

ПЛОВДИВСКИ УНИВЕРСИТЕТ “ПАИСИЙ ХИЛЕНДАРСКИ”
ФАКУЛТЕТ ПО МАТЕМАТИКА И ИНФОРМАТИКА

Катедра “Компютърни технологии”

ОСИГУРЯВАНЕ НА КАЧЕСТВО НА СОФТУЕРА (Q.A.)

/упражнения/

гл. ас. д-р Георги Шарков

гл. ас. д-р Мая Соева

Съдържание

1. Видове софтуерна разработка
2. Тестване на софтуера според модела
3. Видове софтуерно тестване

Видове софтуерно тестване

Софтуерното тестване е процес на изследване и проучване на софтуера, с цел получаване на информация за качеството му. Процесите на софтуерното тестване са важна част от софтуерното инженерство и осигуряването на качествен софтуер.

Съществуващите засега методи за тестване на софтуер не позволяват еднозначно и напълно да се видят, проявят и установят всички дефекти при правилно функциониране на една програма, така че методите на тестване работят в рамките на формалния процес на проверка върху изследвания или разработвания софтуер.

Видове софтуерно тестване

През 1997 г. *Information Systems Examinations Board* (ISEB) на British Computer Society (BCS) започва програма на сертифициране за QA тестери.

Подобно на английския пример и други страни започват да организират такива курсове и критерии за сертифициране. Така се стига до **International Software Testing Qualifications Board (ISTQB)** и разработването на обща международна терминология.

<https://www.istqb.org/>

https://www.istqb.org/documents/ISTQB_Summary_Presentation_2017_June_v1.0.pdf

https://www.istqb.org/documents/ISTQB_2017_14.pdf

<https://www.seetb.org/index.php?page=calendar-bulgaria>

<https://www.seetb.org/index.php?page=book-exam&exam=964>

Видове софтуерно тестване

ISTQB Certified Tester схемата за сертифициране минава през три етапа. Основните положения на тестването са описани във **Foundation Level curriculum (syllabus)**.

Сертификатът **Advanced Level** certificate разкрива по-дълбоко познаване на тестването и оценката на софтуера.

