Podvedomý prístup k testovaniu softvéru

PETER DUŠEK

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava

Abstrakt. Testovanie je dôležitou fázou v etape vývoja softvéru. V tejto eseji opíšem vplyv ľudského podvedomia pri testovaní. Rozoberiem rozdiel medzi testovaním a inšpekciou, kedy a ako ich použiť. Už definícia procesu testovania býva často krát nesprávna, preto sa na tento pojem pozriem bližšie. Pri testovaní je vhodné dodržiavať určité princípy, ktoré sú vodítkom pre dosiahnutie požadovaných výsledkov. V článku ich popíšem podrobnejšie. Ďalej sa zameriam na roly v tíme testerov a ich zodpovednosť. Taktiež opíšem vhodnosť oddelenia vývojového a testovacieho prostredia.

Úvod

Testovanie je dôležitou fázou vývoja softvéru. Pán Myers [1] tvrdí že „...najdôležitejšie vplyvy na testovanie softvéru má psychológia a ekonomika“. Obzvlášť psychológia. Testovanie je výzvou, je viac o objavovaní ako iné disciplíny vo vývoji softvéru. Časy, kedy boli ľudia trestaní za objavené chyby sú už minulosťou a je snaha odhaliť ich čo najskôr.

Psychológia a ekonomika testovania programov

Jednou z primárnych príčin slabého testovania softvéru je fakt, že veľa programátorov začína s nesprávnou definíciou samotného testovania.
Vychádzajú z úvah:

- Testovanie je proces, ktorý preukáže, že v programe sa nevyskytujú chyby.
- Cieľom testovania je ukázať, že program vykonáva požadovanú funkciu správne.
- Testovanie je proces, ktorý vytvára presvedčenie, že program robí to, čo sa od neho očakáva.

Manažment v softvérovom inžinierstve, máj 2005, s. 1-7.

Takéto definície sú ale nesprávne. Testovaním chceme softvéru pridať určitú hodnotu. Pridaná hodnota pomocou testovania znamená zvýšenie kvality a spoľahlivosti programov. Pričom zvyšovanie spoľahlivosti znamená hľadanie a odstraňovanie chýb.

Z toho dôvodu, netestujme programy, aby sme preukázali že pracujú, ale mali by sme vychádzať z predpokladu, že program obsahuje chyby (býva to pravdivý predpoklad takmer pre každý program) a pokúsiť sa nájsť z nich čo najviac.

Vhodnejšia definícia je potom:

Testovanie je proces vykonávania programu s úmyslom nájsť chyby.

Ľudia sú cieľavedomí, s úmyslom dosiahnuť čo si stanovia a stanovenie primeraných cieľov má významný psychologický efekt. Ak je naším cieľom preukázať, že program nemá chyby, potom budeme k tomu podvedome vedení. Môže to byť pri výbere testovacích dát a pravdepodobnosť, že objavíme chybu sa zníži.

Naopak, pokiaľ je naším cieľom ukázať, že softvér obsahuje chyby, je väčšia pravdepodobnosť, že naše testovacie dáta zistia chybu.

Iný spôsob ako upevniť definíciu testovania je analyzovať používanie slova „úspešný“ a „neúspešný“ najme v konkrétnych prípadoch kedy projektoví manažéri zaraďujú výsledky ich testovacích prípadov. Mnohí z nich volajú test, ktorý neodhalí chybu ako „úspešný test“ a tie ktoré odhalia nedostatok ako „neúspešný test“.

Skúsme si toto predstaviť na príklade kedy pacient navštívi svojho lekára s tým, že sa celkovo necíti dobre, je slabý a pravdepodobne je chorý.

Lekár vykoná niekoľko laboratórnych testov, ktoré nepotvrdia chorobu a neurčia diagnózu. Takéto testy by sme nenazvali „úspešné“, pretože boli neúspešné a výdavky na vykonanie testov mohli byť neprimerane vysoké. Pacient je naďalej chorý a môže ho zaujímať odbornosť lekára pri určovaní diagnózy. Ak testy niečo preukázali, boli úspešné a lekár môže začať s adekvátnou liečbou. Analogicky si môžeme pri testovaní predstaviť softvér v úlohe chorých pacientov.

