do you think these requirements are good or bad and why?

1. I want to download the list of all users in CSV or Excel.
2. I want to build a support system with live chat, contact form, help and case management
3. I want to store user's Facebook ID in the database
4. There has to be a way to authenticate
5. The system must have good usability
6. Response time should be less than 5 seconds
7. The system shall work just like the previous one, but on a new platform
8. The system has to be bug-free
9. Easy to use

The End for today