С О Ф И Й С К И У Н И В Е Р С И Т Е Т
"С В. К Л И М Е Н Т О Х Р И Д С К И"

Факултет по Математика и Информатика
Катедра „Информационни Технологии”

Д И П Л О М Н А Р А Б О Т А

тема:
Тестване базирано на модели

Дипломант: Мирослава Владкова Иванова, фак.номер:M21228

Специалност: Информатика

Специализация: Разпределени системи и мобилни технологии

Научен ръководител: доц. Силвия Илиева

Консултант: ст.ас. Елиза Стефанова

София, 2007 г.

Съдържание

Увод

Тестването е най-известният метод, чрез който се осигурява качеството на софтуерните продукти. Тестването на една софтуерна система означава откриване на несъответствия между реалното и очакваното поведение. Един начин за подобряване на тестването е използването на модели, които описват очакваното поведение според изискванията на системата. Процеса на тестване чрез модели е познат като "тестване базирано на модели". Той обикновено се отнася до генериране на тестови случаи и оценка на резултатите въз основа на модел на поведение на системата. Понастоящем съществуват различни методи за дефиниране на модели на софтуера - например машина с краен брой състояния (finite state machine), граматики (Grammars) и други.

Основни цели на разработката

Целта на дипломната работа е да се проучат и приложат методите за тестване, базирано на модели и анализ върху конкретно софтуерно приложение.

Задачи, произтичащи от целта:

- Да се направи обзор/сравнителен анализ на класическите подходи за тестване и тестване, на базата на модели.
- Да се приложи тестване на базата на модели в реално приложение.
- Да се изследва влиянието на параметрите – например график, цена и качество.
- Да се анализират резултатите и да се предложат препоръки/подход за прилагане на тестване на базата на модели.

Структура на дипломната работа

Дипломната работа включва: Увод, Изложение, Заключение, Речник на използваните съкращения и термини, Списък с използвана литература и Приложение.

Увод – Кратко въведение в темата и описание на структурата на дипломната работа.

В Глава 1 се разглежда състоянието в областта – представя се тестването базирано на модели; извършва се сравнение между класическите подходи за тестване и тестването базирано на модели.

Глава 2 е посветена изцяло на тестването базирано на модели. Представят се видовете модели, които могат да се използват в софтуерното тестване. Анализират се основните характеристики на тестването базирано на модели и се разглеждат стъпките при прилагането му.

Глава 3 представя приложение на тестването базирано на модели в реално време. Създаден е модел на поведение на уеб базирано приложение. Показано е определянето на много тестови случаи, чрез този модел.

Глава 4 съдържа изследване на влиянието на следните параметри – график, цена и качество

Глава 5 анализира резултатите и предлага подход за прилагане на тестването базирано на модели.

Използвана литература

Речник на използваните термини и съкращения

Приложение: Програмни кодове от реализираното софтуерно приложение

Глава 1. Обзор

Тестването на една софтуерна система означава откриване на различията между реалното и очакваното поведение на системата. При по-сложните софтуерни системи и кратките им срокове за изпълнение е почти невъзможно да се направи пълно гарантиране на качеството. В тази глава са предствени класическите техники за тестване и е направен сравнителен анализ между тях.

1.1Класически техники за тестване

В следващия параграф се представят част от класическите техники за тестване. Това са ръчно тестване, автоматизирани тестове и тестване със случайни действия.

1.1.1. Ръчно тестване

Най – често в практиката се използва ръчното тестване. То представлява ръчна намеса и взаимодействие на човека със системата, която тества. Тестват се всички функционалности, които се съдържат в приложението. Сравняват се текущите състояние със очакваните. При ръчното тестване винаги има вероятност да се изпусне проверката на някой тестови случай.

1.1.2. Автоматизирани тестове

Автоматизираното тестване включва използването на стратегии и инструменти, които извършват дейностите по тестване на даден софтуерен продукт, и при които намесата на човек е свързана основно с изследване на получените резултати от тези дейности.

Автоматизираното тестване е основано на тестови скриптове и инструменти за тестване. Те са програми, които предоставят механизми за поддържане на различни подходи за автоматизирани тестове. Биват реализирани на следните програмни езици - VB, C, C + +, JAVA, PERL.

