



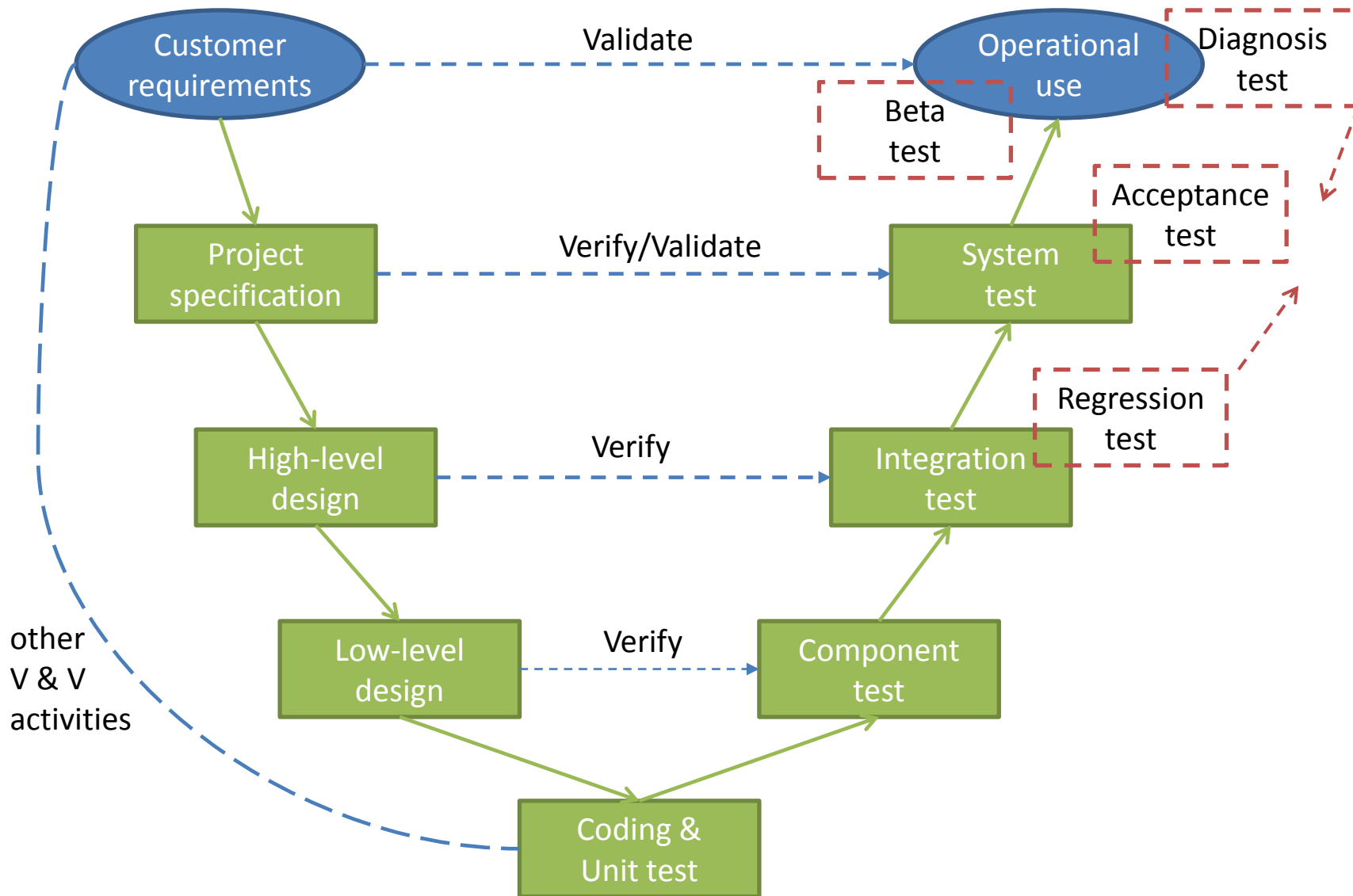
ТЕХНИКИ ЗА ТЕСТВАНЕ: АДАПТАЦИЯ, СПЕЦИФИЧНОСТ И ИНТЕГРАЦИЯ

доц. д-р Десислава Петрова-Антонова

Съдържание

- ❖ Тестови фази и свързани с тях техники за тестване
- ❖ Специфични задачи и техники за тестване
- ❖ Интеграция на техниките за тестване за постигане на специфични цели