Testovaním je prakticky nemožné ukázať, že chyba sa v programoch nevyskytuje, aj v prípade väčšiny triviálnych programov. Väčšie pokroky je možné očakávať v začiatočnej fáze testovania a čím je dlhšie proces testovania trvá, tým menšie pokroky možno pozorovať.

Aj podľa tretej uvedenej definície „Testovanie je proces, ktorý vytvára presvedčenie, že program robí to, čo sa od neho očakáva“, a softvér spĺňa požiadavky používateľa i napriek tomu môže obsahovať chyby. Tie sa môžu prejaviť neskôr v kritických situáciách.

Z iného pohľadu je testovanie hľadaním prípadov, kedy nie je splnená špecifikácia. Aktivity ako prehliadka, inšpekcia a statická analýza zdrojového kódu môžeme nazvať testovanie, kedy program nie je vykonávaný. Takéto aktivity môžeme nazvať statické testovanie.

Dynamické testovanie je počas vykonávania programu. Sem patrí funkcionálne testovanie, kedy zisťujeme či vstupno-výstupné správanie vyhovuje špecifikácii. Bez ohľadu na logiku systému môžeme zostaviť množinu testovacích vstupov.

Štruktúrované testovanie je na základe vnútornej štruktúry, pričom ide o pokrytie tokov riadenia aplikácie alebo o údaje. Ďalším je testovanie rozhraní jednotlivých podsystémov a overenie komunikácie medzi jednotlivými modulmi.

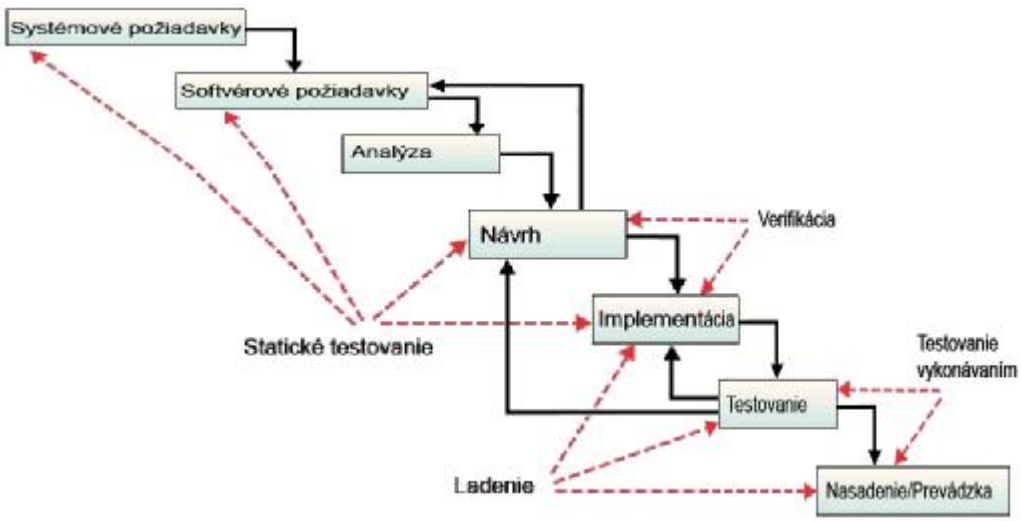
Životný cyklus a testeri

Testeri by mali veľmi dobre rozumieť produktu, aby boli schopní dobre navrhnúť testovacie plány, návrhy, procedúry a jednotlivé testovacie prípady [3]. Účasť testovacích tímov v skorších fázach vývoja eliminuje zmätok okolo funkcionálnych požiadaviek, ktorý by neskôr mohol vzniknúť. Rovnako testeri získajú prehľad o aspektoch, ktoré sú pre koncových používateľov kritické.

Na vodopádovom modeli životného cyklu softvérového projektu (pozri Obr. 1) si ukážeme do ktorých fáz zasahuje testovanie.

