

Введение в анализ требований

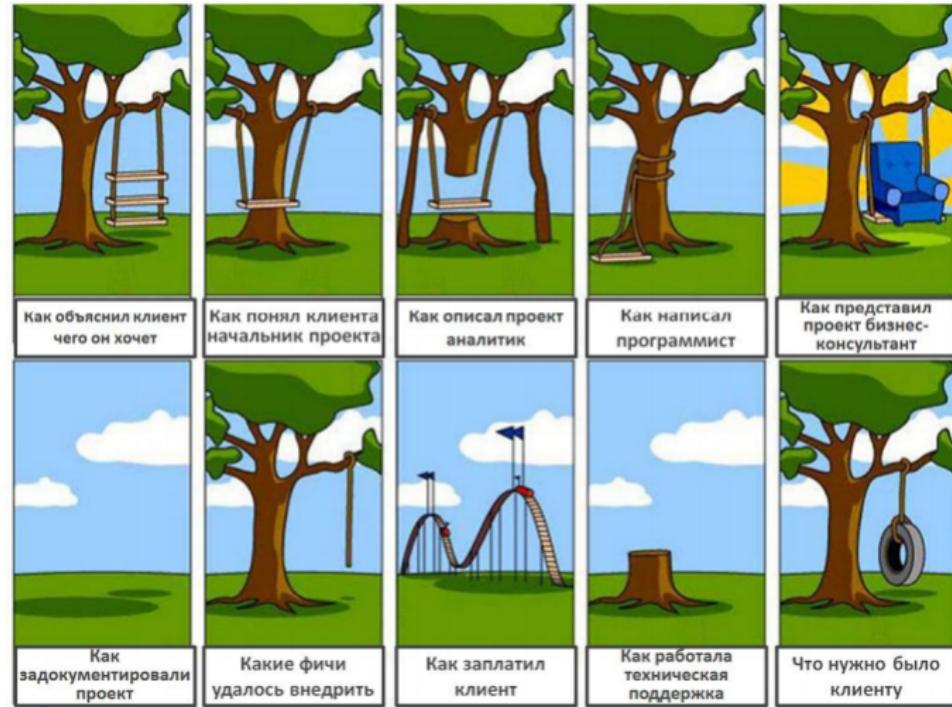
Наумов Д.А., доц. каф. КТ

Основы программной инженерии, 2019

Содержание лекции

- 1 Понятие анализа требований
- 2 Структура анализа требований
- 3 Виды требований
- 4 Визуальное моделирование и его средства
- 5 Понятие и применение UML
- 6 Диаграмма вариантов использования
- 7 Виды требований
- 8 Методы выявления требований
- 9 Инструменты выявления требований
- 10 Критерии формулировки требований

Понятие анализа требований



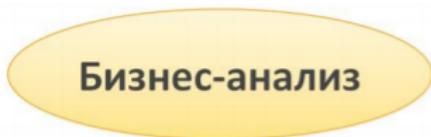
Понятие анализа требований

Требования к программному обеспечению

совокупность утверждений относительно атрибутов, свойств или качеств программной системы, подлежащей реализации.

Создаются в процессе разработки требований к программному обеспечению, в результате анализа требований.

Понятие анализа требований



выявление требований
(requirement elicitation)

анализ требований
(requirement analysis)

моделирование бизнес
процессов
(business process modeling)

Анализ требований — это процесс сбора требований к ПО, их систематизации, документирования, анализа, выявления противоречий, неполноты, разрешения конфликтов в процессе разработки программного обеспечения.



Структура анализа требований

Сбор требований

общение с клиентами и пользователями, чтобы определить, каковы их требования



Анализ требований

определение, являются ли собранные требования неясными, неполными, неоднозначными, или противоречащими, и затем решение этих проблем



Документирование требований

Требования могут быть задокументированы в различных формах, таких как простое описание, сценарии использования, пользовательские истории, или спецификации процессов



Виды требований по уровням

Бизнес-требования

определяют назначение ПО, описываются в документе о видении и границах проекта.

Пользовательские требования

определяют набор пользовательских задач, которые должна решать программа, а также способы (сценарии) их решения в системе.

Функциональные требования

охватывают предполагаемое поведение системы, определяя действия, которые система способна выполнять.

