Документирование как основа тестирования

Определение теста по IEEE

- *ТЕСТ* – набор, состоящий из одного или нескольких тестовых примеров и процедур
- *ТЕСТОВАЯ ПРОЦЕДУРА* – перечень большого числа этапов со своими входными данными, каждый из которых имеет свои промежуточные ожидаемые результаты
- *ТЕСТОВЫЙ ПРИМЕР* – комбинация специфических входных данных и ожидаемых результатов.

ПРИМЕЧАНИЕ. В современной IT-промышленности терминология, касающаяся QA и тестирования, весьма запутана. Например, термины тест, тестовая процедура и тестовый пример путают, используют в разных контекстах по-разному или попеременно.

Особенно плохо дело обстоит с русскоязычной терминологией.

Общепринятое определение теста

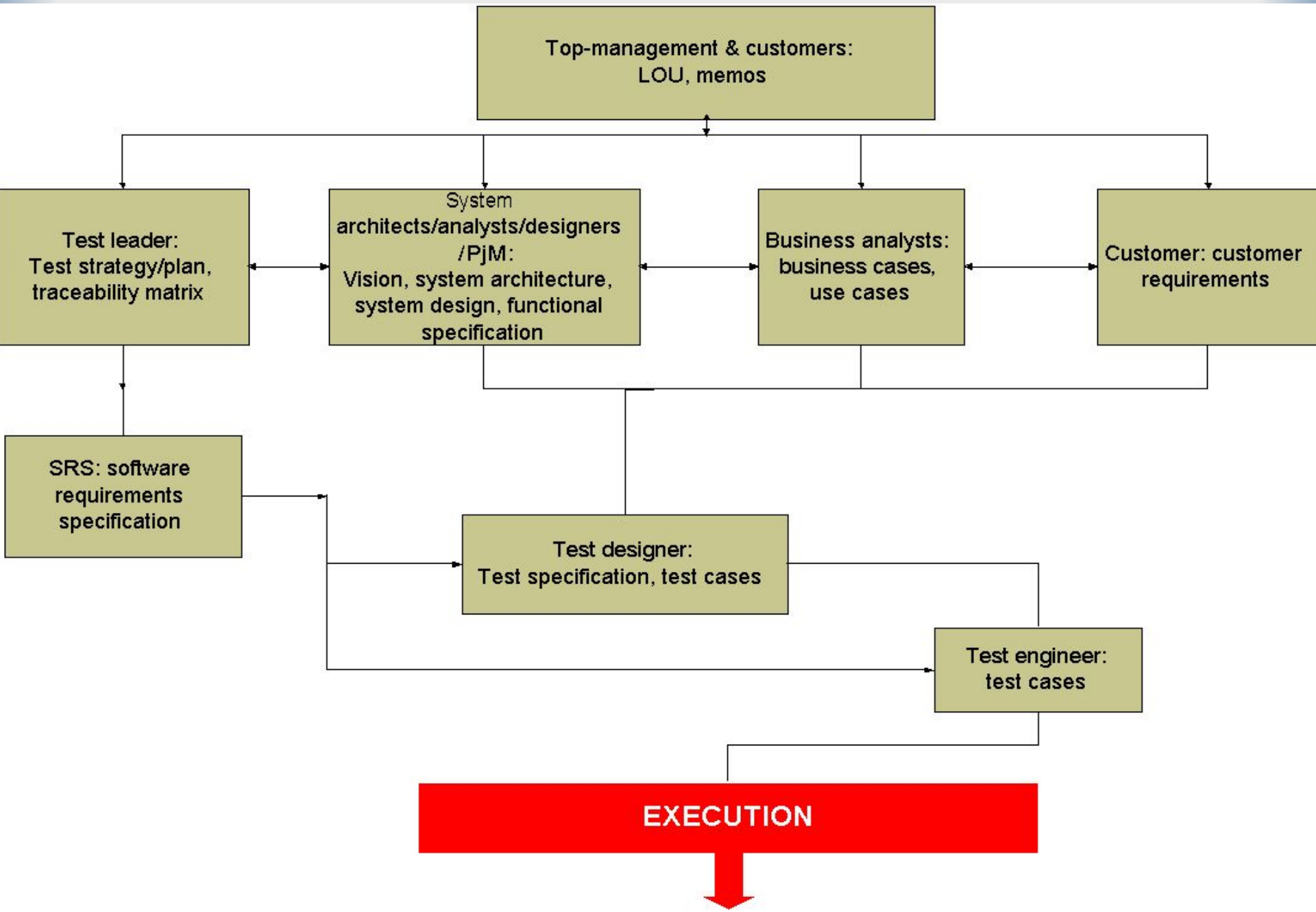
- В настоящее время слова ***тест*** *и* ***тест-кейс*** *(test case,* ТС, тестовый пример) часто используются как синонимы.
- ***Тестовый пример*** – это совокупность
  - **Конфигурации системы**
  - **Входных данных**
  - **Начальных условий**
  - **Сценария** (алгоритма действий). Может содержать условия и переходы, однако лучше, чтобы он был линейным и достаточно коротким
  - **Ожидаемых результатов** (и конечного состояния, которое может отличаться от начального состояния/условий)

