Testovanie v tímovom projekte

MICHAL DÁVID

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava
xdavid[zavináč]is[.]stuba[.]sk

Abstrakt. Pri vytváraní softvérového produktu sa vývojový tím snaží predovšetkým vytvoriť kvalitný softvér. Kvalita sa stala hlavným kritériom pri hodnotení softvérového produktu. Avšak vytvoriť kvalitný produkt hneď na začiatku je veľmi náročné, ba až skoro nereálne. Preto sa na maximálne zvýšenie kvality používa testovanie. Testovaním sa odhalia chyby, ktoré by v prípade nasadenia do prevádzky mohli spôsobiť problémy a ich odstraňovanie by bolo veľmi nákladné, ale taktiež sa testovaním zisťujú prípadné námietky alebo postrehy zo strany zákazníka. Cieľom tejto eseje je v krátkosti priblížiť čitateľovi atribúty hodnotenia kvality, dôležitosť testovania a hlavne poukázať a vysvetliť metódy, ktoré sú najvhodnejšie na použitie v tímovom projekte.

Úvod

Každý tím, ktorý vyvíja nejaký softvérový produkt sa snaží hlavne o jednu vec. Je ňou kvalita vyvíjaného produktu. Čím kvalitnejší produkt vyrobia tým sa zvýši spokojnosť zákazníka a zároveň sa buduje dobré meno firmy (tímu) v tejto oblasti. Vyrobiť kvalitný softvér však nie je vôbec jednoduché a vytvoriť kvalitný softvér hneď na prvý krát je dokonca takmer nemožné. Ako je teda možné dodatočne vniesť do vytvoreného produktu ten prívlastok „kvalitný“? Keďže kvalita nie je jediná vlastnosť ale akýsi súbor vlastností tak jednou z možných odpovedí je testovanie samozrejme za predpokladu, že systém je vyvíjaný správnym spôsobom.

Testovanie sa využíva počas celého životného cyklu. Čím skôr sa testovaním príde na nejaké chyby, tým jednoduchšia je ich oprava a s tým súvisia aj nižšie náklady na opravu. Testovanie teda sprevádza vyvíjaný softvérový produkt od jeho zrodu až po odovzdanie zákazníkovi.

Z predošlých riadkov je možné vyvodiť záver, že čím dlhšie testovanie, tým sa odhalí viac chýb a zároveň sa softvér stáva kvalitnejším. Samozrejme že testovanie môže byť teoreticky nekonečné a preto je nutné rozhodnúť, kedy je už ďalšie testovanie viac-menej zbytočné pretože pravdepodobnosť odhalenia ďalšej chyby kopíruje približne logaritmickú funkciu teda odhalenie ďalšej a ďalšej chyby je časovo

Manažment projektov softvérových a informačných systémov, október 2008, s. 1-6.

stále náročnejšie a náročnejšie. Samozrejme pri testovaní je dôležité vybrať aj najvhodnejšiu metódu pre daný projekt.

Kvalita softvéru

Pri vývoji softvéru sa už od jeho zrodu dbá hlavne na kvalitu. Podľa [3] kvalita znamená prispôsobenie sa špecifickým požiadavkám. Súbor procesov, ktorý sa stará práve o zabezpečenie potrebnej úrovne kvality sa nazýva manažment kvality softvéru. Medzi základné procesy manažmentu kvality softvéru patria podľa normy IEEE12207 [4] nasledovné:

- Procesy na zabezpečenie kvality
- Procesy verifikácie
- Procesy validácie
- Procesy revízie
- Procesy auditov

Štandard IEEE12207 [4] definuje zabezpečenie kvality ako proces, ktorý zabezpečuje, že softvérové produkty spĺňajú špecifické požiadavky a dodržiavajú stanovené plány. Keďže tento proces beží počas celého vývoja softvérového produktu, vykonávanie jeho aktivít odhaľuje problémy, ktoré sa pri komplikovanej činnosti vyskytujú neustále, už pri ich zárodku.

Kvalitu nemôžeme testovať priamo ale je niekoľko faktorov, ktoré ju môžu spraviť viditeľnou. Kvalita má tri druhy faktorov. Funkcionalita, spracovanie a adaptabilita. Každý z uvedených faktorov sa dá ďalej rozložiť na menšie časti, ktoré sú zobrazené v nasledujúcej tabuľke [3].