Виды требований по уровням

Пользовательские требования

могут выражаться в виде фраз утверждений, в виде способов применения (use case), пользовательских историй (user story), сценариев взаимодействия (scenario).

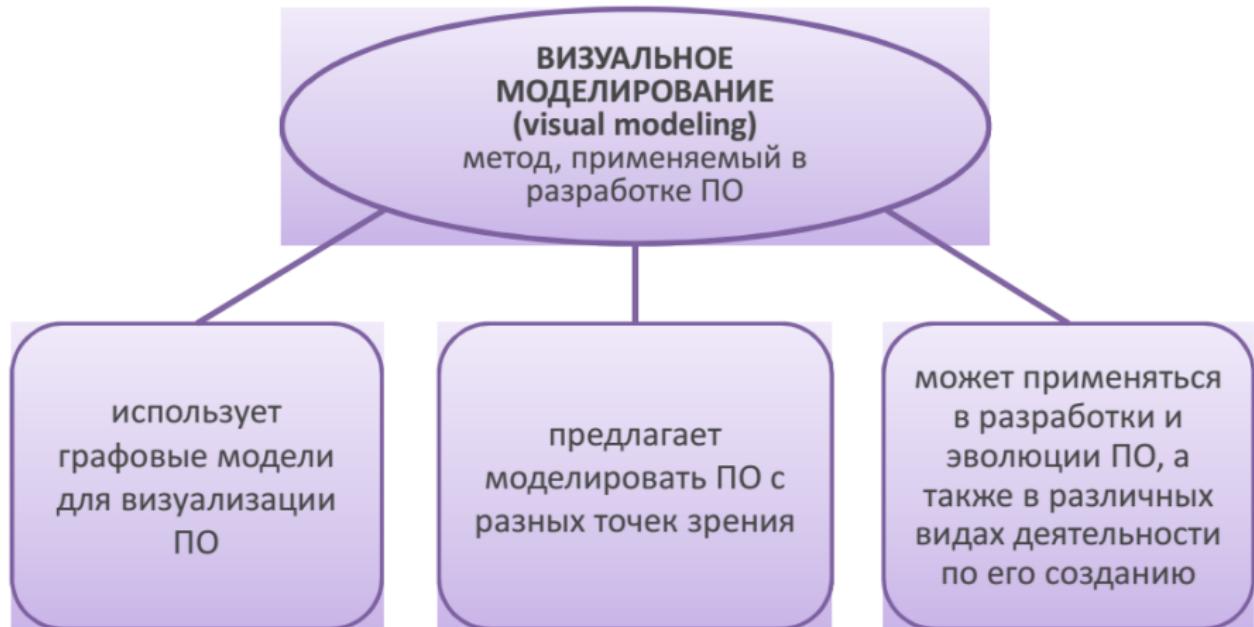
Функциональные требования

описывается в системной спецификации (system requirement specification, SRS).

Пример диаграммы Use Case



Визуальное моделирование и его средства



Язык UML

Унифицированный язык моделирования (UML)

язык графического описания программных сущностей в виде диаграмм.

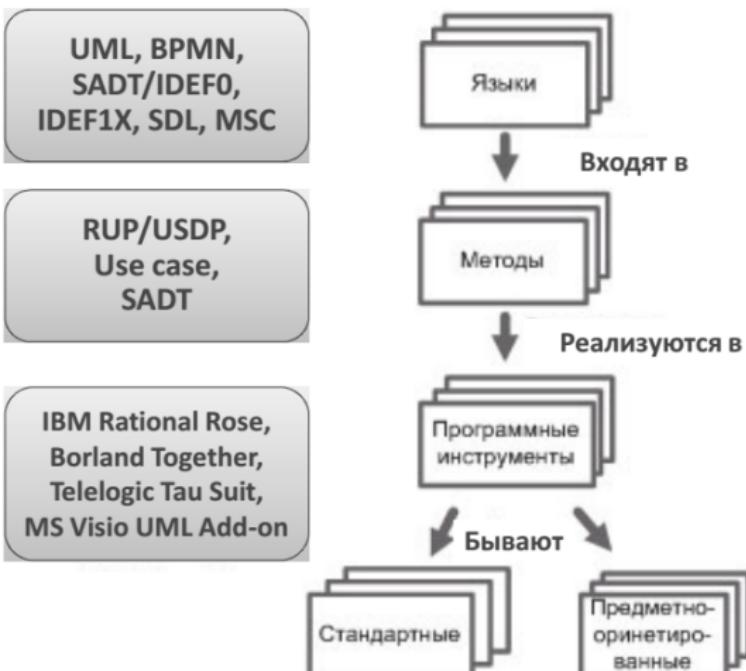
- диаграммы предметной области;
- диаграммы предполагаемого дизайна;
- диаграммы завершенной реализации;

Три уровня:

- концептуальный уровень;
- уровень спецификации;
- уровень реализации.