Типичный набор документов

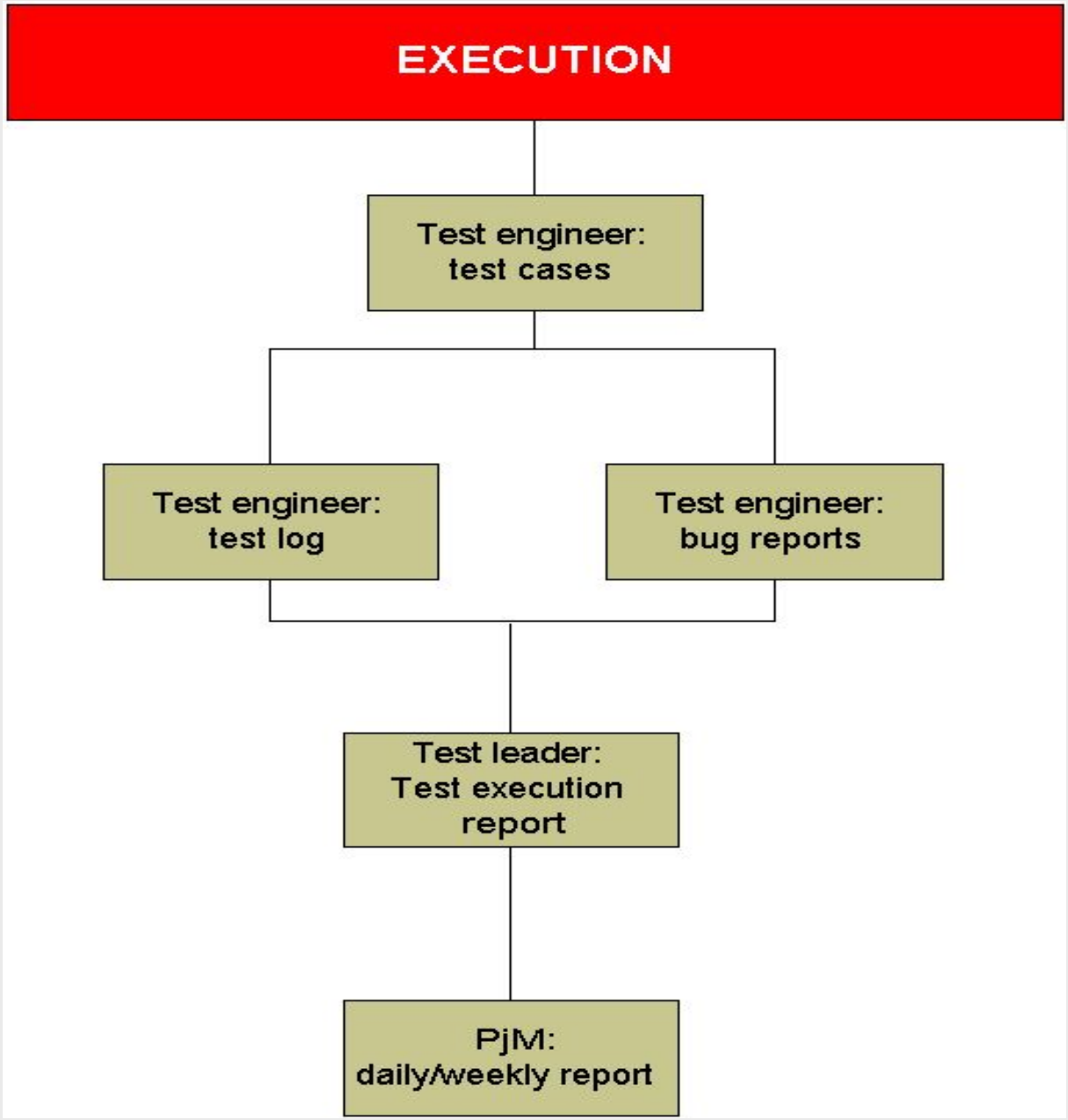
(IEEE Std 829-1998)