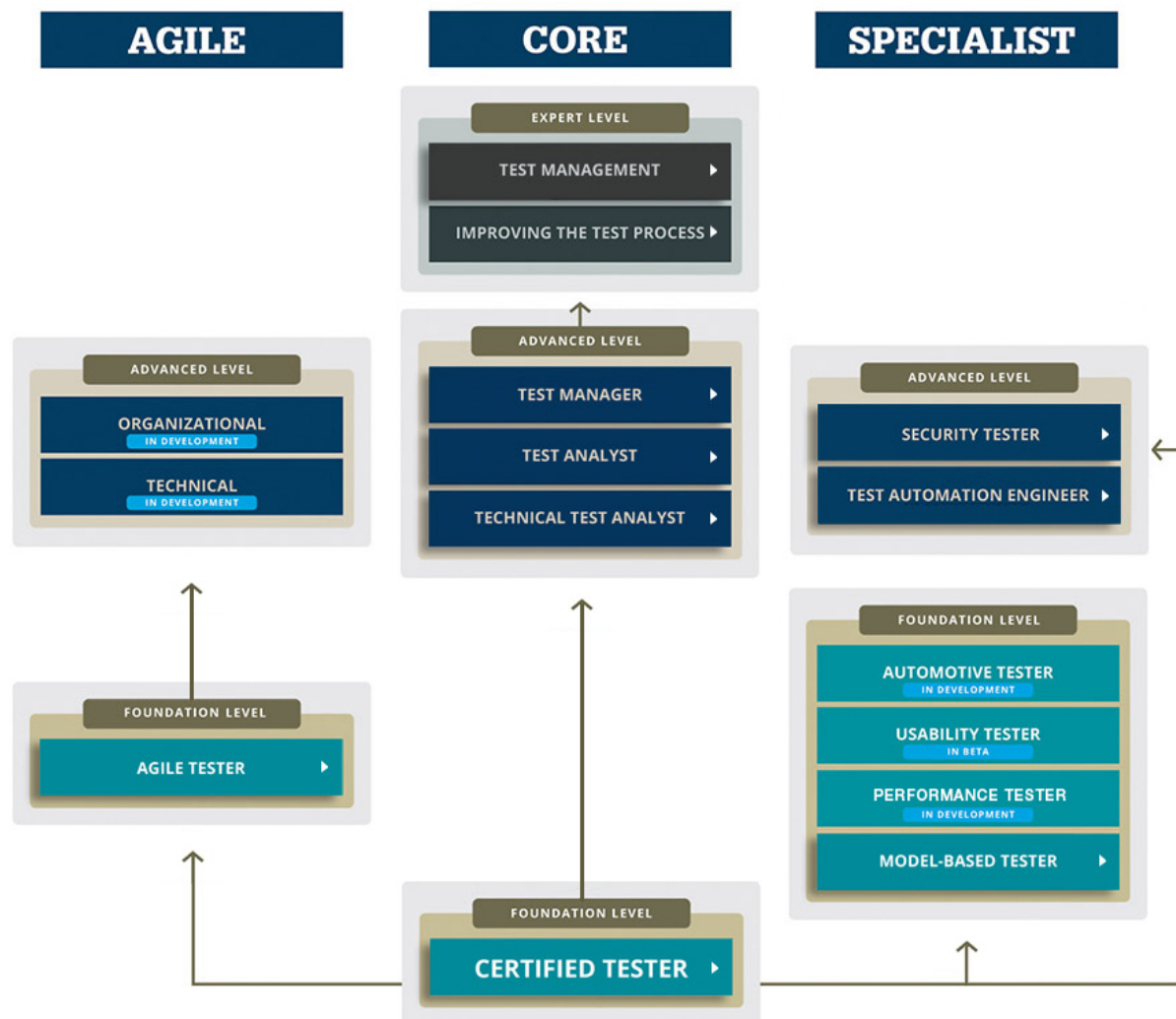
Третото ниво **Expert Level** е предназначено за опитни професионални софтуерни тестери и се състои от няколко модула.

В момента от **ISTQB** са завършени първите четири syllabi. Това са темите за **“Improving The Test Process”** и **“Test Management”**, а тези за **“Test Automation”** и **“Security Testing”** са на път.

