1 Первая лаба

2 Вторая лаба

У ТК не должно быть одинаковых заголовков. Экшены описываются командами: ввести, нажать. Редактируемость и нередактируемость (кликабельность) выделать в один ТК не нжуно. Это все один большой ожидаемый результат главного ТК для главного гуи. ИЛИ в экшене в ТК быть не должно. Общие слова типа "корректное значение"писать нельзя в ТК.

Писать "отображены все поля"нельзя. Перечислять. Ей тяжело посмотреть на картинку в другой вкладке.

Если появляется диалоговое окно, то должно быть его полное описание: заголовок, сообщение, варианты ответа. Должнен быть ТК на ГУИ диалогового окна. Еще 2 ТК, проверяющие нажатие кнопок. ТК на появление диалога тоже должен быть.

На проценты 11 ТК.

Для каждого требования должны быть ТК.

Главный ТК — запуск приложения. В ожидаемом результате нужно описать все, что мы ожидаем увидеть: все элементы ГУИ

Уточнение по поводу всех кнопок и полей (может быть картинка, лол).

Неверное название кнопок — тоже дефект.

Первый шаг — запустить приложение, если тестируем главное гуи. Иначе в предусловии.

Все шаги выполнения ТК (actions) нумеруются, даже если шаг один.

Фича это возможность, не часть гуи Используем декомпозицию для группировки кейсов

Три уровня приоритета тест кейсов. Hight самый важны дефект, который нужно быстрее тестить.

ТК не может ссылаться на несколько фич. ТК может затрагивать несколько фич, но создается только для одной.

На все вопросы должен быть ТК. Ко всем требованиям должен быть ТК.

3 Третья лаба

Нужно делать группировку.

severity — важность для системы priority — определяет срочность выполнения. Приоритет у блокирующих ТК должен быть выше заблокированных.

TC ID — ссылка (одна) на тест-кейс Req ID — ссылка (одна) на фичу.

Pre-condition — копировать с ТК Expected result — копировать с ТК

Статусы тест-кейса (результаты выполнения, Result):

- passed. Полное соответствие
- failed(*номер дефекта*). Несколько фейлов с одним номером дефекта быть не может, так как **дефект описывается по факту каждого фейла**.
- blocked(*номер дефекта*). Возникает, когда не удается выполнить ТК: предусловия или шаги. Должен ссылаться на дефект, который находит другой ТК. Если дефект отсутствует, то можно добавить и выполнить новый ТК, чтобы его найти.

В названии дефекта (Title) причина фейла, не название ТК.

4 Защита

4.1 Что является объектом тестирования?

Объектом тестирования называется то, что мы тестируем, т.е. сравниваем наблюдаемое поведение с ожидаемым.

Объект тестирования определяется уровнем тестирования. В рамках нашего курса мы выполняли интеграционное или системное тестирование, поэтому нашим объектом тестирования была программа.

Термины ниже могут спросить.

- Программа — набор команд и данных, которые позволяют аппаратному обеспечению выполнять вычисления или функции управления.
- Программное обеспечение (ПО, software) — программа или множество программ, которые используются для управления компьютером.
- Системное ПО (system software) — ПО, которое управляет компонентами компьютера (процессор, сетевое оборудование, устройства ввода-вывода) и предоставляет сервисную поддержку прикладному ПО.
- Инструментальное ПО (programming software) — ПО для разработки программ (генераторы документации, текстовые редакторы, компиляторы, отладчики и т.д.).
- Прикладное ПО (приложение, application software) — ПО, предназначенное для выполнения задач, которые напрямую не касаются работы компьютера. Обычно рассчитано на непосредственное взаимодействие с пользователем.
- Программный продукт (ПП) — ПО для широкого распространения или продажи.

4.2 Чем отличается коробочный продукт от заказного?

Коробочный ПП предоставляется с одним набором функций для всех пользователей. В заказном ПП набор функций определяется требованиями конкретного заказчика.

4.3 Перечислите основные этапы процесса разработки ПП. Какова основная задача каждого из них?

Основные этапы процесса разработки ПО. Не обязательно идут последовательно.