- Функциональная спецификация (Functional specification, FS)
- Спецификация программных требований (Software requirement specification, SRS)
- Traceability matrix (матрица прослеживаемости)
- Тест-план (Test plan, test strategy - TP)
- Тестовая спецификация (Test specification, TS)
  - Test cases, Тестовые процедуры
- Test log
- Bug report

«Классический» проект: разработка и кодирование



«Классический» проект: тестирование



Источники информации для тестировщика

- Спецификация
- Личное общение с руководством и программистами
- Документация (черновики руководства пользователя, заметки разработчиков)
- Исследование (результат собственного опыта, полученного в ходе экспериментов над программой)
- Аналогичные проекты

Пример Functional Specification

4.1. Asset management XPe devices

ID	Description	Priority	Source	Dependencies	Status
ID: xpe -man-auto_registaration	**WXPEARMS product should provide a mechanism for automatic registration new XPe devices with the management console.** On start up XPe-device should ask administrative engine about its own registration status. If device is already registered with management console nothing to do. Otherwise XPe-device should send auto-registration request (with detailed information about current XPe-device configuration) to administrative engine. Administrative engine should accept this auto-registration request, update registration status of device and add this device to physical view of management console.	Must	CCI	[xpe-man-physical_view]	
ID: xpe-man-manual_registaration	WXPEARMS product should provide a mechanism of manual discovering and registration XPe devices with the management console. Administrator should be able to initiate operation of discovering XPe devices on the network to have up-to-date information about current XPe devices configuration. By request of administrator WXPEARMS product should send broadcast request to XPe devices on the network, get	Should	SWA	[xpe-man-physical_view]	

Определение объемов тестовых работ

- Тестируйте в первую очередь требования с наивысшим приоритетом
- Тестируйте новые функциональные возможности и программный код, который изменялся
- Используйте разбиение на эквивалентные классы и анализ граничных значений
- Тестируйте те участки, в которых наиболее вероятно присутствие проблем
- Сосредоточьте свое внимание на функциях и конфигурациях, с которыми наиболее часто будет иметь дело конечный пользователь

Тестовый план

Это документ, включающий:

- объем
- ресурсы
- календарный план работ по тестированию
- выполняемые тесты
- тестируемые элементы
- задачи тестирования
- ответственные сотрудники
- вероятность возникновения непредвиденных обстоятельств и меры, которые потребуется при этом принимать

(стандарт ANSI/IEEE 829-2983 for Software Test Documentation)

Назначение тестового плана

- служит для поиска ошибок
- облегчает управление работами и контроль хода их выполнения
- облегчает организацию технических аспектов тестирования
- помогает организовать и скоординировать усилия сотрудников, разрабатывающих и тестирующих программный продукт
- повышает эффективность и полноту тестирования
- документация должна быть не объемной, а эффективной. Любые составляющие плана, не помогающие в поиске ошибок и организации тестирования, являются пустой тратой ресурсов

- продукт (стОит дороже)
- рабочий инструмент

Составлени тест-план

Получить документ с требованиями (технические условия)