Tab.1. Typické faktory softvérovej kvality

Funkcionalita (vonkajšia kvalita)	*Spracovanie (vnútorná kvalita)*	*Adaptabilita (kvalita pre budúcnosť)*
Správnosť	Efektivita	Flexibilita
Spoľahlivosť	Testovateľnosť	Znovupoužiteľnosť
Použiteľnosť	Dokumentácia	Udržiavateľnosť
Integrita	Štruktúra	

Verifikácia a validácia

Ako už nadpis nasvedčuje, ďalším dôvodom testovania je verifikácia a validácia. Pre úplnosť uvediem rozdiel medzi týmito dvomi pojmami. Verifikácia je proces pri ktorom sa uisťujeme, že vyrábame výrobok podľa navrhnutej štruktúry. Validácia je

proces, kde zisťujeme či výrobok, ktorý vyrábame, je naozaj ten, aký potrebuje zákazník. Teda či je vôbec špecifikácia v poriadku.

Z uvedeného je vidieť, že verifikácia predchádza validáciu. Bolo by totiž nezmyslom začať implementovať niečo skôr ako si budeme istí, že to čo implementujeme je naozaj to, čo sa od nás očakáva. Hlavným cieľom týchto dvoch techník je odstránenie závažných chýb ešte pred nasadením do prevádzky. Pri verifikácii sa využívajú prevažne statické metódy testovania. Študovanie plánov, dokumentácií, vyhodnocovanie existujúcich riešení a pod. Pri validácii sa naopak využívajú dynamické metódy testovania.

Verifikáciu a validáciu môžu vykonávať ľudia, ktorí nie sú ani zákazníci ani vývojári. Ak sú na tento účel vybratí práve takýto ľudia zabezpečí sa tým nestrannosť čo je výhodou.

Revízie

Revízie a audity sú podporné procesy verifikácie a validácie. Podľa štandardu IEEE 1028 je revízia proces alebo stretnutie, pri ktorom sa softvérový produkt prekontroluje projektovým personálom, manažérmi, používateľmi, zákazníkmi, prípadne ďalšími oprávnenými osobami. Podľa IEEE 1028 je známych niekoľko druhov revízií:

- Manažérska revízia – používa sa na monitorovanie priebehu, určovanie stavu plánov, potvrdenie požiadaviek a ich systémové rozloženie, alebo ohodnotenie efektívnosti manažmentu.
- Technická revízia – cieľom je ohodnotenie softvérového produktu na rozhodnutie, či je vhodný na jeho určené použitie. Navyše sa pomocou nej identifikujú rozpory so schválenou špecifikáciou a štandardmi.
- Inšpekcia – účelom je odhalenie a identifikovanie softvérových anomálií. Zvyčajne je zameraná iba na malú časť produktu. Pomocou inšpekcie je možné skontrolovať nefunkcionálne charakteristiky.
- Previerka – je podobná inšpekcii avšak menej formálna. Hlavnými cieľmi sú nájdenie anomálií, zlepšenie produktu, uváženie viacerých alternatív a vyhodnotenie súladu so štandardami a špecifikáciami.
- Audit – snahou je poskytnúť nezávislé ohodnotenie zhody softvérového produktu a procesov s aplikovateľnými reguláciami, štandardmi plánmi a procedúrami.

Problémy testovania

Podľa [1] je testovanie súbor procesov slúžiacich na kontrolu kvality softvérového produktu, ktorých cieľom je dosiahnutie požadovanej kvality softvéru z hľadiska funkčnosti, spoľahlivosti, výkonnosti, použiteľnosti a podporovateľnosti. Kontrola

kvality sa môže uskutočniť či už pre jednotlivé časti systému alebo pre systém ako celok. V minulosti sa totiž testoval iba výsledný produkt, zatiaľ čo dnes sa testujú aj jednotlivé komponenty. Niekedy nie sme schopní odhaliť chybu pri testovaní v prostredí kde systém vyvíjame. Pri testovaní teda vyhľadávame chyby, ktoré nepoznáme. Niektoré chyby sa môžu prejaviť až pri nasadení systému do prostredia v ktorom bude nasadený. To je často spôsobené tým, že vyvíjaný systém má tvoriť časť iného systému, alebo s iným systémom spolupracovať. V takomto prípade je ťažké testovať systém na situácie, ktoré môžu nastať pri spolupráci s inými systémami. Ak teda nie je možné testovať v reálnom prostredí je potrebné toto prostredia aspoň nasimulovať. To je však veľmi závislé od toho ako dobre toto prostredie poznáme a ako dostatočne sa nám podarí simulované prostredie napodobniť.