Визуальное моделирование и его средства

Визуальное моделирование применяется на практике с помощью методов, языков и соответствующих программных инструментов.



Язык UML

Три вида диаграмм UML:

- статические - описывают неизменную логическую структуру программы, а именно элементы - классы, объекты, структуры данных - и отношения между ними;
- динамические - показано, как программные сущности изменяются во время выполнения: поток выполнения или изменение состояния сущностей;
- физические - изображается неизменная физическая структура системы: исходные файлы, библиотеки, двоичные файлы, файлы данных и прочее, а также связи между ними.

Понятие и применение UML

UML (Unified Modeling Language) — унифицированный язык моделирования для описания, визуализации и документирования объектно-ориентированных систем в процессе их анализа и проектирования



Назначение UML

- Предоставить разработчикам **легко воспринимаемый и выразительный язык визуального моделирования**, специально предназначенный для **разработки и документирования моделей сложных систем различного целевого назначения**.
- Снабдить исходные понятия языка UML **возможностью расширения и специализации для более точного представления моделей систем в конкретной предметной области**.
- Графическое представление моделей в нотации UML **не должно зависеть от конкретных языков программирования и инструментальных средств проектирования**.
- **Облегчение контроля хода разработки** благодаря повышению уровня абстракции.
- **Облегчение взаимодействия** между разработчиками ПО и представителями других специальностей.

Виды диаграмм UML



Диаграмма вариантов использования

Диаграмма вариантов использования - диаграмма, на которой изображаются варианты использования проектируемой системы, заключенные в границу системы, и внешние актеры, а также определенные отношения между актерами и вариантами использования

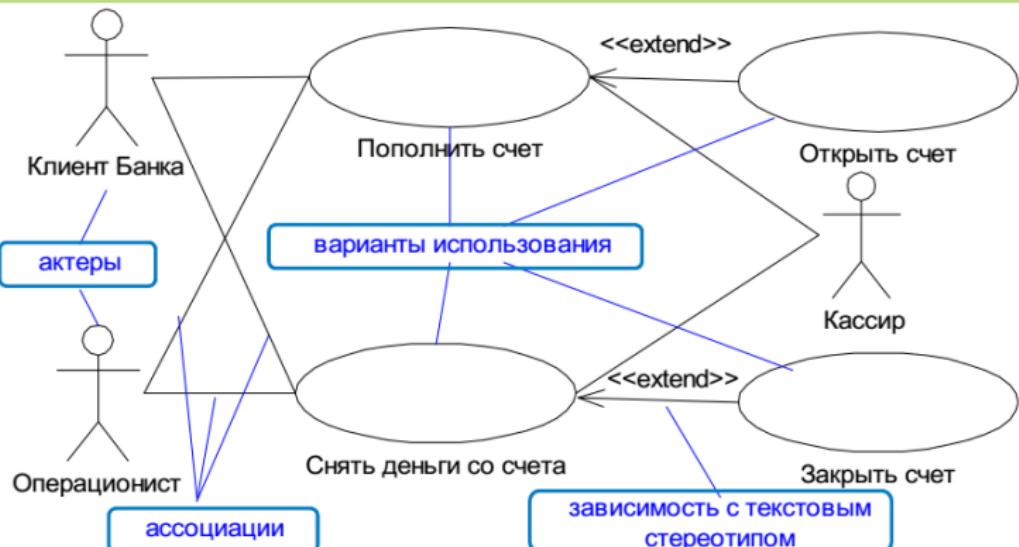


Диаграмма вариантов использования

Назначение диаграммы
вариантов использования

Определить общие границы
функциональности
проектируемой системы в
контексте моделируемой
предметной области

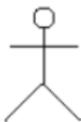
Специфицировать требования к
функциональному поведению
проектируемой системы в
форме вариантов
использования

Разработать исходную
концептуальную модель
системы для ее последующей
детализации в форме
логических и физических
моделей.