Тестови подфази, съответстващи на V-модела

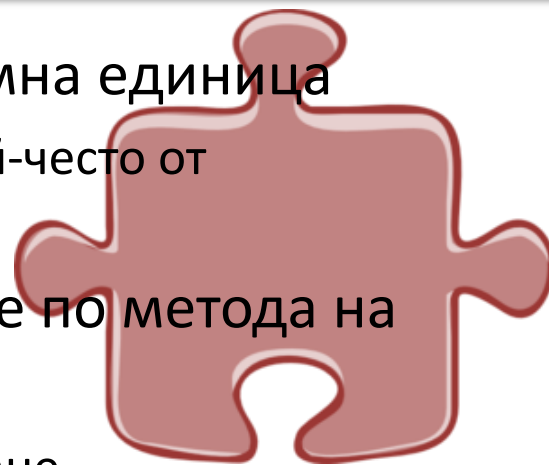


Тестови подфази

- ❖ **Диагностично тестване**
 - Пресъздаване и диагностициране на проблеми, възникнали при клиента
- ❖ **Бета тестване**
 - Предхожда оперативното използване или се извършва в неговото начало
- ❖ **Тестване за приемане на системата**
 - Извършва се в края на системното тестване
 - Определя дали продуктът може да бъде доставен на крайния потребител
- ❖ **Системно, интеграционно и компонентно тестване**
 - Тестови фази, съответстващи на фазите на разработка в модела на “водопада”
- ❖ **Тестване на ниво програмна единица**
 - Подфаза в общия тестов процес и част от кодирането
- ❖ **Регресионно тестване**
 - Фокусира се най-вече върху интеграционното тестване при интегриране на старите и новите системни компоненти

Тестване на ниво програмна единица

- ❖ Същност на тестването на ниво програмна единица
 - Тестване на малки програмни единици, най-често от програмистите, които ги реализират
- ❖ Използва предимно техники за тестване по метода на бялата кутия
 - “Ad-hoc” тестване или неформално дебъгване
 - ✓ Използване на специфична тестова среда (testbed) при наличие на зависимост от друга програмна единица
 - Разделяне на входния домейн
 - Традиционни структурно, базирани техники като тестване на управляващия поток и тестване на данновия поток
- ❖ Възможност за прилагане на техники за тестване по метода на черната кутия
 - Използва се рядко поради наличието на информация за начина, по който програмната единица е реализирана



Компонентно тестване

❖ Същност

- Тестване на софтуерни компоненти от малка група разработчици

❖ Компонентно тестване при структурното програмиране (C, PASCAL, FORTRAN)

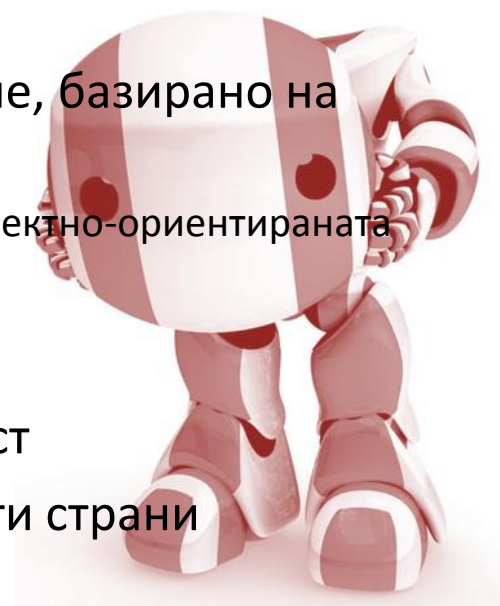
- Наподобява тестването на ниво програмна единица

❖ Компонентно тестване в обектно-ориентираните системи

- Тестване по метода на бялата кутия
- Тестване по метода на черната кутия и тестване, базирано на употреба
 - ✓ Приложението им се обуславя от особеностите на обектно-ориентираната парадигма

❖ Тестване на COTS компоненти

- Повишено изискване за качество и надеждност
- Независима верификация и валидация от трети страни



Интеграционно тестване

- ❖ Същност на интеграционното тестване
 - Тестване на интерфейсите и взаимодействието на компонентите при интегрирането им
- ❖ Използва предимно техники за тестване по метода на бялата кутия
 - Критерий за изход от тестовия процес е постигането на определена степен на покритие
 - Изпълнява се от професионални тестери
- ❖ Управлението на изпълнението се моделира с машина на крайните състояния
- ❖ Възможност за прилагане на техники за тестване по метода на черната кутия
 - Разглеждане на компонентите като цяло и прилагане на техники за системно тестване
 - При някои системи интеграционното тестване е част от системното тестване