- Планирование. Определить методологию разработки (adjile), модель жизненного цикла, распределить роли в команде, сформировать список проводимых работ, назначить каждой работе начало, длительность, стоимость, спрогнозировать возникновение проблем и найти для них решения. Отвечает менеджер.
- Сбор и анализ требований. Изучить идею проекта, собрать требования заказчика, уточнить их, составить требования для разработчиков, определить варианты использования, составить техническое задание, согласовать конечный результат с заказчиком, сформировать дизайн продукта. Отвечает аналитик.
- Разработка архитектуры. Выбрать технологии для реализации, найти ограничения и узкие места продукта, составить модели БД, поведенческие, структурные. Отвечает архитектор.
- Кодирование. Создать минимально жизнеспособный продукт (MVP), написать код для модулей. Отвечает разработчик.
- Тестирование. Определить степень соответствия продукта заявленным требованиям, найти дефекты. Отвечает тестировщик.
- Документрирование. Написать документацию. Отвечает технический писатель.
- Внедрение. Обучить пользователей работе с продуктом, настроить продукт под определенные условия использования, установить продукт.
- Сопровождение. Консультировать заказчика, устранять ошибки, найденные в процессе использования продукта.

4.4 С какими процессами взаимодействует процесс тестирования?

Может со всеми процессами взаимодействовать, как в V-модели.

Тестирование взаимодействует с несколькими процессами в рамках процесса разработки ПП

Сбор и анализ требований $>$ требования для понимания того, какой ПП необходимо будет проверять (основа для создания тестов) $<$ вопросы, уточнения, замечания Разработка архитектуры $>$ информация для структурного тестирования $<$ вопросы, ошибки Кодирование $>$ ПП, который будет тестироваться $<$ дефекты Сопровождение $>$ ПП, который будет тестироваться $<$ дефекты

4.5 Что такое проект? Перечислите основные роли в проекте?

Проект — уникальный процесс, в ходе выполнения которого получают уникальный продукт. Имеются требования, ограничения по ресурсам, времени, контролируется качество.

Команда разработки ПП (роли):

- руководитель проекта,
- аналитик,
- архитектор,
- инженер по тестированию,
- технический писатель.

4.6 Что такое жизненный цикл ПП?

Жизненный цикл ПП — процесс, который начинается в момент принятия решения о создании ПП и заканчивается в момент полного изъятия из эксплуатации этого ПП.

4.7 Какие модели жизненного цикла ПП Вы можете назвать (дайте краткую характеристику каждой модели, когда ее можно применять, когда не следует)?

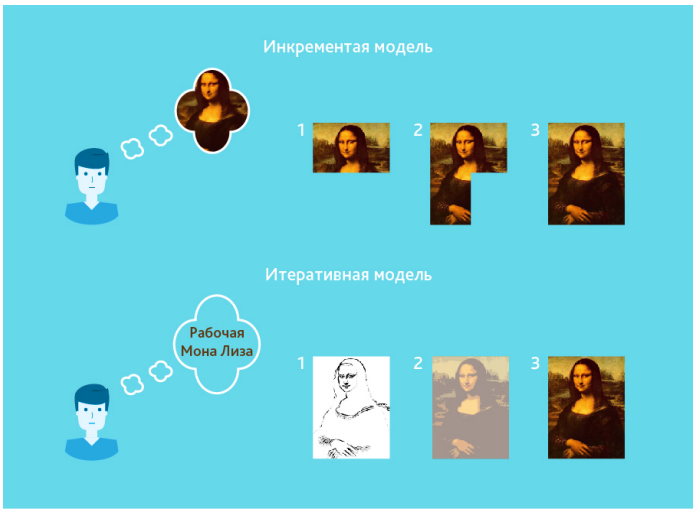
Модели жизненного цикла ПО:

- Каскадная (водопадная). Этапы идут строго друг за другом.

  Список требований строго фиксирован, необходима высокая квалификация исполнителей, сроки и затраты хорошо планируются.

  Дорогая, не подходит, когда тербования часто меняются и не точные, требуется высокая квалификация исполнителей, так как цена ошибки велика

- Итеративная или инкрементная. В первом случае следующая версия создается путем улучшения предыдущей, во втором — дополнения предыдущей.