Подготовить исходную
документацию для
взаимодействия разработчиков
системы с ее заказчиками и
пользователями

Диаграмма вариантов использования

Основные обозначения на диаграмме вариантов использования



Актер (actor)



Вариант
использования
(use case)



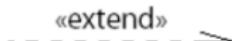
Граница
системы
(system boundry)



ассоциация



обобщение



расширение



включение

Диаграмма вариантов использования

Вариант использования (Use Case)

– представляет собой общую спецификацию совокупности выполняемых системой действий с целью предоставления некоторого наблюдаемого результата, который имеет значение для одного или нескольких актеров

Отвечает на вопрос «Что должна выполнять система?», не отвечая на вопрос «Как она должна выполнять это?»

Имена – отглагольное существительное или глагол в неопределенной форме

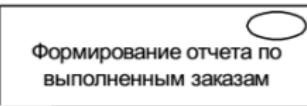
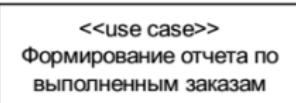
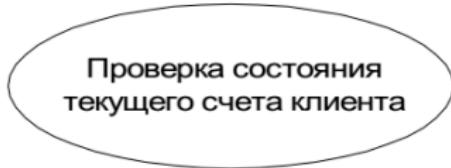
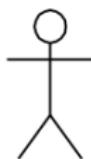


Диаграмма вариантов использования

Актер (actor)

- любая внешняя по отношению к проектируемой системе сущность, которая взаимодействует с системой и использует ее функциональные возможности для достижения определенных целей или решения частных задач

Примеры актеров: кассир, клиент банка, банковский служащий, президент, продавец магазина, менеджер отдела продаж, пассажир авиарейса, водитель автомобиля, администратор гостиницы, сотовый телефон



Клиент банка

<<actor>>
Посетитель
Интернет-магазина



Удаленный
пользователь

Диаграмма вариантов использования

Отношения на диаграмме вариантов использования

<i>Relationship</i>	<i>Function</i>	<i>Notation</i>
ассоциация	Служит только для обозначения взаимодействия актера с вариантом использования.	_____
включение	Специфицирует тот факт, что некоторый вариант использования содержит поведение, определенное в другом варианте использования	«include» - - - →
расширение	Определяет взаимосвязь одного варианта использования с другим, функциональность или поведение которого задействуется первым не всегда, а только при выполнении некоторых дополнительных условий	«extend» - - - →
обобщение	Предназначено для спецификации того факта, что один элемент модели является специальным или частным случаем другого элемента модели	→

Диаграмма вариантов использования

Достоинства модели вариантов использования :

- Определяет пользователей и границы системы;
- Определяет системный интерфейс;
- Удобна для общения пользователей с разработчиками;
- Используется для написания тестов;
- Является основой для написания пользовательской документации;
- Хорошо вписывается в любые методы проектирования (как объектно-ориентированные, так и структурные).

Виды требований по характеру

ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ

требования к поведению системы

Функциональная целесообразность (functional appropriateness) –

реализованные функции соответствуют требуемым

Функциональная полнота (functional completeness) –

полнота реализации требуемых функций

описание того,
что система
должна делать

Функциональная корректность (functional correctness) –

точность реализации требуемых функций

Способность к интеграции (interoperability) –

наличие специальных входов и выходов для интеграции с другим ПО

Безопасность (security) –

инкапсуляция состояний; все входы, выходы – ожидаемые

Виды требований по характеру

НЕФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ

требования к характеру поведения системы

Надёжность (reliability) –

устойчивость к отказам, восстанавливаемость

Удобство использования (usability) –

понятность, простота, эстетичность, user error protection

Эффективность (efficiency) –

эффективность с точки зрения ПО (performance)

эффективность с точки зрения бизнеса заказчика

Удобство сопровождения (maintainability) –

удобство обновления, модификации, тестирования

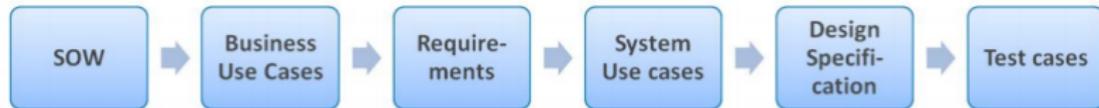
Портативность (portability) –

удобство установки, совместимость с другими ОС, кросбраузерность

описание того, как
система должна
делать то, что
определено в
функциональных
требованиях