Cieľom ladenia je opraviť chybu v implementácii - uplatňuje sa počas samotnej implementácie, pri testovaní, počas prevádzky a údržby. Pri verifikácii ide hlavne o overenie funkcionálnej správnosti, ktorá závisí od kvalitného návrhu a implementácie.

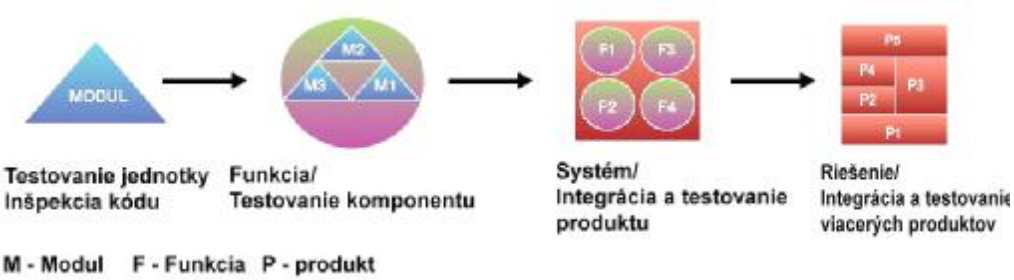
Testovanie v prvých fázach je statické, jedná sa o formálne prehliadky dokumentov a zdrojového kódu. Testovanie vykonávaním prihliada na aspekty ako bezpečnosť, výkon systému, požiadavky na hardvérovú a softvérovú konfiguráciu, zaťaženie systému a prípadné zotavenie. Typickým pri nasadení sú zákaznícke akceptačné kritériá.



Obr.1. Aktivity zahŕňajúce ladenie, testovanie a verifikáciu v typickom softvérovom vývojovom procese [3]

Rozsiahle a zložité systémy možno jemnejšie rozčleniť na podsystémy, komponenty a moduly, ktoré sú spojené v jeden celok. Na Obr. 2. je znázornené takéto rozčlenenie, ktoré nám určuje stratégie testovania. Testovanie v neskorších fázach na úrovni systému alebo riešenia je typu „čierna skrinka“ a nevyžaduje detailné znalosti o implementácii.

Pokiaľ nás zaujíma štruktúra programu ide o testovanie typu „biela skrinka“. Je možné testovať napríklad súdržnosť a zviazanosť jednotlivých modulov, komponentov.



Obr.2. Typické stratégie testovania na jednotlivých úrovniach

Testovanie a inšpekcia

Oboje, testovanie aj inšpekcia sú súčasťou verifikácie a validácie počas vývoja softvéru. V istých prípadoch plnia rovnakú úlohu, obe majú rovnaký cieľ, zvýšiť kvalitu produktu. Je veľmi dôležité odhaliť chyby čo najskôr, pretože často krát sa 40 až 50 percent vývoja venuje následnej oprave nedostatkov. Čo predstavuje veľké množstvo času ktoré možno získať. Inšpekcia je zameraná na odhaľovanie chýb, zatiaľ čo testovanie je zamerané hlavne na hľadanie zlyhaní [2].

Prečo používať inšpekciu

Inšpekcia dokumentov je dôležitá od začiatku projektu. Jej úlohou je nájsť všetky chyby v každej fáze procesu a tak pokračovať v ďalšej fáze so správnymi podkladmi.

Inšpekciu možno vykonávať mnohými rôznymi spôsobmi, rôznymi technikami čítania a podobne. Ak prídeme k hľadaniu chýb v zdrojových kódoch, môže byť ťažký výber medzi inšpekciou a testovaním. Správna voľba závisí od toho, čo hľadáme. Pomocou inšpekcie môže byť oveľa jednoduchšie odhaliť chyby v štandardoch programovania, než pomocou testovania. Rovnako v prípadoch ladenia programu, keď zisťujeme čo je zle. Podľa pána Sommervilla [4] je inšpekcia oveľa efektívnejšia. Spomína dva dôvody prečo je tomu tak. Po prvé, pri jednom sedení možno odhaliť veľa nedostatkov. Je to preto, že rôzne chyby často ovplyvnia vykonávanie programu a niektoré iné chyby sa potom neobjavia ak program testujeme.