Системно тестване

❖ Същност на системното тестване

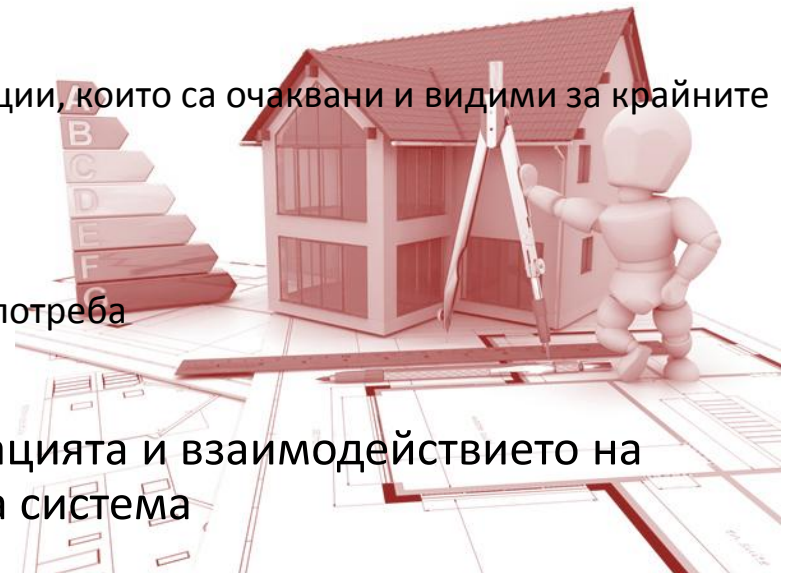
- Тестване на външните системни операции като цяло обикновено от гледа точка на клиента
- Необходимост от знания за продуктите функции, приложния домейн, пазарния сегмент и потребителските очаквания

❖ Техники за тестване

- Функционални контролни списъци от високо ниво
 - ✓ Тестване, базирано на покритие
 - ✓ Статистическо тестване, базирано на употреба (оперативен профил на Муса)
- Машини на крайните състояния
 - ✓ Състоянията представят главните функции, които са очаквани и видими за крайните потребители
 - ✓ Тестване по метода на черната кутия
- Модел на Марков
 - ✓ Статистическо тестване, базирано на употреба

❖ Супер-системно тестване

- Тестване на интерфейсите, интеграцията и взаимодействието на подсистеми, изграждащи цялостна система



Тестване за приемане на системата

❖ Същност на тестването за приемане на системата

- Тества се готовността на продукта за доставяне до крайните потребители
 - ✓ Какви са очакванията за продуктова надежност на клиента?
 - ✓ Какво е нивото на поддръжка след доставка на продукта?

❖ Тестване за приемане на системата като част от системното тестване

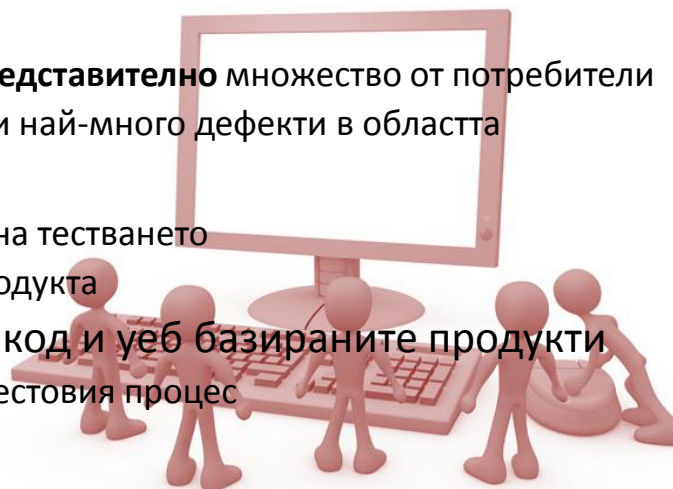
- Системно тестване: Коригиране на дефектите преди доставяне на продукта
- Тестване за приемане на системата: Предполага се, че дефектите няма да бъдат коригирани поради предстоящото доставяне на продукта
 - ✓ Отстраняване на дефекти при поддръжката след доставка на продукта
 - ✓ Забавяне на доставката на продукта при наличие на сериозни проблеми

❖ Техники за тестване

- Статистическо тестване, базирано на употреба
 - ✓ Възможност за оценка на надеждността
- Приложение на оперативните профили на Муса и Марков