Treba si uvedomiť, že testovanie nie je exaktná veda a preto neexistuje nijaký postup na správne testovanie. Myslím si, že aj pri testovaní v tímovom projekte nastáva práve takýto problém. Vytvorený softvér je vyvíjaný v inom prostredí v akom bude skutočne nasadený. Preto sa bude musieť testovanie vykonávať aj pri samotnom vytváraní ale taktiež sa bude musieť umiestniť na svoje trvalé stanovisko a testy sa budú musieť uskutočniť aj tam.

Techniky testovania

Techniky testovania poznáme statické a dynamické. Statické techniky som už spomínal. Sú to techniky, pri ktorých sa priamo nevykonáva vyvíjaný softvér. Jedná sa len o kontrolovanie špecifikácií a dokumentácií. Dynamické metódy sú už zamerané na priame vykonávanie testovaného softvéru. Softvéru sa zadávajú vopred precízne vybraté vstupy a sú známe očakávané výstupy. Ak sa výstupy z testovaného softvéru rovnajú (prípadne sú v nejakej tolerancii) môžeme povedať, že softvér je správny. I keď to hneď neznamená, že je aj kvalitný je predpoklad, že sa ním môže stať.

Softvér sa môže testovať na rôznych úrovniach. Je veľmi dôležité si uvedomiť, ktorá úroveň sa práve testuje, pretože každá má iné nároky. Treba si uvedomiť, že testovanie nikdy neodhalí všetky chyby a testovanie na jednotlivých úrovniach má obrovský vplyv na výslednú kvalitu. Všeobecne môžeme rozlišovať tri základné úrovne testovania:

- Testovanie komponentov – overujú sa jednotlivé fungujúce a testovateľné komponenty a skúma sa ich implementácia na základe návrhu.
- Testovanie integrity – je overovanie integrity a komunikácie medzi jednotlivými komponentmi. Odhaľuje problémy, ktoré mohli vzniknúť pri spájaní jednotlivých komponentov do komplexného systému.
- Testovanie systému – testuje sa kompletný systém ako celok a overuje sa jeho zhoda so špecifikovanými požiadavkami.

V tímovom projekte sa testuje hlavne konečný systém. Ja si myslím, že náročnosť vyvíjaného systému v tímovom projekte nie je až taká, aby systém potreboval niekoľko desiatok komponentov. Preto prvé dva druhy testovania nie sú až také potrebné. Počet

komponentov je menej početný a ich funkcionalita a vzájomná komunikácia sa zvládnu otestovať aj pri konečnom testovaní celého systému.

Pri testovaní sa ďalej dajú kontrolovať funkcionálne a nefunkcionálne vlastnosti. Podľa [1] sú základné druhy tohto testovania nasledujúce:

- Akceptačné testovanie – sleduje sa správanie systému a porovnáva sa s požiadavkami zákazníka.
- Alfa testovanie – testujú pracovníci z organizácie, ktorá tento softvér vyvinula, a podávajú správy o funkčnosti.
- Beta testovanie – veľmi podobné alfa testovaniu s rozdielom, že testuje široká skupina ľudí spravidla nie z organizácie, ktorá systém vyvinula.
- Regresné testovanie – opätovné testovanie systému po zmene nejakého komponentu, prípadne po odstránení nejakej komplexnejšej chyby.
- Výkonnostné testovanie – testovanie na overenie výkonnostných požiadaviek
- Záťažové testovanie – testovanie systému pri jeho maximálnej záťaži a následné pozorovanie jeho správania.