- Документ определения требований
- Документ спецификации требований
- Матрица прослеживаемости требований

Определение стратегии тестирования

- Объем испытаний (что должно тестироваться)
- Методика тестирования (как будет выполняться тестирование)
- Критерий вхождения и выхода из испытаний и точки контроля качества
- Стратегия автоматизации

Определение состава и структуры испытательной системы

- Архитектура тестов
- Условия испытаний
- Конфигурации средств тестирования

Оценка *объемов работ* по тестированию

- Определение задач (структура разбиения работ на зад
- Оценка объемов работ в человеко-днях и длительность выполнения работ в неделях
- Разработка графика работ по тестированию и таблица наиболее важных этапов этих работ
- Оценка рисков выполнения графиков работ и составлен планов мероприятий по смягчению последствий от овеществления этих рисков

Подготовка и утверждение плана проведения испытаний

- Подготовка документов с описанием страт системы, номенклатуры и графика работ
- Анализ и утверждение плана проведения испытаний совместно с коллективом разработчиков проекта

На этап проектирования и реализация тестов

Совершенствование тестового плана

Как правило, применяется эволюционный подход (проведение тестирования параллельно с разработкой его плана)

Первый этап - начальная разработка:

1. Проработка спецификации / пользовательской документации
2. Первая версия списка функций программы
   (полнота списка определяет полноту тестирования)
   (список будет постепенно расширяться)
3. Анализ входных данных и ограничений
   (простейший анализ граничных условий)

Направления развития плана

1. Наиболее вероятные ошибки
   (чем больше ошибок обнаружено в некоторой области программы, тем больше их там же)
2. Наиболее заметные ошибки
   (пользователю)
3. Наиболее часто используемые области программы
4. Отличительные особенности программы
   (то, что отличает от конкурентов)
5. Самые сложные аспекты для тестирования
6. Самые понятные функциональные области

Компоненты тестового плана

списки таблицы планы матрицы

отчетов и экранных форм

вх. и вых. переменных

возможностей и функций

файлов

сообщений об ошибках

совместимого оборудования

совместимых программ

публикуемых документов

конфигураций совместимой операционной среды

перечень материалов

отчетов

вх. и вых. значений

ввода-вывода

решений

клавиатурных комбинаций

совместимых принтеров

диаграмма граничных значений

диаграмма потоков данных

иерархический список функций

Матрицы:

- аппаратной и программной совместимости
- аппаратных конфигураций
- операционных окружений
- комбинаций входных значений
- сообщений об ошибках и клавиатурных комбинаций

- Кроме того, ведется матрица прослеживаемости требований (отображение каждого требования на тест-кейсы).

Пример таблицы ввода-вывода

Входная переменная	Выходная переменная	Связь
Цена_товара	Цена_товара_в_счете	= Цена_товара
	Общая_стоимость	Сумма стоимостей заказанных товаров
	Налог_с_продаж	7% от Общая_стоимость

Иерархический список функций системы

1. Перечень всех высокоуровневых действий пользователя
2. Подфункции всех функций (все доступные опции и варианты)
3. Детализация до элементарных логических действий программы
4. Перечислить входные и выходные условия для каждой функции и подфункции
5. Список всех способов диалога с программой при выполнении каждой из функций (клавиатура, мышь)

Каждая строка этого списка в конце концов преобразуется в тестовый пример

Разделы тестового плана по стандарту

- идентификатор
- введение
- тестируемые элементы (программные компоненты, подлежащие тестированию)
- тестируемые функции
- нетестируемые функции
- подход к тестированию (кто, виды работ, технологии и средства, критерии, крайние сроки)
- критерии прохождения тестов
- документация
- необходимое оборудование
- календарный план
- ответственность
- …

Test Specification – обязательный документ

- Test Specification – документ, *обязательный* к исполнению: *все,* что там написано – д.б. выполнено
- Оптимизация Test Specification – одна из основных задач
- Вообще набор видов тестирования содержится в Test Plan’е

Структура Test specification

Как у обычного проектного документа:

- Заголовок
- Авторы
- История модификации
- Логотипы
- Сведения о степени конфиденциальности
- Содержание
- Введение
- Фактическая часть – тестовые примеры (test cases)

Пример Test specification

2.1.9. XPE-INST-CDINST09. correct moving between setup screens

Revision: 1.0

Purpose: Check setup program screens order.