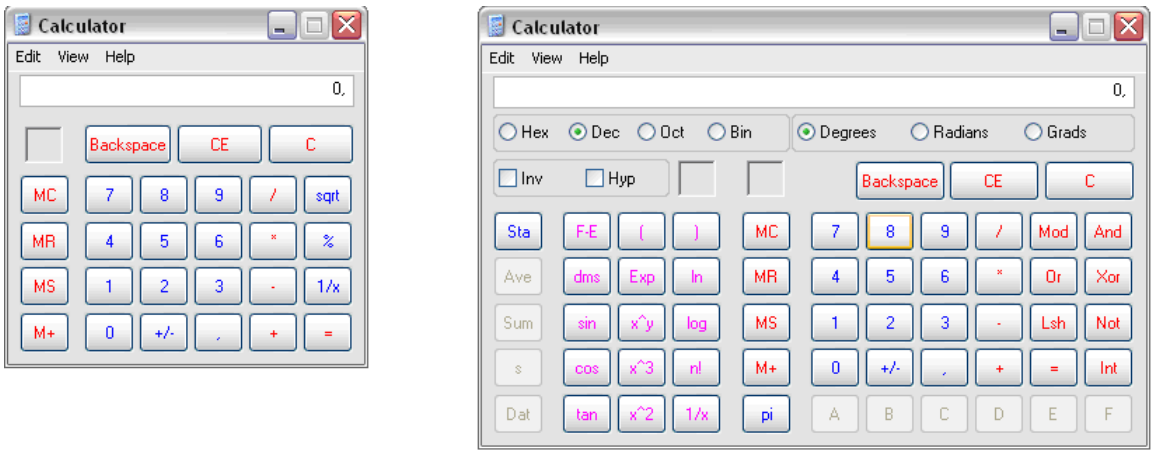
При тестването се изпълняват тестови скриптове:

- Инициира се системата, която ще се тества (System Under Test - SUT)
- Обхождане на множеството от тестове, като за всеки случай се изпълняват следните дейности
    - Инициализация на SUT
    - Инициализация на изходните данни, различни от очакваните (ако е възможно)
    - Въвеждане на данните
    - Изпълнение на SUT
    - Извличане на изходните данни и съхраняването им до момента, когато доклада (report) за теста може да бъде създаден

Пример:

За целта на примера ще използваме калкулатора в Microsoft Windows [3] и ще разгледаме как тази техника може да бъде приложена върху него.

Приложението има меню, което дава възможност да се избере модел(Фигура 1): стандартен (standard) или научен (scientific). При вторият модел има по-голяма възможност за избор на функционалност: Hex, Decimal, Octal, Binary, Qword, Dword, Word, Byte.



Фигура 1 Калкулатор в Microsoft Windows.

Статичен автоматичен тест при калкулатора може да бъде във формата на скрипт (script), който описва една и съща последователност от действия в един същи ред. Пример:

- Стартиране на Калкулатор
- Избиране на "Standard" от View
- Избиране на "Scientific" от View
- Избор на възможностите на калкулатора
- Затваряне на приложението

Целта на този пример беше да покажем недостатъкът на тази техника. Всеки скрипт е изцяло фиксиран и трябва да бъде поддържан индивидуално. Няма възможност да се изпълнят различни варианти на последователност от действия.

1.1.3. Изпълнение на случайни действия

Случайните тестове се прилагат на приложенията, като действията които се извършват са случайно избрани. При тях няма значение какво се случва след дадено действие и какво трябва да се изпълни. Да вземем за пример калкулатора, които разгледахме.

При стартиран калкулатор модел Scientific може да натискаме произволни клавиши на клавиатурата, докато в един момент с натискането на F6 избираме Decimal.

1.1.4 Сравнителен анализ на класическите техники за тестване

Тук ще направим един сравнителен анализ на класическите техники за тестване – ръчно тестване, автоматизирани тестове и изпълнение на случайни действия.

- Системата тествана от нас, променя малко или много своята функционалност. Това означава, че атоматизираните тестове, които са били създадени трябва да бъдат променени съобразно новата функционалност за да може да поддържат новото поведение. За съжаление това е много скъпо струващ процес.
- Автоматизирани тестове могат да откриват само специфична част от грешките и тези тестове стават безсмислени след като определените грешки са открити и поправени. Този феномен е известен като "pesticide pardox".
- Изпълнението на случайни действия за тестове води до трудно контролиране на последователността от действия, дори невъзможно, което пък говори за неизпълнение на тестовете покриващи цялата функционалност.
- Ръчното тестване не е подходящо за сложните системи.