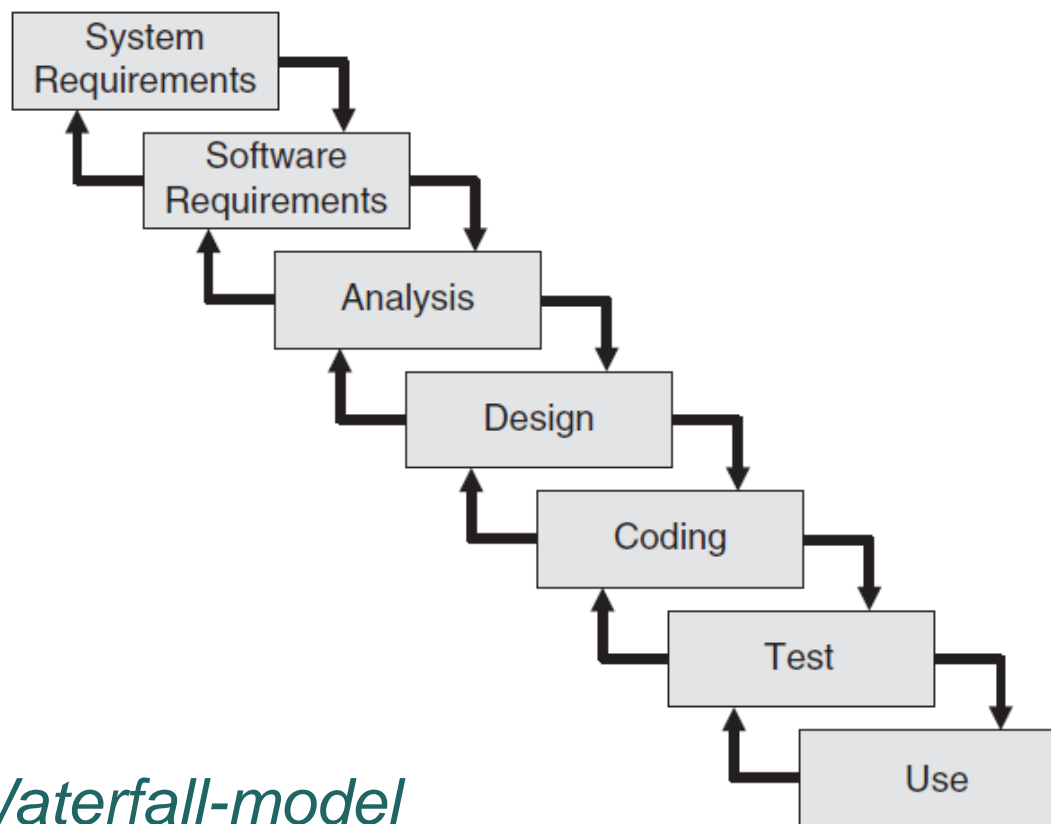
Видове софтуерно тестване

В момента са налични следните сертификати:

Foundation Level
Agile Tester Extension
Model-Based Tester Extension
Advanced Level
Security tester
Expert Level