Трассировка требований

Возможные уровни представлений требований по времени возникновения:



Возможные уровни представлений требований по зависимости (вложенности):



Трассировка требований

- SOW – Statement of Work, требования высокого уровня, наименование конкретные, без четкой детализации.
- Business Use Cases – варианты использования (Use Case) на уровне бизнеса
- Requirements – список требований, разделенные на функциональные, нефункциональные и т.п. Структурированный документ.
- System Use Cases – варианты использования (Use Case) на уровне ПО Design Specification – проектная спецификация, техзадание (для разработчиков).
- Test Cases – варианты использования готового ПО, соответствия реального поведения и желаемого. Наиболее формализованные требования.

Методы выявления требований

Традиционные методы

Интервью

Анкетирование

Наблюдение

Изучение документов и программных систем

Мозговые штурмы, семинары

Современные методы

Прототипирование

Совместная разработка приложений (JAD-метод)

Быстрая разработка приложений (RAD-метод)



Методы выявления требований

Традиционные методы

это простые и экономичные методы. Эффективность традиционных методов обратно пропорциональна риску проекта.

Современные методы

более глубокое проникновение в суть требований, но за счет большей цены и усилий.

Факторы, обуславливающие высокий риск проекта:

- неясные цели;
- недокументированные процедуры;
- нестабильные требования;
- слабое знание дела пользователями;
- неопытные разработчики;
- недостаточная приверженность пользователей разработке.

Интервью

Необходимо принять во внимание такие характеристики интервьюируемого как **предрасположенность, опыт и мастерство**, поскольку данные особенности могут повлиять на качество полученной во время интервью информации.

- *Подготовка* – планирование процесса опроса и выработка стратегии управления этим процессом. - выбор нужного собеседника; договоренность о встрече; формирование предварительной программы встречи; изучение сопутствующей информации; согласование плана опроса с группой проектирования.
- *Проведение опроса.*
- *Завершение.* Опрос нужно завершать, если: получен достаточно большой объем информации; поступает большой объем неподходящей информации; информация перестает усваиваться; эксперт начинает уставать; с экспертом возник конфликт.)



Анкетирование

Анкетирование проводится при условии готовности опрашиваемых к правдивым ответам.

Преимущество: наименее затратный способ извлечения информации.

Недостаток: наименее эффективный способ сбора данных

- *Многоальтернативные вопросы.* Предполагает множественные ответы на вопросы; может расширяться комментариями респондента в свободной форме.
- *Рейтинговые вопросы.* Предполагает использование лингвистических переменных: "абсолютно согласен" "согласен" "отношусь нейтрально" "не согласен" "абсолютно не согласен" "не знаю".
- *Вопросы с ранжированием.* Предусматривает ранжирование (упорядочивание) ответов путем присваивания им порядковых номеров, процентных значений и т.п.)

Наблюдение

Применяется для сбора сведений о параметрах, признаках и объектах в соответствующей предметной области.

Важные для изучения параметры, признаки и объекты точно оцениваются сотрудниками и регистрируются в карточках или в формуларах (например, по частоте, количеству, продолжительности, затратам).

- *Достоинство:* сбор информации, которую невозможно получить путем опроса или изучения документации.
- *Недостаток:* наблюдатель «вносит помехи» в результаты измерений

Изучение документов и программных систем

Используется при наличии:

- хорошо структурированной документации, описывающей устоявшиеся в организации бизнес-процессы;
- большого опыта разработки ИС в схожих предметных областях.
- *Достоинство:* предварительное формирование требований происходит в удобном для аналитика режиме.
- *Недостаток:* возможность пропуска важной информации, связанной с выполнением бизнес-процессов в реальной жизни и не вошедшей в документы.

Семинары

Групповое обсуждение проводится проектировщиками совместно с заказчиками, включая пользователей с целью обобщения и обсуждение важных для решения проблем вопросов.

- *Недостаток:* одна из наиболее затратных
- *Достиныства:* быстрота принятия решений, снижение количества ошибок, выработка нетривиальных идей.

Прототипирование

Прототипирование - это техника для построения быстрой и приблизительной версию желаемой системы или части этой системы.

Программный прототип – это «зеркало», в котором видно отражение того, как понял исполнитель требования заказчика.