Класическите техники за тестване имат и свойте недостатъци. Скриптовете(scripts) написани при автоматизираните тестове не могат да се използват след промяна на функционалността на системата. Ръчното тестване е ефективно за малки системи, а изпълняването на случайни действия може да доведе до

неизпълнение на тестове.
Тестването базирано на модели решава тези проблеми като предоставя описание на поведението или модел на системата, която се тества. Отделен компонент използва този модел да генерира тестовите случаи. След това тези тестови случаи се изпълняват за системата, която се тества.

1.2 Видове тестване

Следващият параграф описва видовете тестване, основните им характеристики и дава изчерпателна дефиниция на тестването базирано на модели.

1.2.1Тестване в процеса на разработка.

При изработването на една софтуерна система, процеса на тестване се състои от няколко различни фази. Тъй като по време на всяка фаза се сравняват изискванията на системата с реалната реализация, във всяка една от тях е приложимо тестването базирано на модели.
Във всяка фаза е необходимо да се определят тестовите случаи. При тест процесите се използват различни видове и начини на тестване. Ще разгледаме някои от използваните в практиката определения за тест процеси:

- **Unit Testing**

Тестване на най-малката програмна структурна единица – метод, клас, компонент или модул; обикновенно това се прави от програмистите, като се дебъгва програмен код или се записват и кодират автоматизирани тестове за многократно изпълнение; По

време на тази фаза на тестване тестването базирано на модели може да се използва, за да се тества даден отделен модул. Чрез моделиране на отделните входните данни за всеки отделен компонент могат бързо да се генерират множество от различни тестове, които базирани на изискванията да верифицират функционалността им.

- **System Testing**

Black-box тип тестване, което се базира на пълния обем специфицирани изисквания на потребителите; покрива пълното тестване на всички комбинирани и интегрирани части на системата. Ако по време на този етап се приложи тестването, базирано на модели, може да се гарантира, че изискванията са изпълнени.

- **Integration Testing**

Тестване дали комбинирани части на приложението функционират заедно правилно; "частите", които се интегрират могат да бъдат модули, подсистеми, индивидуални приложения, приложения на клиент и сървър на мрежа, и т.н. Тестват се дали интерфейсите между отделните компоненти са дефинирани и реализирани коректно. Проверява се дали отделните компоненти на системата работят заедно, както е описано в спецификацията и дизайна на системата. Ако тук се приложи тестването, базирано на модели, много по-рано ще се открият несъответсвията между изискванията и реализацията на системата.

1.2.2 Тестване според характеристиките, които се тестват

Според характеристиките, които се тестват може да се определят няколко основни видове тестване:

- Тестване на сигурността (Security) - тестване на степента на защита на системата от вътрешен и външен неоторизиран достъп.
- Издръжливост (Robustness) - тества се дали системата се държи коректно при появата на неочаквани входни данни. Прилагат се известните "stress" тестове за особено голямо натоварване на потребителския интерфейс, широка гама от запитвания към голяма база данни и др.
- Производителност (Performance) - измерва се времето за извършване на дадена работа.
- Използваемост (Usability) - тества се използваемостта на система според дефинирани стандарти.
- Надежност (Reliability) - тества се надеждността на системата.
- Финален (Acceptance) – финален /приемателен/ тест, базиран на изискванията на крайните потребители; или определен за използване от потребителите за определен период от време.
- Възстановяване (Recovery) – тестване на степента на възстановяване на системата след софтуерен или хардуерен срив
- Съвместимост (Compatability) – тества се съвместимостта на софтуера по отношение на хардуер, операционни системи, мрежи, browsers /за web приложения/ и др.
- Повторно-тестване (Regression) – повторно тестване след модифициране или добавяне на нови функционалности на софтуера, или промяна на средата /environment/, в която ще функционира; при този тип тестване е особено подходящо

използването на автоматизирани тестови средства (testing tools).

1.2.3 White-box и black-box тестване

След като са завършени дизайна и реализацията на дадено приложение или част от него, то трябва да се тества. Проверяват се дали са изпълнени функционалните изисквания и дали успешно са реализирани. Изпълняват се тестове, които проверяват възможните сценарии на употреба. Така нареченото функционално тестване бива white-box и black-box.