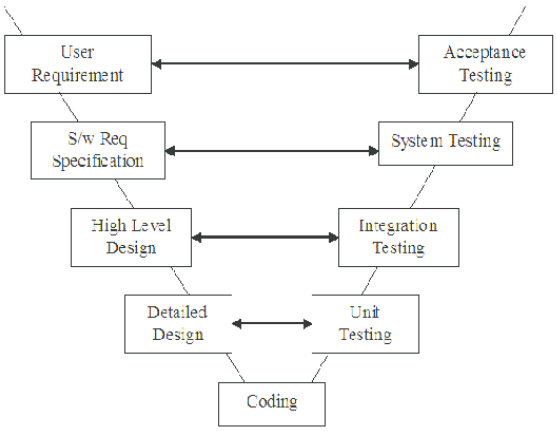


- Спиральная. Сочетает каскадную и итеративную модель. Каждый виток спирали соответствует созданию новой версии, скорость работы определяется длиной витка. Акцент на анализе рисков, который выполняется на каждом витке.

  Хорошо походит, когда что-то новое, новая технология, подход

  Минусы в том, что модель бесконечная, версии продукта нужно поддерживать, сложно оценить точку перехода не следующий цикл.

- V-модель. Идея заключается в параллельном тестировании каждого этапа разработки.



4.8 Какую модель ЖЦ можно применять при условии частых изменений в требованиях? Основные принципы Agile методологий?

4.9 Каковы преимущества и недостатки каскадной модели?

Проста и понятна заказчикам Процесс разработки выполняется поэтапно Невысокие требования к исполнителям Способствует контролю управления проектом Независимость стадий (могут выполнять разные команды) Точное планирование сроков и затрат ***Обнаружение ошибок на ранних фазах (проверка по окончанию каждой фазы)

Высокая стоимость ошибок на фазе интеграции Запаздывание с получением результатов Рост затрат при возврате на одну или две фазы назад Предполагает корректность результатов на предыдущей стадии Не применима в большинстве современных IT проектах, т.к. требования меняются быстро и часто

4.10 Кто определяет цели и задачи тестирования в проекте?

4.11 Кто формулирует требования к продукту?

Заказчик, аналитик

4.12 На что влияет качество существующих процессов?

4.13 Что влияет на выбор модели ЖЦ проекта?

4.14 Что такое тестирование?

Тестирование – процесс выполнения программы с намерением найти ошибки

Проверка ПО на соответствие требованиям к продукту (функциональные, нефункциональные

4.15 Цели и задачи тестирования?

Проверить приложение на соответствие требованиям к функциональности, производительности и т.п.

Найти дефекты (несоответствия требованиям) и предотвратить обнаружение ошибок заказчиком.

а также

выполнить тестирование параллельно с разработкой;

уложиться в сроки;

удовлетворить требованиям к тестированию.

4.16 В чем отличие тестирования от отладки?

В процессе отладки вы исправляете ошибки. В тестировании они тоже выявляются, но не исправляются.

Отладка (debugging) – деятельность, направленная на установление точной природы известной ошибки, а затем - на исправление этой ошибки. Результаты тестирования являются исходными данными для отладки.

4.17 Что такое верификация (verification) и что такое валидация (validation)?

Верификация (контроль,verification) – попытка найти ошибки, выполняя программу в тестовой, или смоделированной, среде.

Верификация дефекта — проверка исправления дефекта. На новой сборке повторно выполняется проблемный текст-кейс. Регрессионное тестирование обязательно выполняется, так как для исправления дефекта были внесены изменения в код.

Исправленный дефект должен быть проверен тестировщиком.

Выполните соответствующий тест на новой версии (build).

Выполните дополнительно тесты для функциональности, в которой был исправлен дефект (регрессионное тестирование).

Выполните дополнительно тесты для функциональности, которая интегрирована с исправленной (регрессионное тестирование).

Валидация (испытание, validation) – попытка найти ошибки, выполняя программу в заданной реальной среде

4.18 Какие подходы (методологии) тестирования вы знаете?

Черный ящик. Заключается в разработке тест-кейсов без рассмотрения исходного кода, мы не используем знания о внутренней реализации продукта.

белый ящик используем знания о внутренней структуре, коде. Юнит тесты, их делают разработчики.

серый ящик черный ящик, при котором мы используем некоторые знания о структуре (базы данных, алгоритм). Чаще всего используется она.

4.19 Какие уровни тестирования вы знаете?