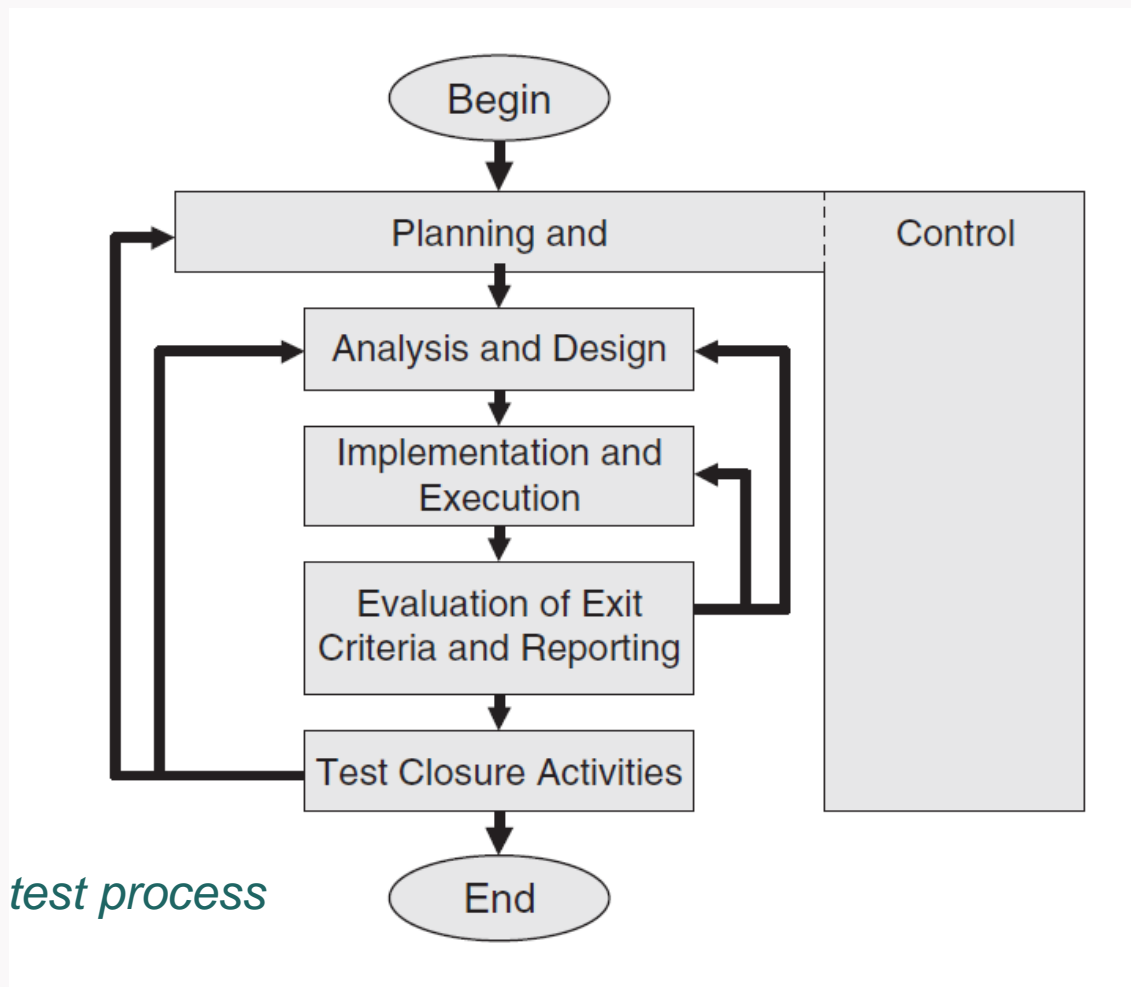


Тестване на софтуера според модела



Waterfall-model

Тестване на софтуера според модела

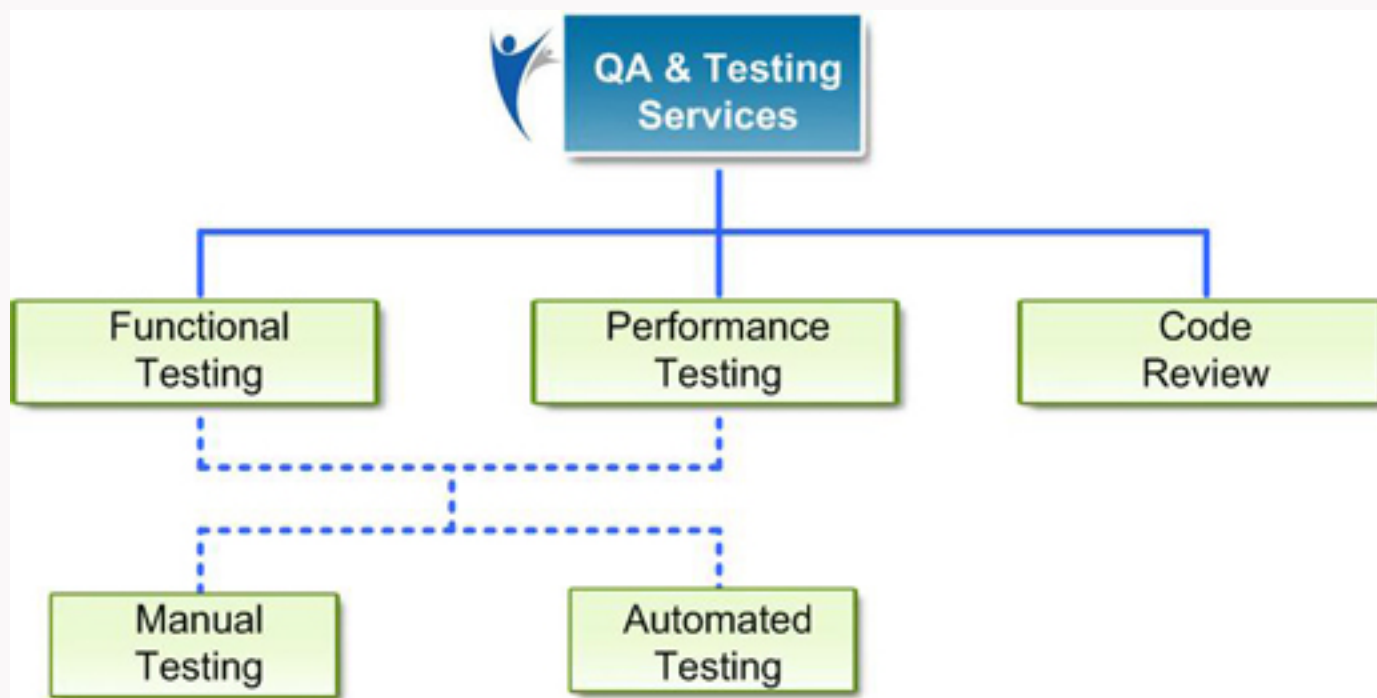


ISTQB fundamental test process

Видове софтуерно тестване



Видове софтуерно тестване



Видове софтуерно тестване

1. **Функционални тестове и нефункционални тестове**
2. **Black box testing техники**
3. **White box testing техники**
4. **Стратегия при провеждане на VBT**
5. **Стратегия при провеждане на WBT**