- White-box

Основава се на познания за вътрешния дизайн и / или на програмен код – вътрешна програмна логика, оператори, процедури, условия и т.н.

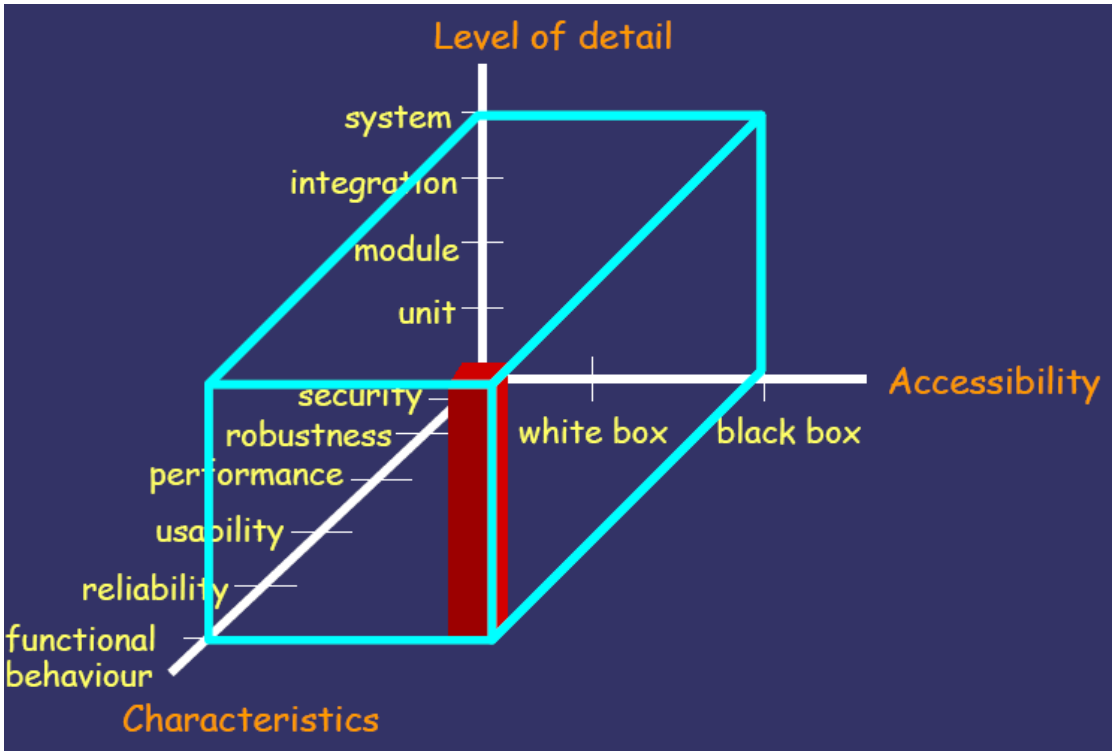
- Black-box

Не е основано на каквото и да е познание за вътрешен дизайн или програмен код на софтуерния продукт; тестовете са основани единствено на функционалните и технологичните изисквания на потребителите.

1.3 Анализ на тестването базирано на модели

Може да се дефинира, че тестването базирано на модели е "black-box" тестване, в което определянето на тестовите случаи и верификацията на тестовете са базирани на модел на очакваното