Druhým dôvodom je skúsenosť inšpektorov v problémovej oblasti. Samozrejmosť je skúsenosť programovacieho jazyka a znalosť bežných chýb v zdrojových kódoch s ktorými sa už v minulosti stretli. Kompilátor striktne sleduje logickú postupnosť kódu, ale ľudia sa zameriavajú aj na obsah, ktorý sa môže vyskytnúť mimo normálneho vykonávania programu. Ale ľudia tiež robia chyby... Inšpekcia je lepšia pri hľadaní chýb v návrhu, v dokumentoch požiadaviek, v zdrojových kódoch a podobne. Záleží na znalostiach a predchádzajúcich skúsenostiach členov tímu a od ich celkových poznatkov z problémovej oblasti, inak je možné niektoré dôležité skutočnosti prehliadnuť.

Prečo používať testovanie

Testovanie môže v istých prípadoch predstavovať až polovicu času vývoja softvéru. Dôležitým faktorom je aj to, či je zákazník schopný akceptovať chyby, ktoré sú opravované počas používania vo fáze údržby, čím sa môžu často krát znížiť náklady na vývoj.

Testovanie je jediný spôsob ako odhaliť operačné zlyhania a ako overiť, že nefunkcionálne požiadavky sú splnené na požadovanej úrovni. Na rozdiel od inšpektorov, testeri sa väčšinou riadia jednotlivými testovacími prípadmi, ktoré by mali dodržiavať. Výhodou testovania je blízkosť s používaním po nasadení. Koncoví používatelia majú lepší dojem z produktu a cítia vyššiu kvalitu, pretože zlyhania sú mimo bežného používania, ktoré bolo testované. Produkt sa po vylúčení zlyhaní počas bežnej prevádzky stáva výhodnejším pre nasadenie. Často krát sa stáva, že fáza testovania má menšiu prioritu než ostatné. Ak sa projekt omešká, testovanie sa skracuje. Špeciálne ak je testovanie výlučne poslednou fázou a nie počas celého trvania projektu. Testovanie je niekedy jedinou možnosťou, ak napríklad nemáme prístup k zdrojovým kódom programov, využívame komponenty od tretích strán a rôzne API funkcie. Tak isto je potrebné vykonávať testovanie v rozdielnych prostrediach.

Vzhľadom na uvedené skutočnosti je lepšie vykonať najprv inšpekciu a potom testovanie. Samozrejme v praxi obľúbenejším prístupom býva práve opačný, najprv testovania a potom inšpekcia, ak vôbec.

Princípy testovania softvéru

Pri testovaní je vhodné dodržiavať určité zásady, ktoré prispejú k zvýšeniu kvality testovania. [4]

1. Nevyhnutnou časťou testovacích prípadov je definovanie očakávaných výstupov a výsledkov.
2. Programátor by sa mal vyhnúť pokusom testovať svoj vlastný program.

3. Organizácia zaoberajúca sa tvorbou softvéru, by nemala testovať svoje vlastné produkty.
4. Vykonávať dôkladne inšpekciu výsledkov každého testovania.
5. Testovacie prípady musia byť napísané pre vstupné podmienky, ktoré sú neplatné a neočakávané, tak isto ako pre platné a očakávané vstupy.
6. Preskúmanie či program vykonáva požadované úlohy je len polovičná práca, rovnako dôležité je zistiť, či nevykonáva to čo nemá.
7. Vyhnite sa jednoúčelovým testovacím prípadom, pokiaľ to nie je nevyhnutné, teda ak program tiež nie je jednoúčelový.
8. Neplánujte si úsilie testovať softvér s predpokladom, že chyby nebudú odhalené.
9. Pravdepodobnosť výskytu viacerých chýb vo funkčnom bloku programu je priamo úmerná počtu už nájdených chýb v danom bloku.
10. Testovanie je veľmi kreatívnou a intelektuálnou výzvou.

