Obsah

Kapitola 1
Úvod

Testy řízené programování nebo také TDD (Test Driven Development)je metodika realizace softwarového projektu, která spočívá v psaní testů pro jednotlivé části programovaného softwaru podle zdokumentovaných požadavků ještě před tím, než je kód dané části napsán.

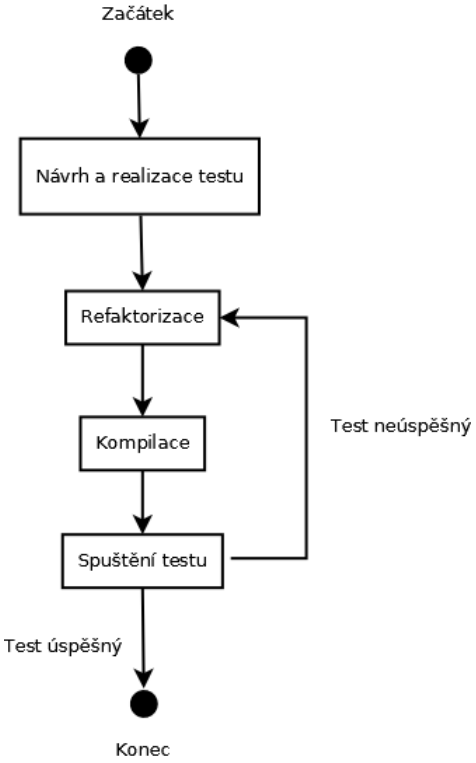
Opakem TDD je postup kdy je napsán kód pro danou část sotwarového projektu a na základě požadavků je testována jeho funkčnost. Na tomto přístupu není nic špatného, ale je náchylný na duplikace kódu a nadbytečnosti (programátor má tendenci před otestováním přidávat nadbytečný kód, který rozšiřuje základní, ale požadovanou funčknost kódu).

Výhodou TDD oproti klasickému způsobu programování je jednoduchý a rychlý přehled, které části jsou již hotové a odladěné, které části jsou již napsané, ale je nutné opravit a částí, které je třeba ještě dodělat.

TDD úzce souvisí s testováním softwaru, protože využívá techniky a postupy pro otestování funkčnosti jednotlivých částí programu, aby byl eliminován výskyt chyb ve výsledném programu.

Kapitola 2
Proces vývoje řízeného testy

Podstatou TDD je navrhnout a vytvořit test a zkompilovat. Kompilace takového kódu vždy skončí chybou, protože testy se snaží volat prozatím neexistující kód. Pomocí kompilace je ale možné zjistit které části programu je nutné upravit, aby byla kompilace úspěšná. Následuje refaktorizace kódu tak aby byl kód zkompilovatelný. dalším krokem je postupné vytváření části programu, která je testována. Přitom jsou cyklicky spouštěny testy, které ověřují funčknost nově vytvořeného kódu.



Kapitola 3
Unit test

3.1 Požadavky na testy

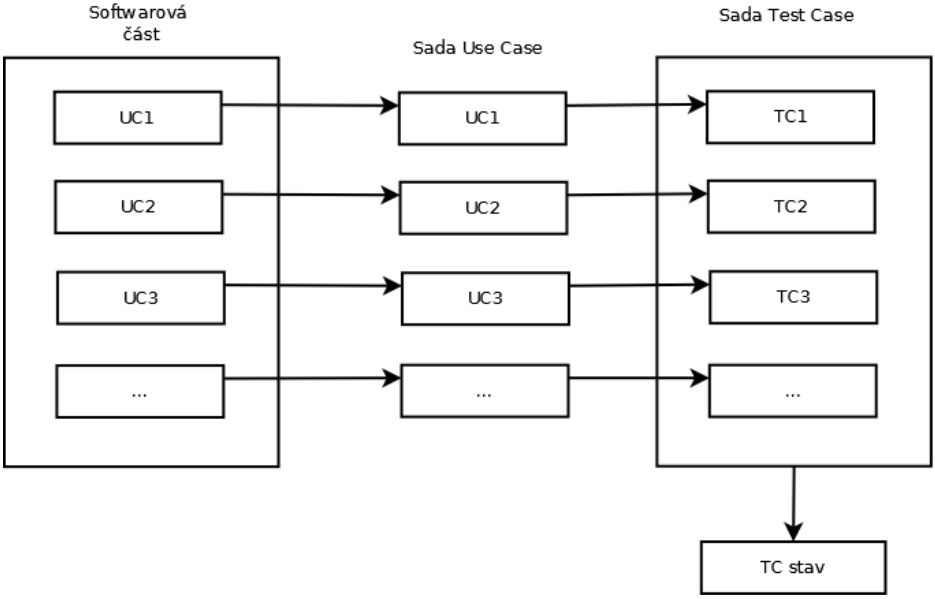
Test by měl být multiplatformní, to znamená, že při spuštění na libovolné platformně pro kterou je daný software portován by měl fungovat stejným způsobem. V opačné případě by došlo ke zkreslení výsledků a vníšení chyb do softwaru.