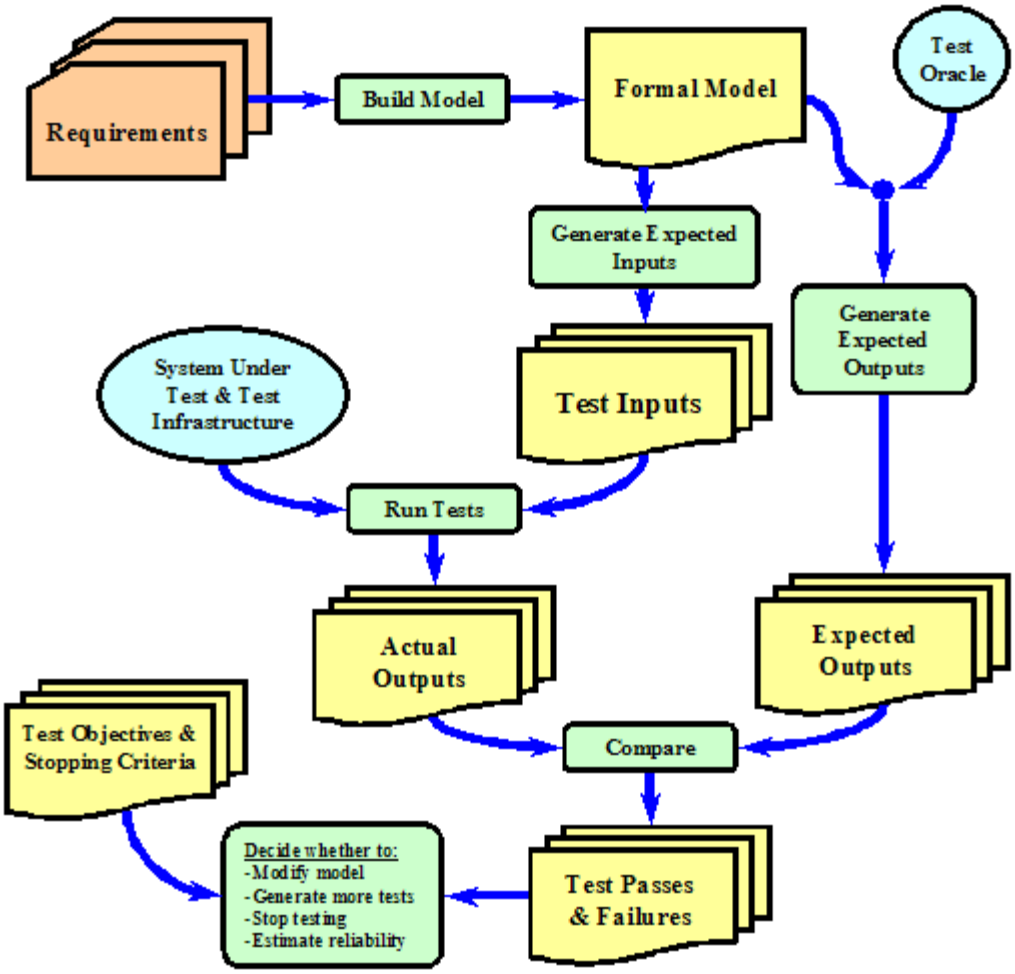
поведение на системата [3]. На фигура 2 е показано как тестването базирано на модели е свързано с други методи на тестване.



Фигура 2 Тестване базирано на модели и други методи на тестване

Изграждането на тестването базирано на модели започва от изискванията на системата. За да се изгради модела е необходимо много добро познаване на системата, която ще се тества, характеристиките на потребителите, условията при които дадено действие може да се случи [1]. Определеният модел се използва за генериране на тестови случай (test cases). Спецификацията на тестовите случай включва информация за очакваните входни данни. Моделът може да генерира информация за входните данни, такива каквито се очакват от системата. След изпълняването на тестовете, входните данни се сравняват с очакваните. Характеристиките се използват, за да се идентифицират грешките в системата. Тестовите данни се използват също, за да се направят

сравнения, до каква степен тестването е покрило функционалността.



Фигура 3 Основните дейности при тестването базирано на модели

На фигура 3 е показано как тестването базирано на модели е свързано с автоматизираното тестване. Основните дейности при тестването базирано на модели са оцветени в зелено [1] – изграждането на модел, генериране на очаквани входни и изходни данни, стартиране на тестове и т.н.

Специфичните действия са:

- Изграждане на модел
- Генериране на очаквани входни данни
- Генериране на очаквани изходни данни
- Стартиране на тестове
- Сравнение на действителните входни данни със очакваните такива
- Определяне на следващи действия (дали да се променя модела, генериране на още тестове, спиране на тестването и т.н.)

Тестването базирано на модели, дава резултат в следните области [1]:

- Повишава се комуникацията между разработчиците и тестерите
- Възможност за ранно откриване на различия в спецификацията и дизайна
- Има възможност за автоматично генериране на множество тестове. Чрез много от другите подходи тестовете се генерират ръчно.
- Лесно подновяване на тестовете след промяна на спецификацията.
- Способност за оценяванае на regression тестове.
- Възможност да се определи модела на поведение на системата, според изискванията на потребителя.

Глава 2.