- Модульное (unit testing) — тестирование отдельных юнитов (классы, функции) (белый ящик)
- Интеграчионное (integration testing) — тестирование взаимодействия между юнитами (интерфейсы) (белый ящик)
- Системное (System testing) – тестирование целой системы (работающая интегрированная система) * «черный ящик»серый ящик» (может быть также Системное интеграционное тестирование – интеграция между системами)
- Приемочное (Acceptance testing) – тестирование системы конечными пользователями * на реальном окружении

Чем выше ребования к качеству конечного продукта, тем выше объем тестирования на каждом уровне.

4.20 Какие виды тестирования вы знаете?

По способу использования ПО в процессе тестирования:

- Статическое тестирование – тестирование, в ходе которого тестируемая программа (код) не выполняется (не запускается).
- Динамическое тестирование – тестирование, в ходе которого тестируемая программа (код) выполняется (запускается).

Способ выполнения тестов:

- Ручное тестирование (Manual testing) – тесты выполняются вручную.
- Автоматизированное тестирование (Test Automation) – выполнение тестов автоматизировано.

Тестирование после внесения изменения в код:

- Регрессионное тестирование (Regression testing) – повторное выполнение тестов после внесения любого изменения в код (рефакторинг) (тестирование функциональности, которая была уже протестирована до внесения изменений).
- «СМОУК» тестирование (Smoke Testing) – набор базовых тестов для принятия решения о готовности продукта к дальнейшему использованию после внесения изменений.

По характеристике продукта:

- Функциональное тестирование (Functional testing)
- Тестирование GUI (GUI testing)
- Тестирование совместимости (Compatibility) testing
- Тестирование локализации (Localization testing)
- Тестирование удобства использования (Usability testing)

- Тестирование безопасности (Security testing). Авторизация.
- Тестирование производительности (Performance testing). Время при стандартной нагрузке
- Тестирование нагрузки (Load testing). Время при максимально повышенной
- Стрессовое тестирование (Stress testing). Нагрузка превышена, свет выключается, соединение разрывается, данные сохраняются.

По времени выполнения:

- 'Альфа' тестирование (Alpha testing) – тестирование в процессе разработки.
- 'Бета' тестирование (Beta testing) – тестирование выполняется пользователями (end-users).

4.21 Перечислите основные этапы процесса тестирования.

- Планирование (Test Planning). Ознакомление с требованияминализ требований

  Выбор средств тестирования и автоматизации тестирования

  Подготовка тестового окружения

  Разработка тест плана
- Разработка тестов (Test Case Design)
- Аудитевью тестов (Test Case Review). Оценка покрытия ТК, качество описания, устранение неточностей, неоднозначностей
- Выполнение тестов (Test Case Execution). Выполнение, сравнение с результатом, создание отчета о найденных дефектах.
- Верификация исправленных дефектов (BugFixes Verification).
- Метрики и отчеты (Test Reporting and Metrics Collection)

  Метрики – показатели, которые помогают оценивать качество, прогресс и принимать решения

4.22 Что такое требования к ПО?

Требование к ПОП задокументированная уникальная потребность (необходимость) того, что должен делать конкретный продукт или каким он должен быть. задокументированное ожидание заказчика совокупность утверждений относительно атрибутов, свойств или качеств программной системы, подлежащей реализации.

Спецификация - инженерный термин, обозначающий набор требований и параметров, которым удовлетворяет некоторая сущность (в т.ч. часть технического задания, ТЗ).

User Story (История Пользователя) – требование к ПО, сформулированное в виде одного или двух предложений на языке пользователя, который будет использо-

вать данный продукт (Agile: XP, SCRUM).

4.23 Какие бывают типы требований?

Требования к Проекту

Требования к Процессу

Требования к Продукту

Бизнес-требования - определяют назначение ПО, отражают бизнес-цели Пример: «система должна сократить срок оборачиваемости обрабатываемых на предприятии заказов в три раза»

Бизнес-правила - положение, определяющее или ограничивающее какие-либо стороны бизнеса (законы, регламенты и т.п.).

Пользовательские требования - конкретные способы использования продукта конечным пользователем.

граничения - внешние по отношению к системе условия или принятые ранее решения, которые модифицируют требование (-я), сужая выбор возможных решений.

Внешние интерфейсы - описание интерфейса между системой и пользователем, другой системой или оборудованием.

Системные требования – требования к архитектуре, способам реализации, окружению.