Нефункционални тестове

Storage testing = Resource testing

Installability testing

Documentation testing

Recovery testing

Compatibility Testing

Volume Testing

Функционални тестове

Usability Testing

Performance Testing

Load Testing

Stress Testing

Security Testing

Sanity Testing

Smoke Testing

Regression Testing

Функционални тестове

С провеждането на функционалните тестове се цели да се съпоставят реалните действия и състояние на системата с тези описани във функционалната спецификация. При наличие на разминавания с функционалната спецификация ще се направят предложения и препоръчки за тяхното отстраняване. Провеждането на този вид тестове е запланувано за всеки един етап и версия, от разработката на софтуера.

Функционални тестове

Цел:	<i>Проверка на всички функционалности на системата, според наличните отнасящи се до пълната функционалност на всеки един модул, включително навигация, входящи данни, процеси и тяхното повторение.</i>
Техника:	<p><i>Изпълнение на целия набор от написаните тестови сценарии (ТС), като се използват валидни и невалидни данни за всеки един от параметрите, за да е в състояние да се потвърди:</i></p> <ul style="list-style-type: none"> • Очаващите резултати, при правилно въведени данни, според функционалната спецификация. • Визуализацията на информационни съобщения и състоянието на системата при неправилно / некоректно зададени входящи данни • Всяка една потребителска и бизнес роля за правилно прилагане в системата
Критерии за успешно завършване на тези тестове:	<ul style="list-style-type: none"> • <i>Всички предвидени и създадени ТС са изпълнени</i> • <i>Всички намерени бъгове / дефекти са документирани</i>

Stress Testing

Целта е да се изследва системата в процес на наторване. Ще се симулират определен брой виртуални потребители, възпроизвеждащи заявки към системата. Всичко това ще се проведе, за да се отчете състоянието на системата при максимален трафик. Евентуално тестовете ще се осъществят основно върху използването на модули, за които се очаква да са най-натоварената част от системата.

Провеждането на този вид тестове се осъществява след като бъде завършена първата версия на **On demand** системата – бета версия.

Stress Testing

Цел:	<p><i>Проверка на системата според написаните ТС за функционални тестове. Тук ще се проверява системата в режим на натоварване, според следните параметри:</i></p> <ul style="list-style-type: none">• липса на достатъчно памет на сървърната машина (RAM)• максимален брой на свързаните клиенти (connections)• извършване на едно и също действие, по едно и също време от две и/или повече регистрации.
Техника:	<p><i>Използване на ТС, създадени за провеждането на Load или Performance тестове.</i></p> <p>Тестове трябва да бъдат извършени от една и съща машина, като паметта на сървъра трябва да бъде намалена (или лимитирана).</p>
Критерии за успешно завършване на тези тестове:	<p><i>Всички сценарии са изпълнени, като се определят минимум ресурси за правилното функциониране на система, без да се отчитат грешки в нея.</i></p>

Regression Testing

С провеждането на този вид тестове се потвърждава дали вече отстранените бъгове и дефекти, не се репродуцират в следващите версии.

Този вид тестове се провежда за всяка една версия на продукта, с цел максимално ефективно управление на качеството.

Regression Testing

Цел:	<i>Проверка на документираните бъгове / дефекти за тяхното отстраняване. При репродуциране отново се документират.</i>
Техника:	<i>Използване на ТС, създадени за провеждането на функционални тестове.</i>
Критерии за успешно завършване на тези тестове:	<i>Всички предвидени и създадени ТС са изпълнени Всички документиран бъгове / дефекти са ретествани. При наличие на репродуциране на грешки, те отново се документират.</i>

User-interface testing

Провеждането на тези тестове се извършва, за да се определи до каква степен потребителят е максимално улеснен при ползването на услугите от даден софтуер.