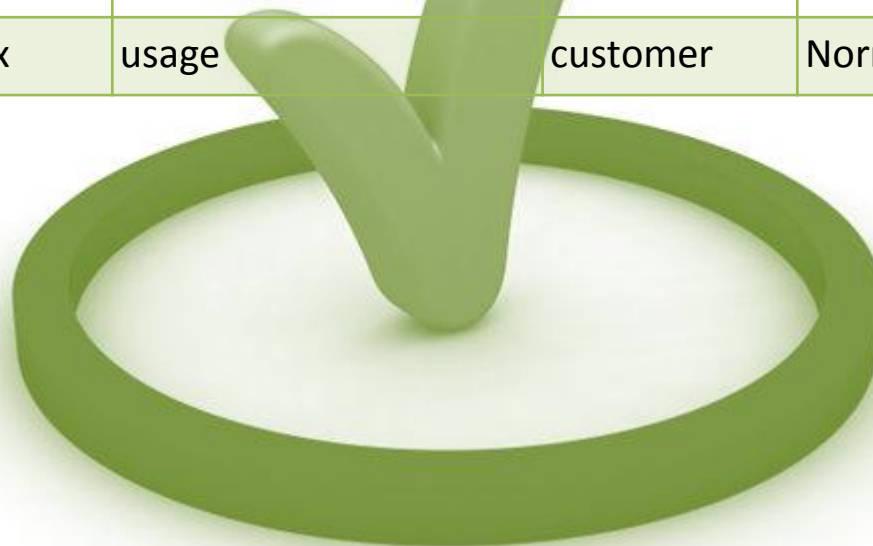
Бета тестване

- ❖ Ограничения при статистическото тестване, базирано на употреба
 - Трудности при събиране на информация за случаите на използване поради конфиденциалността на бизнес данните
 - Трудности при създаването на оперативни профили поради разнообразието на клиентите
- ❖ Същност на бета тестването
 - Тестване при контролирана и ограничена доставка на продукта до крайните потребители
 - ✓ Прилага се при приложения с широко разпространение, при които не е приложимо статистическото тестване, базирано на употреба
- ❖ Цена на бета тестването
 - Директни разходи и ограничения
 - ✓ Идентифициране на **достатъчно голямо и представително** множество от потребители
 - ✓ Използване на потребители, които са открили най-много дефекти в областта
 - Индиректни разходи
 - ✓ Време и усилия за подготовка и изпълнение на тестването
 - ✓ Време за запознаване на потребителите с продукта
 - Високо качество на продуктите с отворен код и уеб базираните продукти
 - ✓ Включване на широк кръг от потребители в тестовия процес



Сравнение на тестовите подфази

Подфаза	Гледна точка	Критерий за спиране	Изпълнител	Техники
Unit	White box	coverage	programmer	db, s-list, BT, CFT, DFT
Component-I	White box	coverage	programmer	s-list, BT, CFT, DFT
Component-II	Black box	both	tester/3p	BT, CFT, DFT
Integration	White box	coverage	tester	FSM, CFT, DFT
System	Black box	both	tester	f-list, FSM, Musa, Markov
Acceptance	Black box	usage	tester/3p	Musa, Markov
Beta	Black box	usage	customer	Normal usage



Тестване за диагностика на дефекти

❖ Типично ниво на приложение

- Диагностициране на проблеми, получени при работа на крайните потребители със софтуера или при бета тестване
 - ✓ Пресъздаване на проблема, наблюдаване на поведението на системата, събиране и анализ на информация

❖ Реализация

- Изпълнение на последователност от свързани тестови серии, базирани на сходни сценарии

❖ Приложение при ниските подфази на тестване

- Тестване в организацията: тестовите сценарии и резултатите от тестовите серии се предоставят от тестерите на разработчиците
- Бета тестване: Информацията за случаите на употреба и възникналите проблеми е ограничена

❖ Приложение при инспекция и други дейности, свързани с осигуряването на качеството

❖ Обобщение на приложимостта

- Колкото повече информация се получава от тестовите дейности, толкова по-малка е необходимостта от диагностично тестване

Тестване, базирано на дефекти

❖ Същност

- Тестване, при което се използват открити или потенциални дефекти
 - ✓ “ad hoc” тестване, базирано на субективни предположения за местоположението на дефектите

❖ Стратегии за тестване

- Тестване, базирано на риска от поява на дефекти
 - ✓ Идентифициране на области с висок риск или области, които са с най-голям потенциал за възникване на дефекти
- Инжектиране на дефекти и тестване
 - ✓ Тестване за прихващане както на инжектираните, така и на реално допуснатите дефекти в системата
- Тестване на мутации
 - ✓ Извършване на малки промени в програмите (създаване на мутанти) и изпълнение на тестови пакети с цел откриването им
 - Идентифициране на синтактични вариации
 - Оценка на съществуващите тестови пакети