Функциональные требования (functional requirements, software functional requirements) охватывают предполагаемое поведение системы, определяя действия, которые система способна выполнять для того, чтобы пользователь смог выполнить свои задачи. Пример: «Система должна по электронной почте отправлять пользователю подтверждение о заказе». «Заказ может быть создан, отредактирован, удален и перемещен с участка на участок

Нефункциональные - требования к производительности, безопасности, совместимости (окружение), удобству использования, надежности и т.п.

4.24 Что такое Test Case?

Тест-кейс (ТК, test case) — единица тестирования, целью которой является нахождение одного дефекта (несоответствия поведения системы требованию). Последовательность тест-кейсов называется **тестовым сценарием**.

4.25 Что должно быть в описании тест-кейса (приведите пример)?

Тест-кейс: входные условия (pre-conditions), действия (actions), ожидаемый результат (expected result)

4.26 Что такое Test Matrix (Тестовая матрица)?

создание и описание тестов в Тест Матрице или тестовой базе данных (в специальной среде)

4.27 Что является источникомснованием для создания тест-кейсов?

создаются на базе списка требований (функциональные и нефункциональные)

4.28 Какие техники создания тест-кейсов Вы знаете? Перечислите и опишите основную идею каждой из них.

Техники тестирования черного ящика:

- Эквивалентное разбиение. **Классом эквивалетности** (КЭ, equivalence class) называется множество данных, которые в данном контексте считаются неразличимыми. Один ТК должен покрывать один КЭ. Например, если мы тестируем функцию $f : Z \to Z$, то достаточно использовать одно целое число в качестве аргумента, чтобы понять, что функция работает.

  КЭ формируются из имеющихся требований и группируются в правильные и неправильные классы. Правильный класс содержит данные, которые не нарушают условия, неправильный — данные, которые нарушают **одно** условие (описывают строго один недочет).

  Может возникнуть желание описать несколько правильных классов одним регулярным выражением, но это не информативно. Если условия допускают несколько наборов данных, отличных друг от друга по какому-нибудь признаку, то следует проверить каждый из этих наборов.

  Обычно классы определяются при помощи текстового описания. Стандарта описания нет, поэтому следует избегать формальных обозначений и общих фраз, чтобы не допустить неоднозначного понимания. Описание класса следует приводить подробно, чтобы не было необходимости смотреть в требования, так как это отнимает время. В описании не следует использовать понятие бесконечности, так как это затруднит тестирование.

  Пример описания: «значение содержит меньше 7 или больше 12 символов, имеется хотя бы одна заглавная буква русского алфавита, хотя бы одна цифра, хотя бы один символ из списка: ?, %, *, возможно наличие строчных букв русского алфавита, других символов значение не содержит».

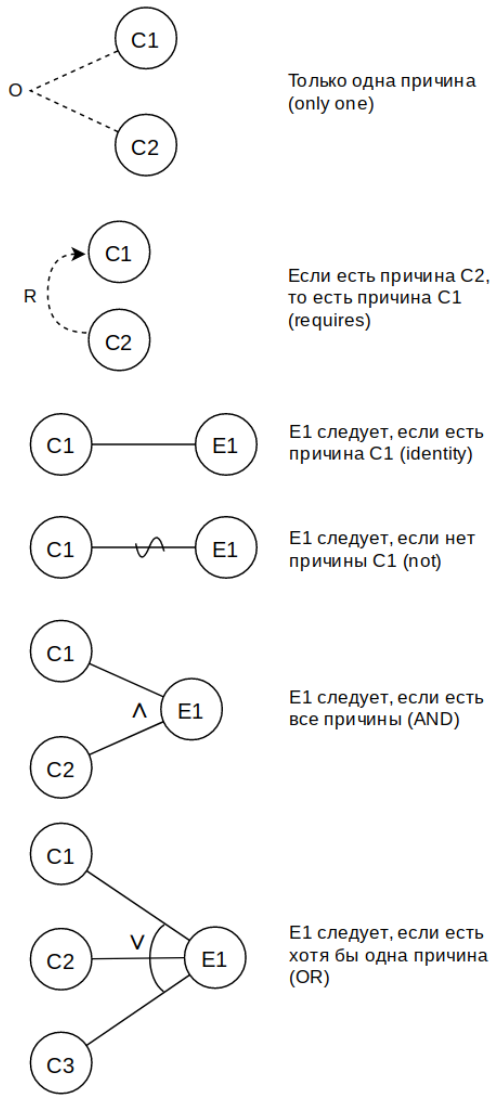
- Анализ граничных значений (boundary value testing). Такие значения возникают на верхних и нижних границах входных КЭ. Если граница одна (например, слово состоит из b символов), то граничными значениями будут: $b - \epsilon$, b, $b + \epsilon$. Если границы две (например, номер должен принадлежать отрезку от a до b), то граничные значения: $a - \epsilon$, a, b, $b + \epsilon$.