Env.Needs: MC setup is started. Test configuration 1 - test configuration 4.

Input: Click "Next" in any screen.

Output: next screen is expected.

Input: Click "back" in any screen.

Output: previous screen is expected.

Input: Click "cancel" button in any screen.

Output: "Exit installation" screen is expected.

Более подробно о создании тест-кейсов - далее

2.1.10. XPE-INST-CDINST10. MS .NET detection and installation

Revision: 1.0

Purpose: Check setup ability of .Net detection and installation.

Env.Needs: MS .NET has not been installed yet. Test configuration 1 - test configuration 4.

Input: Run setup.exe from mapped drive where MC distributive is placed.

Output: MS .NET installation screen is expected. Continuation of setup work after .NET installation is expected.

2.1.11. XPE-INST-CDINST11. Microsoft .NET detection and installation

Revision: 1.0

Test Log

- Список тестовых примеров
- Список версий продукта (билдов)
- Отметки об успешном или неуспешном прохождении

	Build 54	Build 57	Build 59
XPE-INST-MSINSTALLER01	ok	ok	ok
XPE-INST-CDINST01	ok	ok	ok
XPE-INST-CDINST02			ok
XPE-INST-CDINST03	ok	ok	ok
XPE-INST-CDINST04	ok		
XPE-INST-CDINST05			
XPE-INST-CDINST06			
XPE-INST-CDINST07	ok	ok	ok
XPE-INST-CDINST08			
XPE-INST-CDINST08A			
XPE-INST-CDINST09		ok	
XPE-INST-CDINST10			
XPE-INST-CDINST11	ok	ok	ok
XPE-INST-CDINST11a	ok	ok	
XPE-INST-CDINST12		ok	ok
XPE-INST-CDINST13		ok	
XPE-INST-CDINST14		ok	
XPE-INST-CDINST15			
XPE-INST-CDINST15a		ok	ok
XPE-INST-CDINST15b			
XPE-INST-CDINST15c		ok	ok
XPE-INST-CDINST15d			
XPE-INST-CDINST15e			
XPE-INST-CDINST16		ok	
XPE-INST-CDINST17		ok	ok
XPE-INST-UNINST01	ok		ok
XPE-INST-UNINST02	ok	ok	ok