- прототип демонстрирует возможности системы пользователям и дизайнерам.
- прототип представляет механизм связи, позволяющий рецензентам, понять взаимодействие внутри системы.
- В некоторых случаях, создание прототипов может создать впечатление, что разработчики зашли дальше в развитии проекта, чем есть на самом деле, что может предоставить пользователям нереалистичные ожидания окончания проекта

Прототипирование

Прототипирование является ключевым компонентом методологии **быстрой разработки приложений** (RAD – Rapid Application Development).

RAD базируется на следующих принципах:

- эволюционное прототипирование;
- использование CASE-средств, обладающих возможностями прямого и обратного проектирования и автоматической генерации кода;
- высококвалифицированные специалисты;
- совмещение живого общения с разработкой в режиме online;
- жесткие временные рамки.

Характеристики качественных требований

Корректность

Недвусмысленность

Полнота набора требований

Непротиворечивость набора
требований

Проверяемость
(тестопригодность)

Трассируемость

Понимаемость

Корректность

Насколько корректно наше требование? Действительно ли это то, что требуется от системы или кто-то допустил ошибку/опечатку в процессе написания требования?

Пример: после набора номера пользователь должен слышать короткие гудки (символизирующие о том, что идет звонок)

Описание: просто перепутали слово. Гудки должны быть длинными. Как тестировать на корректность и находить такие ошибки:

- нужны хорошие доменные знания в области
- нужно смотреть на трассировку вверх и вниз, возможно обнаружится нестыковка и будет зацепка;
- процесс ревью также может помочь (желательно, чтобы это ревью проводил; тест-аналитик или тот, кто тоже имеет отношение к написанию требований).

Недвусмысленность

Могут ли 2 различных человека понять требование по-разному?

Пример: Телефон должен работать в автономном режиме, когда он питается от батареек. В автономном режиме недоступны следующие функции: bla-bla-bla.

Описание: должны ли быть доступны функции «bla-bla-bla», если телефон с батарейками, но подключен к сети? Я могу подумать, что да, программист — что нет. Заказчик тоже может это интерпретировать как захочет

Как тестировать на корректность и находить такие ошибки:

- Проверять «ветвистость» требований
- Ревью, аналогично предыдущему пункту
- В идеале — стараться избегать ветвлений в требованиях. Если это невозможно — форматировать их в виде таблиц со всеми возможными вариантами.

Полнота набора требований

Насколько полным является набор требований? Если есть секция в SRS, определяющая функциональность модуля, то вся ли функциональность этого модуля покрыта требованиями? Нет ли дыр?

Пример: Есть секция требований, определяющая работу со спец-кнопками, и в этих требованиях упущена из виду одна из спец-кнопок. Для этой кнопки просто нет требований.

Как тестировать на корректность и находить такие ошибки:

- нужно смотреть на трассировку требований вверх и вниз (на бизнес-требования и на низкоуровневые требования — дизайн, макеты, детальное описание реализации).
- Если в этих требованиях есть то, что упущено в SRS — такая ошибка сразу обнаружится.
- Если этого и там нет — то, возможно, этой функциональности и не должно быть? Или должна быть, но она упущена из виду во всех документах

Непротиворечивость набора требований

Поиск требований, которые противоречат друг другу.

Пример:

Требование 1 (из раздела функциональности спец-кнопок): когда активен режим «Mute», телефон не должен издавать никаких звуков

Требование 245 (из раздела интерфейса): когда пользователь снимает трубку, телефон должен издавать тоновые гудки

Как тестировать на корректность и находить такие ошибки:

- обращать внимание на общие формулировки в требованиях;
- делить на категории (например, выделить все требования, регламентирующие звуковые сигналы) и ревьюовать их направленно на предмет противоречий;
- выделять все требования, трассирующиеся на одно верхнеуровневое требование и анализировать такие наборы.

Проверяемость (тестопригодность)

Один из основных и самых важных критериев (для QA инженеров). Возможно ли проверить это требование и убедиться, что оно выполняется?

Пример: в случае возникновения критической ошибки телефон должен перезагрузиться

Пример 2: информация на дисплее телефона должна отображаться в понятном пользователю виде

Как тестировать на корректность и находить такие ошибки:

- Во время анализа требований задавать вопрос: «Как я буду это проверять?». Если однозначного ответа нет — значит нужно более детально анализировать, и, возможно, вносить правки в требование (уточнения, ограничения)
- Во время анализа требований выявлять общие формулировки, требующие перебора неопределенного числа вариантов для проверки выполнения требования.