Провеждат се тестове за ползваемост и за да се установи до колко е лесна ориентацията в софтуера, и дали всички менюта и опции са леснодостъпни за потребителя, и дали следват логиката.

User-interface testing

Цел:	Потвърждаване на следните параметри: <ul style="list-style-type: none"> • Навигацията в системата отговаря на бизнес логиката и логиката на процесите. • Всички обекти отговарят на стандартите, включително менюта, размери, полета, позиции.
Техника:	<i>Примерно провеждане на тестове за всеки един екран , за да се провери за раз мествания по екрана, раз мествания на обекти и полета, използвайки браузъри с различни ядра – Internet Explorer, Mozilla Firefox, Opera.</i>
Критерии за успешно завършване на тези тестове:	<i>Всеки един екран отговаря на стандартите и изискванията. Липса на различия по дизайна, използвайки различни браузъри</i>

Black & White testing

Black Box Testing са тестове, включително функционални и нефункционални, които не се отнасят до вътрешната структура на системата.

White Box Testing са базирани на анализ на вътрешната структура на компонента или системата.

Black box testing

Концентрира се върху бизнес функциите и логиката и се прилага се по време на целия жизнен цикъл.

Най-интензивно след първичното кодиране.

Няма отношение по изискванията за използване на технология или нуждата от определен код.

Black box testing

Тоест Black box testing е техника за функционално тестване, в която са получени данни с определено функционално изискване. Акцентираща се върху изпълнението на функциите и разглеждането на техният принос и изходните данни. При **Black box testing** не се разглежда сорс кода, а само получените резултати от него. Това са потребителският интерфейс и функционалностите, които са достъпни до потребителите на софтуерния продукт.

Black box testing

Black box testing включва тестове базирани на спецификацията.

Нуждаем се от VBT техники тъй като пълен формат на тестовете е невъзможен.

Създаване на поредица от тестове.

Концентрация върху риска.

Black box testing

Всяка VBT техника включва:

Метод – как да го направим?

Тест дизайн – как да създадем тест сценарии

Техники за измерване

Black box testing

Equivalence Partitioning:

Входящи данни

Изпълнение: валидно или невалидно

Очаквани резултати

Black box testing

Пример:

Информация за конфигурация:

Модел

Година на производство

Процесор

Диск

Определяме: *Валидни и невалидни стойности и тест сценарии с очакван резултат*

Black box testing

Boundary Value Analysis:

- Използва модел, за да установи границата на всеки един компонент или система.
- Използват се входящи и изходящи, валидни и невалидни данни.
- Определяне на максималната допустима граница при системата, която тества.
- Най-често се комбинира с Equivalence partitioning тестове

Входящи параметри към компонента

Изследване на граници – под / в / над

Очаквани резултати

Black box testing

BBT се концентрира върху тестване и проверка на функционалността на системата и неговите техники ни позволяват да разширим обхвата на тестовете.

BBT е уместно да се използват през целия процес на тестване.

White Box testing

White Box testing още е познато като “**Glass Box testing**” или “**Structural testing**”.

Фокусира се върху самия код и отговаря на специфични изисквания.

Интересува се от устройството/структурата на тестваната система.

Прилага се в ранна фаза на тестването.

White Box testing

White box testing е вид функционално тестване и се изпълнява веднага след като е написан кода, не се нуждае от завършена система, а само от познания по използвания програмен език.

White box testing е техника базирана на вътрешната структура на базите данни и кода. Тестера преглежда кода и открива бъговете, но за да прочете кода и анализира грешките в софтуера, му трябва познания по съответния програмен език.