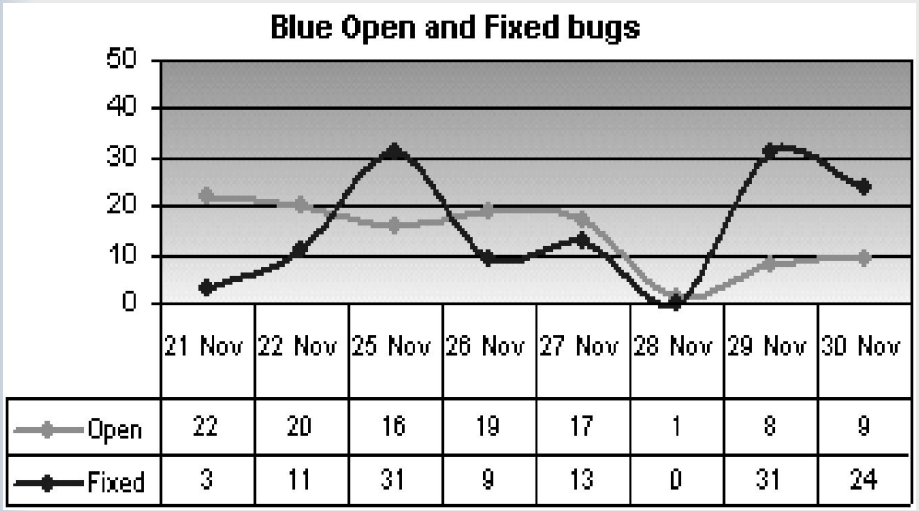
Test Log – дополнительные поля

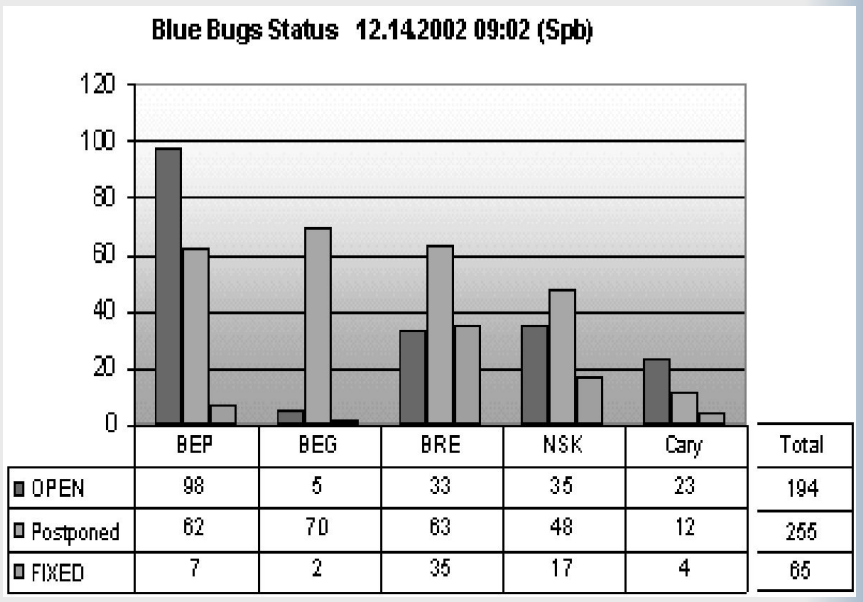
- Разбиение по платформам, конфигурациям, средам выполнения, ...
- Приоритеты
- Группы и подгруппы
- Детализация результатов выполнения
  - Критический/некритический/косметический
  - Номер ошибки в системе сопровождения ошибок
  - Комментарии относительно хода выполнения

Выводы по результатам тестирования

- Тестирование пройдено/не пройдено (для билда)
- Статистика:
  - Время выполнения
    - В среднем на тестовый пример (возможно доп. разбивка по подгруппам)
    - На каждый билд
    - На последний билд
    - На каждой платформе
  - Процент покрытия функциональности/тестовых примеров
    - по каждому билду
    - По каждой платформе
    - По последнему тестируемому билду
    - .....

Примеры отчетов (Терехов А.А.)





- Такие отчеты могут выполнять две основных функции:
  - фиксировать состояние в данной контрольной точке, т.е. отчет отвечает на вопрос вида "да или нет" — выполнены необходимые для этой точки условия или нет;
  - показывать динамику процесса и переход от одной его фазы к другой, т.е. отчет предоставляет информацию для принятия решения о возможности перехода от одного этапа процесса к последующему.

Разработка тестовых примеров (ТС)

Пример ТС

TC ID/Priority	CCPG0002	1

IDEA: Оплата может быть произведена картой **MasterCard**
SETUP and ADDITIONAL INFO:
Эккаунт: testuser1/pa$$wOrd
Данные карты:
Номер: **3333-7112-4444-7844** Окончание действия: **12/08**
CVV2: **676**

Revision History	
Created on: 11/17/2003 by O.Тарасов	Новый тест-кейс
Modified on: 11/26/2003 by И. Новикова	Шаги были упрощены, чтобы сделать тест-кейс более удобным для поддержки
Modified on: 01/17/2003 by И. Новикова	Изменение шагов и второй ожидаемый результат с целью удостоверения в снятии денег со счета

Execution part	
PROCEDURE	**EXPECTED RESULT**
1. Запиши баланс счета карты 2. Открой www.main.testshop.rs 3. Войди в систему. 4. Найди любой товар. 5. Добавь товар в корзину. 6. Произведи *оплату картой* из секции SETUP and ADDITIONAL INFO (!!! запиши полную сумму заказа:). 7. Запиши номер заказа 8. Запроси базу данных с SQL1.	> "20"
9. Запиши баланс счета карты	> Шаг 1 - Шаг 6

Структура тестового примера (ТС) - основное

- **Идентификатор** (уникальный)
- Название
- Автор
- Название проекта
- Цель (идея ТС, краткое описание)
- Ссылки (в т.ч. на спецификацию)
- Среда выполнения (setup & additional info)
- **Пошаговое описание**
- **Критерий выполнения (ожидаемый результат)***

* Лучше, когда ожидаемый результат один, но м.б. и несколько.