Test by měl být také opakovatelný, tedy při nekonečně velkém počtu spuštění by měl daný test se stejnými parametry vracez stále stejné výsledky, pokud není navržen k opaku, nebo výsledek testu není ovlivněn vnějšími podmínkamy.

Kapitola 4
Abstraktní vrstva

Kapitola 5
Test mode

Test mode nebo také testovací režim je vnitřní stav vyvíjeného programu, který se vnitřně sestává z dilčích programových částí a sady testů, které mají za úkol jednotlivé části otestovat. Tetovací režim by neměl být součástí výsledného programu a návrh softwarové architektury s test modem by měl být takový, aby neaktivní části test modu tvořily mrtvý kód.



Test mod se v softwarovém projektu nachází v samostatném souboru, který je buď spouštěn samostatně (pokud to programovací jazyk umožňuje) a nebo je pomocí podmíněného překladu vkládán do hlavní programové funkce.

5.1 Test Case

Test Case označovaný zkratkou TC nebo také **testovací případ** popisuje konkrétní akce prováděné s určitou softwarovou komponentou a jejich očekávané výsledky. Softwarovou komponentou v tomto případě může být například část aplikačního rozhraní, funkce, třída nebo také softwarový systém běžící na několika strojích souběžně. Test Case se používá pro automatizované testy nebo pro manuální testy.

Testovací případ je tedy dokument, popisující určitou činnost, kterou je potřeba otestovat. Obecně lze říci, že obsahuje kroky se skutečnými vstupními hodnotami spolu s očekávanými výsledky. Test Case by měl obsahovat informace:

- ID - jedinečný identifikátor, který identifikuje daný TC, slouží pro odkazování v jiných dokumentech
- Účel - jednoduchý a krátký popis co daný TC ověřuje
- Podmínky - seznam potřebných dat potřebných pro provedení daného testu
- Specifikace vstupních dat
- Specifikace kroků - seznam kroků nutných pro provedení testu
- Očekávané výsledky - souhrn všech informací potřebných k určení zda daný test uspěl či nikoliv

5.2 Automatizované testy

Automatizované testovací případy se také někdy označují jako **testovací skript**. Tvoří je sada programových instrukcí a na rozdíl od manuálních testů by automatizované testy měly být schopny sami rozpoznat, zda uspěly či selhaly. Automatizované testy se skládají ze sady unit testů, které umožňují určit zda test dané části uspěl či selhal.

Hlavním cílem automatizace testů softwaru je časová úspora při jejich spouštění. Obecně lze říci, že automatizace testů softwaru má za účel usnadnit a zefektivnit provádění postupu testování softwaru. Pokud jsou testy prováděny zcela automaticky, tj. bez možnosti zásahu lidského faktoru, výrazně se tím snižuje možnost chybného provedení postupu testování softwaru a tím také pravděpodobnost bezporuchového provozu softwaru.

5.3 Mauální testy

Ne vždy se vyplatí nebo je možné vytvářet automatizované testy. V takovém případě přicházejí na řadu manuální testy, kde tester/programátor ručně navodí požadované podmínky a následně zkontroluje výsledky. Pro účely manuálního testování jsou tvořeny seznamy prováděných kroků a očekávaných výsledků.