- Анализ причинно-следственных связей (decision table testing). Тестировщик формирует набор причин и следствий. Под причиной понимается входной КЭ, под следствием — результат системы на данном КЭ. Результатом анализа является таблица решений, которая отражает все возможные комбинации причин и следствий. Если имеются невозможные комбинации, то это поясняется в примечании к таблице. Пример упрощенной небинарной таблицы:

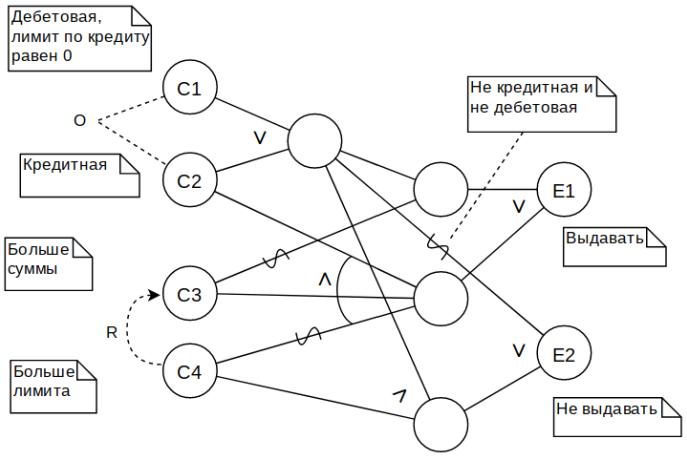
  ТК создается для каждого столбца.

  Функциональная диаграмма (cause-effect graph) позволяет упростить или автоматизировать построение сложной таблицы решений. Нотация диаграммы:

Кейс	1	2	3
Условие			
Пол	М	М	Ж
Возраст	Мол.	Зрел.	Пож.
Действие			
Предложить напиток	Сок	Кофе	Чай
Дать кредит	0	1	0



Пример диаграммы для функции выдачи денег у банкомата:



- Предположение об ошибке. Тестировщик с большим опытом формирует список возможных ошибок без использования конкретных методов (эвристический подход).

4.29 Что такое дефект ПО («ошибка», «баг»)?

Дефект (defect bug) – реальный результат выполнения теста не равен ожидаемому результату

Дефект – несоответствие требованию.

Дефект – несоответствие полученного результата при выполнении тест-кейса ожидаемому результату этого тест-кейса (тест-кейсы создаются по требованиям).

4.30 Какая информация обязательно должна быть в отчете о дефекте, чтобы его можно было воспроизвести?

Отчет о дефекте должен быть понятным и однозначно воспроизводящим проблему.

- Номер дефекта. Уникальный идентификатор.
- Заголовок. Краткое описание дефекта (что не так).
- Описание. Предусловия, шаги, полученный и ожидаемый результат (берется информация из соответствующего тест-кейса).
- Версия продукта, в которой обнаружен дефект (affected version).
- Окружение. Тестовое окружение, в котором запускается приложение.
- Серьёзность (критичность, severity). Степень влияния дефекта на работоспособность системы.
- Приоритет. Порядок исправления.
- Автор. Кто нашел дефект.
- Дата и время. Когда был обнаружен дефект.

- Ответственный. Кто обрабатывает данный отчет о дефекте в текущий момент.
- Статус. Этап жизненного цикла дефекта.
- Версия продукта, в которой дефект будет исправлен (fix version).
- Компонент. Ункциональный модуль, в котором найден дефект.

4.31 Приведите пример жизненного цикла дефекта.

Каждый дефект имеет свой жизненный цикл (создание отчета о дефекте ->...-> закрытие дефекта).