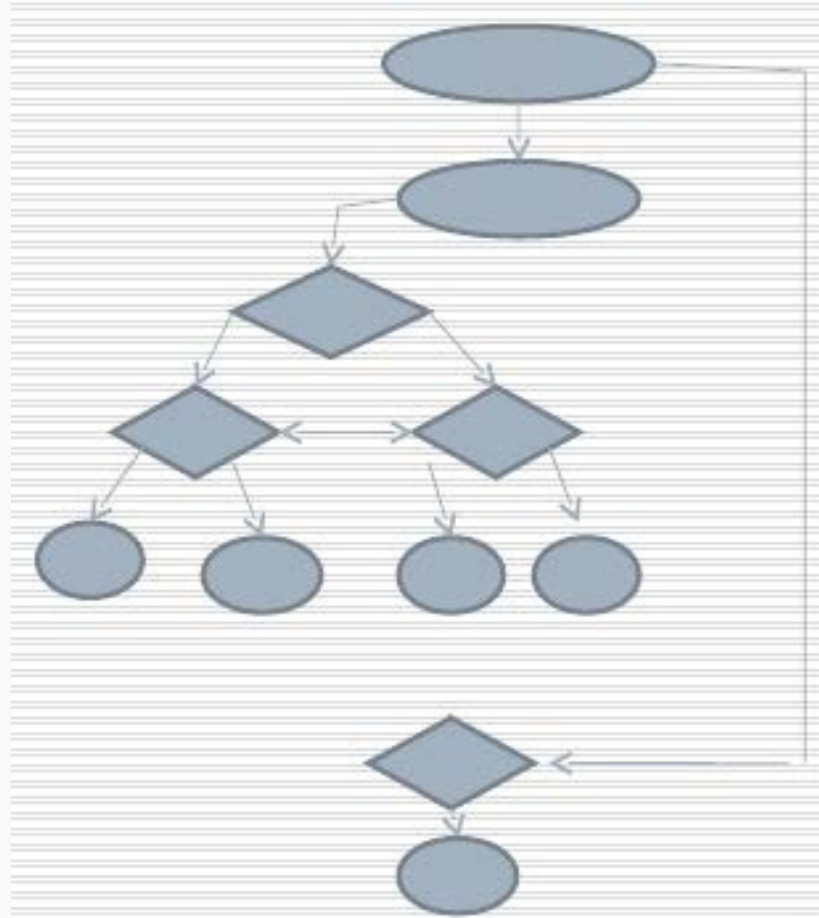
White Box testing

Тестове базирани на анализ на вътрешната структура на компонента или системата.

Защо се нуждаем от White Box тестове:

Изследваме структурата на кода и определяме мерни единици за тестове и колко компонента трябва да бъдат тествани. *Например:* 1000 реда код, 100 000 000 възможни комбинации на 4 секунди да се извършват 100 теста.

White Box testing



White Box testing

За да проведем успешни WBT трябва да знаем:

Използвана технология

Използвана база данни

Операционна система

Познания по дадения програмен език

White Box testing

Statement Testing

WBT техника, при която се проверява изпълнението на всеки един етап от изграждането на системата.

Тест сценарии за проверка на всички възможни комбинации от дадена система поне веднъж.

Стремим се да изпълним сценария с коректни данни.

White Box testing

WTB може да се изпълнява веднага след като е написан кода

Не се нуждае от завършена система

Изисква се познания по използвания език за програмиране

Извод

Един добър план съдържа комбинация от **Black box testing** и **White box testing**.

Стратегия при провеждането на White Box Testing

Разделят се на два вида:

Тестове с участие на потребител

Тестове без участие на потребител

Стратегия при провеждането на White Box Testing

Тестове без участие на потребител

Функционални тестове

Stress testing

Load testing

Ad-hoc testing

Exploratory testing

Usability testing

Smoke testing

Recovery testing

Volume testing

Стратегия при провеждането на White Box Testing

Тестове с участие на потребител

User acceptance testing

Alpha testing

Beta testing

Стратегия при провеждането на White Box Testing

Стратегия при WBT

Unit testing

Static and dynamic analysis

Обхват на тестването

Обхват на кода

Security testing

Тестове при мутации на кода

План за тестване според IEEE Standard 829-2008 – основни положения:

1. Test Plan Identifier
2. Introduction - Test Objects or Items, Features to Be Tested, Features Not to Be Tested, Test Approach or Strategy
3. Acceptance Criteria (Test Item Pass/Fail Criteria)
4. Suspension Criteria and Resumption Requirements
 - 4.1. Test Documentation and Deliverables
 - 4.3. Testing Tasks
 - 4.4. Test Infrastructure and Environmental Needs
5. Responsibilities and Authority
6. Staffing and Training Needs
7. Schedule
8. Risks and Contingencies
9. Glossary (not in IEEE829-1998, but lower case!)

План за тестване според IEEE Standard 829-2008 – основни положения:

1. Introduction

1.1. Document identifier

1.2. Scope

1.3. References

1.4. System overview and key features

1.5. Test overview

1.5.1 Organization

1.5.2 Master test schedule

1.5.3 Integrity level schema

1.5.4 Resources summary

1.5.5 Responsibilities

1.5.6 Tools, techniques, methods, and metrics

Планът за тестване според IEEE Standard 829-2008 включва:

2. Details of the Master Test Plan

2.1. Test processes including definition of test levels

2.1.1 Process: Management

2.1.2 Process: Acquisition

2.1.3 Process: Supply

2.1.4 Process: Development

2.1.4.1 Activity: Concept

2.1.4.2 Activity: Requirements

2.1.4.3 Activity: Design

2.1.4.4 Activity: Implementation

2.1.4.5 Activity: Test

2.1.4.6 Activity: Installation/checkout

2.1.5 Process: Operation

2.1.6 Process: Maintenance

2.1.6.1 Activity: Maintenance test

2.2. Test documentation requirements

2.3. Test administration requirements

2.4. Test reporting requirements

The Level Test Plan (само за едно ниво)

1. Introduction

- 1.1. Document identifier
- 1.2. Scope
- 1.3. References
- 1.4. Level in the overall sequence
- 1.5. Test classes and overall test conditions

2. Details for this level of test plan

- 2.1 Test items and their identifiers
- 2.2 Test Traceability Matrix
- 2.3 Features to be tested
- 2.4 Features not to be tested
- 2.5 Approach
- 2.6 Item pass/fail criteria
- 2.7 Suspension criteria and resumption requirements
- 2.8 Test deliverables

3. Test management

- 3.1 Planned activities and tasks; test progression
- 3.2 Environment/infrastructure
- 3.3 Responsibilities and authority
- 3.4 Interfaces among the parties involved
- 3.5 Resources and their allocation
- 3.6 Training
- 3.7 Schedules, estimates, and costs
- 3.8 Risk(s) and contingency(s)

4. General

- 4.1 Quality assurance procedures
- 4.2 Metrics
- 4.3 Test coverage
- 4.4 Glossary
- 4.5 Document change procedures and history

Планът за тестване според IEEE Standard 829-2008 включва:

3. General

3.1. Glossary

3.2. Document change procedures and history

Литература

<http://www.softwaretestinghelp.com/types-of-software-testing/>

<http://www.softwaretestinghelp.com/category/basics-of-software-testing/>

https://bg.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BD%D0%B0_%D1%81%D0%BE%D1%84%D1%82%D1%83%D0%B5%D1%80

<http://www.testingexcellence.com/types-of-software-testing-complete-list/>

<https://bulgariaqa.wordpress.com/2012/08/03/black-box-testing-white-box-testing/#more-48>

<https://www.slideshare.net/kalin4y/ss-4672183/2>

<http://www.istqb.org/>

<http://kvasilev.com/sqa/test-plan-vidove-testove/>

https://www.techstreet.com/standards/ieee-829-2008?product_id=1599445

<http://ieeexplore.ieee.org/document/4578383/?reload=true>

Andreas Spillner, Tilo Linz, Hans Schaefer, „**Software Testing Foundations: A Study Guide for the Certified Tester Exam**”

**БЛАГОДАРЯ ЗА
ВАШЕТО ВНИМАНИЕ!**