Структура тестового примера – дополнительные поля

- Журнал изменений (created… modified… change…)
- Метка (для конфигурационного менеджмента)
- Краткое описание
- Полное описание
- Приоритет
- Статус (new, modified, retired)
- Название модуля

Улучшение поддерживаемости тест-кейса

1. Сделать тест-кейс data-driven (по возможности вынести конкретные данные в «шапку», чтобы их было легко изменить).
2. Не описывать шаги по явно очевидным сценариям (например, логин, если проверяется не он).
3. Не давать конкретных деталей, если они не играют роли при исполнении тест-кейса (например, имя товара).
4. Вынести во внешний документ повторяющиеся сценарии (например, семь шагов оплаты).

(из Р. Савина)

Пример

Test Case ID	Priority	Card	Card Number	Card Expiration date	Card CVV2	Expected Result
CCPG0001	1	VISA	9999-5148-2222-1277	12/07	778	10
CCPG0001	1	MasterCard	3333-7112-4444-7844	12/08	676	20
SWPL0001	1	Switch	3333-1988-4444-5699	12/05	451	30

Другое оформление ТС

IDEA: Оплата может быть произведена картами из Таблицы 1. Для каждого тест-кейса из Таблицы 1: 1. Запиши баланс счета карты :
www.main.testshop.rs/<четыре_*последних цифры_карты*>/balance.htm 2. Открой www.main.testshop.rs. 3. Войди в систему как testuser1/paSSwOrd. 4. Найди любой товар. 5. Добавь товар в корзину. 6. Произведи *оплату картой* (!! !запиши полную сумму заказа:).
7. Запиши номер заказа
8. Запроси базу данных: select result from cc_transaction where id = <номер заказа>; Сравни с Expected resultl.
9. Запиши баланс счета карты
Шаг 1 - Шаг 6

К чему необходимо стремиться при создании ТС

1. Независимость тест-кейсов друг от друга (отсутствие ссылок на другие тест-кейсы; независимость от "следов", оставленных другими тест-кейсами в ПО или базе данных)
2. Четкая формулировка шагов (хороший ТС может без труда воспроизвести другой человек, а также вы сами через год; излишние детали тоже ни к чему).
3. Четкая формулировка идеи и/или ожидаемого результата (ожидаемый результат — это информация, на основании которой, вкупе с фактическим результатом, принимается решение об исходе тест-кейса. Следовательно, точность и четкость в формулировке ожидаемого результата играют важнейшую роль. Не рекомендуется отсылка к внешнему документу).

Отладка тест-кейсов

В первый раз тест-кейсы должны исполняться их автором, задача которого:

- если необходимо, добавить новые тест-кейсы;
- если необходимо, внести изменения по существу, например если при создании тест-кейса тестировщик неправильно понял спек;
- если возможно, удалить лишние (дублирующие) тест-кейсы;
- сделать тест-кейсы более удобными для поддержки;
- отшлифовать формулировки.

Примеры тест-кейсов

##	**Test point / Description**
	CTC#1.19 “Location register verification” Description: TC is to verify data from “Location” register. PURPOSE: To verify data from “Location” register, check values and the type of fields. DataSet: DDEMO2
1.	Open Location register: **Registers->Location data->Location**
2.	Add Location by pressing **add** button. Take values from below table. Write value.
3.	Press **Save** button
4.	**Repeat this test case with min, max and invalid values.**
3.	Close Location register by pressing **Exit** button.
4.	**End.**