Статус + резолюция (причина) - примеры возможных типичных статусов:

Новый — дефект зарегистрирован;

В разработку — отправлен на исправление, назначен на разработчика (В разработке – взяли в работу)

На анализ – отправлен для уточнения аналитикам (На анализе – взяли в работу)

В тестирование — отправлен в тестирование с определенной резолюцией для обработки, назначен на тестировщика (В тестировании – взяли в работу) Резолюции: исправлен (указывается версия, в которой исправлен дефект) дубликат (повторяет дефект - ID, найденный ранее) не ошибка (работает в соответствии с требованиями) недостаточно информации (запрос дополнительной информации об условиях, в которых дефект проявляется) и т.п.

Закрыт – после верификации тестировщики могут закрыть дефект с определенной резолюцией Резолюции: исправлен (указывается версия, в которой исправлен дефект) дубликат (повторяет дефект - ID, найденный ранее) не ошибка (работает в соответствии с требованиями) и т.п.

4.32 Что такое Тест План?

Документ, который составляется на этапе планирования тестирования.

Тест План:

Стратегия тестирования, цели и приоритеты Типы тестирования Роли и обязанности Тестовое окружение (конфигурации) Автоматизированные средства для тестирования Отчеты по тестированию, метрики Процедура тестирования Расписание

4.33 Что такое качество ПП?

Степень соответствия присущих характеристик требованиям.

4.34 Как можно описать качество ПП?

4.35 Назовите характеристики (атрибуты) ПП?

4.36 Что такое «внешнее качество»?

Качество для заказчика (внешнее качество) Продукт должен быть удобен для использованияизнеса (“Fitness for Use” - Joseph Juran, Usability) Отсутствие ошибок

Производительность

4.37 Что такое «внутреннее качество»?

Качество для производителей (внутреннее качество) Соответствие требованиям (“Conformance requirements” - Phillip Crosby) Удобная архитектура Простота модификации

4.38 Что такое Quality Assurance (Обеспечение качества)?

QA – это проактивный процесс, направленный на предотвращение возможных дефектов. Он выполняется во время разработки продукта.

Улучшить процессы разработки и тестирования и таким образом избежать появления дефектов.

4.39 Каковы основные задачи, цели и Quality Control?

QC – это реактивный процесс, целью которого является подтверждение качества конкретного продукта посредством тестирования, выявления и устранения неисправностей. Он проводится после того, как продукт был разработан.

Выявить недостатки продукта после того, как его разработка завершена, но до его релиза.

4.40 К чему относится тестирование: Quality Assurance или Quality Tracking?

QA возможно. Что такое QT не понятно

4.41 Кто отвечает за качество продукта в проекте?

4.42 Что такое стоимость качества?

суммарная стоимость издержек на: инвестиции в предупреждение несоответствий требованиям оценку продуктаервиса на соответствие требованиям исправление несоответствий требованиям

4.43 Какие системы управления качеством вы знаете?

CMMI

4.44 Что такое Комплексная модель производительности и зрелости процессов (Capability Maturity Model Integration)?

доверяются только компаниям, прошедшим сертификацию на соответствие какому-либо международному стандарту, зачастую им становится модель CMMI.

набор моделей (методологий) совершенствования процессов в организациях разных размеров и видов деятельности. CMMI содержит набор рекомендаций в виде практик, реализация которых, по мнению разработчиков модели, позволяет реализовать цели, необходимые для полной реализации определенных областей деятельности.

Уровень зрелости – это главный, итоговый показатель оценки по модели CMMI.

. Качественные продукты данных организаций производятся не за счет устойчивых и отлаженных процессов, а благодаря титаническим усилиям отдельных личностей. В случае ухода таких людей очень тяжело повторить успешные проекты. На данном этапе очень тяжело предсказать производительность процессов, протекающих в организации.

Уровень зрелости 2 – (повторяемый) управляемый уровень. На данном этапе основные процессы описаны, их, возможно, использовать неоднократно. Однако процессы все же имеют некоторую долю реактивности в своей сущности.

Уровень зрелости 3 – определенный уровень. В этом случае процессы определены. Установлены стандарты в пределах организации. На данном этапе процессы описаны не на уровне отдельного проекта, а на уровне всей организации. Люди легко заменяются.

Уровень 4 — (управляемый) собираются метрики, аналитка, количественное управление проектом и организацией.

5 уровней.(оптимизирующий) На последнем процессы посноянно улоучшаются и оптимизируются.