Трассируемость

Трассируемость — это связь с требованием выше и требованием ниже.

Пример 1. Бизнес-требование не имеет ни одного SRS требования. Соответственно, получается пробел в реализации (мы не сделаем того, что нужно бизнесу)

Пример 2. SRS требование описывает то, чего не было в бизнес-требованиях. Получается, мы делаем то, что не требовалось. Как тестировать на корректность и находить такие ошибки:

- если используется какая-то система менеджмента требований — то там, скорее всего, уже есть функциональность, позволяющая в автоматическом режиме следить за трассировкой
- если системы менеджмента требований нет, то остается "дедовский" способ — составлять матрицы трассируемости и отслеживать связи всех требований на всех уровнях.

Понимаемость

Могут ли все участники процесса понять, что требуется от системы по описанию требования? Часто бывают ситуации, когда требование может быть понятно разработчику, но не понятно тестировщику (или наоборот). При этом требование может вполне соответствовать остальным критериям.

Пример: передатчик телефона должен использовать амплитудно-фазовую модуляцию с несущими от 2 до 7,5 МГц с шагом 500 кГц

Как тестировать на корректность и находить такие ошибки:

- стараться представлять себя на месте заказчика/аналитика/простого пользователя и пытаться представить, будет ли понятно это требование;
- обращать внимание на двойственные термины (особенно, аббревиатуры), которые могут интерпретироваться по-разному различными людьми.

Пример выявления требований

Что хочет Заказчик?

- Продавать книги через интернет
- Более гибко управлять имеющимся ассортиментом
- Расширить свой бизнес

Что это всё значит для Исполнителя?

- ???
- ???
- ???

Пример выявления требований

Заказчик

Покупатель приходит в один из трёх моих магазинов. Он либо покупает нужную ему книгу либо не находит нужную на полках магазина и уходит. некоторые покупатели просят продавца заказать не найденную книгу, оставляют деньги в залог и ждут, когда продавец перезвонит и сообщит, что книга привезена – приходят и покупают её, доплатив нужную к задатку сумму. Бывают конечно случаи, когда книги возвращают на основании закона о потребительских правах, но это, как правило, редко происходит.

Что я еще могу сказать... для частых покупателей мы ввели систему скидок. Для этого мы изготовили пластиковые карты, которые выдаем тем покупателям, которые совершают покупку более, чем на 400 грн за один раз. Карточки эти накопительные: первоначально скидка равна 2%, но по мере того как покупатель докупает книги на сумму на 3000 грн скидка становится равной 5%, а в случае если покупатель купил книг суммарно на 10000, в том числе потратив 2000 за последние пол года то мы даем 10% скидку. Мы называем эти скидки Стандартная, Премиум и VIP.

Да, вот еще: иногда, во время того, как покупатель расплачивается, мы предлагаем ему оставить нам его адрес электронной почты, что бы мы могли присыпать ему информацию о новых книгах на интересующую его тему.

Это всё, что я могу вам рассказать.

Пример выявления требований

Составляем словарь предметной области (ядро словаря)

Владельцы глаголов	Глаголы	Цели действия глаголов
Покупатель	приходит	(в) магазин
	покупает	книгу
	находит	книгу
	уходит	(из) магазин
	просит	продавца
	оставляет	деньги
	ждёт	продавца
	возвращает	книгу
	оставляет	адрес
Продавец	заказать	книгу
	сообщит	что-то (клиенту)
Магазин	выдает	карту
	даёт	скидку
	присыпает	информацию

Пример выявления требований

••• Сформулированные бизнес-требования:

1. Покупатель должен иметь возможность выполнять следующие операции:
 - покупка книги;
 - просмотр книги;
 - заказ книги;
 - возврат книги.
2. Купленная книга должна быть однозначно идентифицируемой для осуществления действия возврат.
3. Должна предоставляться возможность документировать информацию о покупателях, их контактных данных и книжных предпочтениях.
4. Должна предоставляться возможность документировать информацию о совершенных покупателем в разные моменты времени покупках с указанием их стоимости.
5. Должна предоставляться возможность снижать стоимость покупки покупателю, с учетом истории его предыдущих покупок.