Definovanie rolí a zodpovednosti testerov

Schopnosti testovacích tímov môžu výrazne ovplyvniť úspech prípadný neúspech testovania. V závislosti od problémovej oblasti je vhodné mať v tíme členov, ktorí detailne ovládajú túto oblasť. Tieto vedomosti im umožňujú efektívne vytvoriť testovacie prípady, skripty a iné mechanizmy pre overenie požiadaviek, funkčnosti, správnosti a iných kritérií softvérových produktov. Takéto tímy sa zväčša skladajú z niekoľko desiatok pracovníkov, ktorí majú svoje úlohy. Ich zodpovednosť vyplýva z ich pozície a schopností.

Manažér
Mal by rozumieť procesom a metodológiám testovania, byť schopný určiť ciele, objekty a stratégie. Tak isto rozumieť manuálnym a auto matickým technikám testovania. Byť oboznámený s rôznymi podpornými nástrojmi, byť dobrý v plánovaní vrátane ľudských zdrojov a zariadení.

Vedúci tímu
Mal by rozumieť obchodným požiadavkám v danej oblasti a aplikačným požiadavkám, byť expertom v rôznych technických prostriedkoch, v programovacích jazykoch, databázových technológiách a operačných systémoch. Taktiež pokročilé znalosti a skúsenosti v procesoch testovania.

Tester
Mal by byť schopný v návrhu testovacích prípadov, profesionál v návrhu GUI štandardov, profesionál v jednotlivých fázach testovania softvéru.

Tester, odborník na siete
Mal by byť odborníkom na databázové a sieťové technológie, tak isto aj správu operačných systémov.

Tester, odborník na bezpečnosť
Mal by byť odborníkom na nástroje a techniky ohľadom počítačovej bezpečnosti.

Oddelenie vývojového a testovacieho prostredia

V niektorých prípadoch je vhodné oddeliť vývojové prostredie o testovacieho prostredia. Testovacie centrá bývajú špecializované a vedia vytvoriť niekoľko rôznych prostredí, či už rôzne platformy, konfigurácie zostáv, prípadne nasimulovať rôzne druhy záťaže systémov. Pre vývojárov je vhodné nechať otestovať svoje produkty na takýchto pracoviskách, kde pracuje tím profesionálov.

Záver

V tomto článku som sa pokúsil zhrnúť základné myšlienky o testovaní softvéru, napísal a zdôvodnil som definíciu testovania, ktorú by sme si mali uvedomiť pri každom testovaní. Ľudí je vhodné nasadiť na testovanie už v počiatočných fázach procesu vývoja, čo nám prinesie úsporu nákladov, času a zvýši kvalitu produktu. Testovanie a inšpekcia sa nevylučujú, je vhodné si uvedomiť ich vhodnosť použitia, pretože sa navzájom dopĺňajú.

Použitá literatúra

1. Phillip G. Armour: *The Unconscious Art of Software Testing*, COMMUNICATIONS OF THE ACM January 2005/Vol. 48, No.1
2. Gustav Evertsson: *Inspection vs. Testing*, Blekinge Institute of Technology, Software Verification and Validation (PAD001), 2002-11-20
3. Hailpern and Santhanam: Software debugging, testing, and verification. IBM Systems journal, vol 41, no 1,2002
4. Dustin, Elfriede: *Effective software testing : 50 specific ways to improve your testing* (Addison-Wesley) December 2002
5. I. Sommerville, Software Engineering, Excerpt: *Verification & Validation*, Adison-Wesley, 2000

Annotation

Subconscious approach to software testing

According to article in ACM I will to contribute to theme „*Subconscious approach to software testing*“. Testing is an important phase in the software development process. In this article I will inscribe impact of human been subconscious by testing. I will discuss the difference between testing and inspection and when to use them. Often the definition of testing is wrong, I'll try to formulate a proper one. It is good to respect some principles of testing, when performing it. Next I will focus on roles in testers team and their responsibility. Also I would like to describe suitability of development and testing environment separation.