Регресионно тестване

❖ Поддръжка на софтуера

- Анализиране и отстраняване на дефекти, открити от клиентите
 - ✓ Приложение на тестването за диагностика на дефектите

❖ Обновяване на софтуера

- Адаптиране на софтуера към различни оперативни среди
- Подобряване на софтуерния продукт

❖ Регресионно тестване

- Проверява дали поддържаните софтуерни функции са засегнати от новите версии на продукта
- Фокусира се основно върху интеграционното тестване при свързването на стари и нови компоненти

❖ Компоненти на регресионното тестване

- Анализ на различията между предишната и текущата версии на продукта въз основа на формални и неформални модели
 - ✓ Избор на съществуващи и генериране на нови тестови сценарии
- Обект на новите тестови сценарии
 - ✓ Тестване на нови или обновени части от софтуера
 - ✓ Тестване на интеграцията между старите и новите части на софтуера

Тестови цели

❖ Базови тестови цели

- ↑ Високо ниво на надеждност
- ↓ Ниско ниво на дефекти

❖ Тестване за постигане на допълнителни цели

- Тестване на производителността (performance testing)
 - ✓ Тестване в реална оперативна среда при специфични изисквания за производителност като време за отговор, пропускателна способност и др.
- Стрес тестване (stress testing)
 - ✓ Натоварване с голям обем от данни или дейности за кратък период от време
 - ✓ Типично е за уеб базираните приложения, при които се тества натоварване при голям брой конкурентни потребители
- Тестване на използваемостта (usability testing)
 - ✓ Съответствие на потребителския интерфейс и индивидуалните характеристики на потребителите (интелект, образование и среда)
 - ✓ Разбираемост на съобщенията за грешка
 - ✓ Унифицираност на потребителския интерфейс от гледна точка на синтаксис, семантика, форматиране, стил и аббревиатури
 - ✓ Удобство на използване

Динамични анализи и техники, свързани с тях

❖ Симулация

- Верифицира проектирането от високо ниво или характеристики на системната архитектура през ранните фази на софтуерния процес
- Използват се софтуерни и хардуерни симулатори
- Пример: използване на компютърна симулация за тестване на софтуерните компоненти в Boeing 777

❖ Прототипизиране

- Използване на софтуерни прототипи или опростени системи за тестване на проектирането от високо ниво или характеристики на системната архитектура

❖ Анализ с дърво на събитията

- Възстановяване на инциденти при системи, които са критични по отношение на безопасността

Интеграция на тестовите техники и дейности

❖ Необходимост от интеграция

- Различна ефективност на тестовите техники по отношение на специфични проблеми
- Наличие на различни аспекти и проблеми в софтуерните системи

❖ Допълнителни ползи

- При интегрираното използване на техниките могат да се открият проблеми, които с отделните техники са неоткриваеми

❖ Повишена гъвкавост

- Възможност за екстраполиране на резултатите от интеграционните стратегии върху различни ситуации, типове продукти и пазарни сегменти

❖ Намалена цена

- Спестяване на някои от дейностите, свързани с изпълнението на отделни техники за тестване
 - ✓ Споделяне на източници на информация
 - ✓ Преизползване на модели
 - ✓ Използване на тестови резултати като “оракули”

Тестване на уеб приложения

CASE STUDY

Техники за функционално уеб тестване

❖ Проверка на HTML синтаксиса



- Валидиране на HTML файлове по отношение на определена езикова спецификация и стандарти

❖ Проверка на хипервръзките



- Валидиране на уеб приложение по отношение на целостта на хипервръзките

❖ Тестване на форми

- Валидиране на атрибутите на елементите във формите (type, name и др.)
 - ✓ Your Name: `<input type="text" name="name" required>`

❖ Тестване на Java компоненти

- Прилагане на традиционните техники за функционално тестване върху Java аплети, сървърни скриптове и др.

Техники за нефункционално уеб тестване

❖ Тестване на натоварване

- Подмножество на стрес тестването или тестването на производителността
- Тестване на времето за отговор при голям брой потребители

❖ Тестване на използваемост

- Тестване на удобството за използване, определено от уеб дизайна, общия изглед на страниците и навигацията

❖ Тестване за съвместимост с браузъри

- Тестване на функционалността и визуализацията на компонентите на уеб страниците при различни браузъри