Пример выявления требований

•••• Что нужно учесть для полноты моделирования?

- Точки зрения на происходящее **всех** владельцев глаголов из словаря;
- Специфичность среды в которой будет функционировать конечный продукт (нормативные документы, законы и прочие правила, действующие в предметной области существующего бизнеса Заказчика и в области, в которой будет функционировать создаваемый продукт);
- Все детали в рассказе Заказчика;
- Тот факт, что Заказчик не обязан рассказывать правду – он всего лишь описывает ИСТИНУ со своей точки зрения;
- Тот факт, что Вы всё равно ошибаетесь в интерпретации услышанного от Заказчика;
- Тот факт, что если Вам абсолютно ясно, о чем говорит Заказчик, то это значит, что Вы нафантализировали свой смысл по его словам и надо подвергнуть сомнению свою уверенность в понимании и уточнить информацию. И еще раз уточнить.

Некоторые инструменты выявления требований

- **Метамодель НЛП** — модель языка, которая определяет те лингвистические паттерны, которые делают неясным смысл коммуникации, благодаря процессам искажения, упощения и обобщения.
- **Метамодель НЛП** определяет конкретные вопросы, имеющие целью прояснить и поставить под сомнение неточности языка, чтобы восстановить их связь с сенсорным опытом и конкретизировать.

ОСНОВНЫЕ ПАТТЕРНЫ МЕТАМОДЕЛИ НЛП

- Неконкретные имена существительные и местоимения.
- Неконкретные глаголы.
- Сравнительные (сравнение с умолчанием).
- Модальные операторы невозможности и необходимости (указатели границ опыта).
- Универсальные количественные.

Некоторые инструменты выявления требований

Неконкретные имена существительные и местоимения

Цель вопросов	Вопросы	Примеры
Восстановить исчезнувшую/ упущенную информацию, уточнить реальный опыт.	<u>Кто</u> именно? <u>Что</u> именно? <u>Какой</u> конкретно? <u>Что ты имеешь ввиду?</u>	<ul style="list-style-type: none">• Люди нервничают из-за меня. <u>Какие конкретно люди?</u>• Города опасны для жизни. <u>Какие конкретно города?</u>• Окружающие не любят меня. <u>Кто конкретно?</u>

Некоторые инструменты выявления требований

Неконкретные глаголы

Цель вопросов	Вопросы	Примеры
Уточнить информацию об описываемых процессах и понять, что конкретно происходило или будет происходить.	<u>Как именно?</u> <u>Что именно?</u> <u>Какие действия Вы под этим подразумеваете?</u>	<ul style="list-style-type: none"> Жена одобрила меня. <u>Как конкретно она одобрила Вас?</u> Мои друзья полагаются на меня. <u>Как именно они полагаются на Вас?</u> Я получил информацию. <u>Откуда и как именно Вы ее получили?</u> Подчиненные расстраивают меня. <u>Что они делают, чтобы расстроить Вас?</u>

Некоторые инструменты выявления требований

Сравнительные (сравнение с умолчанием)

Цель вопросов	Вопросы	Примеры
Выяснить эталон сравнения — то, с чем сравнивают.	По сравнению с кем/с чем?	<ul style="list-style-type: none">• Наш товар лучше. <u>Лучше чего?</u>• Здоровье важнее. <u>Важнее чего? По сравнению с чем?</u>

Некоторые инструменты выявления требований

Модальные операторы невозможности и необходимости (указатели границ опыта)

Цель вопросов	Вопросы	Примеры
Выйти за установленные границы возможностей. 1. Уточнение имеющейся информации. 2. Получение доступа к желаемой информации.	<p>1. Что Вам мешает?</p> <p>2. Что случится, если бы Вы сделали/ не сделали? смогли/ не смогли?</p> <p>3. Кто это сказал?</p>	<ul style="list-style-type: none"> Я не могу добиться успеха. <p><u>Что Вам мешает? Что случилось бы, если бы Вы добились успеха?</u></p> <ul style="list-style-type: none"> Я должен быть совершенным. <p><u>Кто говорит, что Вы должны быть совершенным?</u></p> <p><u>Что бы случилось, если бы Вы не были совершенны?</u></p>

Критерии формулировки требований



Достижение выполнения требований зависит от их формулировки, и первый шаг к успеху дела — правильно сформированные требования.