##	Test point / Description	Comment
	CTC#1.1 "Date of closing the books register verification" Description: TC intended to verify data from "Date of closing the books" register with min, max and invalid value. PURPOSE: To verify data of "Date of closing the books" register, check values and the type of fields. DataSet: DDEMO2	Language is English in General and Unit settings. The user identifier and password is '1111'.
1.	Open Date of closing the books register: **Registers->Date of closing the books**	The Last Date of closing the books is 31.12.2004
2.	Add Date of closing the books register by pressing **add** button. Date of closing the books = 30.13.2005	
3.	Press **Save** button	
4.	Press **Cancel** button	
5.	Add Date of closing the books register by pressing **add** button. Date of closing the books = 1.1.2005	
6.	Press **Save** button	
7.	Press **Delete** button	
8.	Add Date of closing the books register by pressing **add** button. Date of closing the books = 30.6.2006	
9.	Press **Save** button	
10.	Press **Delete** button	

	Test point / Description	Tes ted	Error /r Req.	Comment
	CTC#16.1.1 "Security – Login form" Description: TC is intended to verify the new Login form of ZZZZZ application Predefined conditions: User "1111" with password = 1111 exists in the system			
	Run ZZZZZ application			Login form should be displayed - Check that the correct caption is written in the main frame of the application: Economa Fixed Assets ZZZZZ 2.0 - Check that the form caption is "Login"
	Select "Windows authentication"			Check that "Login name" and "Password" fields are disabled
	Select "SQL Server authentication"			Check that "Login name" and "Password" fields are enabled

	Press “More >>” button			Check that the hidden fields are displayed: - “SQL server”; - “Database” Check that label “More>>” is changed to label “<<Less”
	- Register the correct SQL server name if the field is empty - Register the correct name of the Common database in the “Database” field			
	Register user name = “1111” Do not register password Press OK button			The error message box should be displayed: “Wrong login or password”
	Press OK button in the error message			
	Register password “1111”			

	Change the name of the server to the nonexistent one, for example add “1” to the end of the server name Press OK button Wait…			The error message box should be displayed: “Wrong SQL Server name!” [server doesn’t exist or…]” (the language of the text in the [] depends on the regional settings)
	Press OK button in the error message			
	Change the name of the server to the correct one			
	Change the name of the database to the nonexistent one, for example add “1” to the end of the database name Press OK button Wait…			The error message box should be displayed: “Wrong database name!”
	Press OK button in the error message			

	Change the name of the server to the correct one Press OK button			You are successfully logged into the ZZZZZ application
	Open **Settings | User ID's** form			
	Create new user: Press "Add" button - Select "Windows authentication" - Register: DOMAIN\LOGIN_NAME, for example: ***"ARCADIA\Natasha" –*** *where ARCADIA is the name of the domain,* ***Natasha*** *is the name of the login to this domain* (Please register the domain and user name correctly to your situation) Register "User Name" = TEST USER for LOGIN Press "Save" button			
	Close the "User Ids" form with "Exit" button			
	Close ZZZZZ application			
	Run ZZZZZ application			
	Select "Windows authentication"			
	Press "OK" button			You are successfully logged into the ZZZZZ application
	End			

Тест-комплект (test case suite)

Совокупность тест-кейсов, которые проверяют:

- какую-то определенную часть проекта (например, "Оплату")
- и/или определенный спек (например, спек номер 1455 "Рассылка пользователям е-мейлов на основании истории заказов").

- Обычно располагается в одном файле.

Домашнее задание

- Прочитать: Савин - с. **33-66**, 173-204, Канер - гл. 7, Калбертсон – с.98-115, гл. 15.
- Скачать с сайта pta-ipm.narod.ru примеры ТС и изучить их
- **Разработать не менее 6 тест-кейсов** *различной сложности* для BugPad:
  - язык - английский.
  - форма таблиц – аналогичная примерам.
- Срок выполнения – до зачетной недели.
- Темы «Тестирование белого ящика» и «Автоматизация тестирования» – самостоятельно.
- Те, кому нужен зачет по УИРС, должны в дополнение к двум заданиям (**баги** и **тест-кейсы**) написать **итоговый тест** (на зачетной неделе или ранее, как скажете).