V tímovom projekte sa vyžíva hlavne akceptačné testovanie. Regresné testovanie sa využíva len zriedka, napríklad pri odstránení veľmi závažnej chyby v implementácii, ktorá je naviazaná na ďalšie funkcie.

Keďže testovanie je vykonávané za účelom nájdenia chýb, je potrebné pre úplnosť uviesť techniky na ich odhaľovanie. Techniky sú založené na mnohých faktoroch ako sú skúsenosti programátorov a vývojárov, štruktúra kódu, špecifikácia, oblasti použitia softvéru a mnohé ďalšie. Podľa [3] sú to dve základné metódy.

- Metóda čiernej skrinky
- Metóda bielej skrinky

Metóda čiernej skrinky je [3] testovacia metóda, v ktorej sú testovacie dáta odvodené od špecifických funkcionálnych požiadaviek bez ohľadu na finálnu štruktúru systému. Pri testovaní metódou čierna skrinka sú viditeľné iba vstupy, výstupy a špecifikácia softvéru. Na základe špecifikácie sa odvodia zo vstupov výstupy a následne sa skontroluje, či aj vytvorený systém dáva rovnaké výstupy ako boli očakávané. Treba zdôrazniť, že implementačná časť je nepodstatná a neberie sa pri tejto metóde vôbec do úvahy.

Metóda bielej skrinky je metóda, pri ktorej sa berie do úvahy implementácia konkrétneho systému, použitá logika, programovací jazyk a vôbec všetky aspekty, ktoré boli v metóde čiernej skrinky ukryté. Testovacie dáta sa vyberajú na základe preskúmania implementácie a celkovej štruktúry systému. Je teda potrebná znalosť softvérového návrhu a aj napísaného zdrojového kódu.

V tímovom projekte sa využívajú obe metódy. Metódou čierna skrinka budú testovať manažéri a ľudia, ktorí neboli pri písaní zdrojového kódu. Zabezpečí sa tak nezaujatosť, ktorá by pri programovaní určite vznikla. Naopak metódou biela skrinka budú testovať ľudia, ktorí poznajú zdrojový kód a teda majú v ňom prehľad. Vedia na

aké chyby sa majú sústrediť, pretože pri programovaní museli riešiť rôzne implementačné problémy a teda vedia odhadnúť výskyt možnej chyby.

Záver

V dnešnej dobe sa každý výrobca softvérových produktov snaží vytvárať čo najkvalitnejšie produkty. Čím kvalitnejší softvér vytvorí, tým si zlepšuje svoje dobré meno v tejto oblasti. Aby bolo možné vytvoriť kvalitný softvér je nevyhnutné použiť testovanie. Tento proces je pri určovaní kvality veľmi rozhodujúci, pretože na základe jeho výsledkov sa hodnotí a opravuje vyvíjaný softvér. Testovanie sa využíva pri každom vytváraní softvérového systému, Výnimkou nie je ani tímový projekt kde sa ako výstup očakáva taktiež kvalitný softvér a jedinou možnosťou ako ho vytvoriť je dôsledné testovanie. Spomenul som základné techniky a druhy testovania aké sa využívajú pri vytváraní softvérového produktu. Zameral som sa hlavne na tie, ktoré sa najviac využívajú aj pri prácach na tímovom projekte. Testovanie je veda ale bez striktných pravidiel. Každý má právo si vybrať druh testovania a na čo sa sústrediť pri testovaní najviac, ale najpodstatnejšie je nepodceňovať testovanie. A podľa mňa mnohokrát ušetrí viac ako len peniaze.

Použitá literatúra

1. Gelperin, D., Hetzel, B.: *The Growth of Software Testing*. CACM Vol. 6, 31 (1988).
2. Chillarege, R.: *Software Testing Best Practices*. Center of Software Engineering, 1999.
3. Pan, J.: *Sofware Testing*. Carnegie Melon university,1999.
4. Rico, D.F.: *IEEE12207 Software Life Cycle*, 2003.

Annotation

Testing in the project team

In creating a software product, the development team seeks to create quality software. Quality has become the main criterion when assessing software. However, to create a quality product at the very beginning it is extremely difficult, even almost unrealistic. Therefore, to maximize the quality testing is used. Testing reveal errors, which if deployed in service could cause problems, and their removal would be very costly, but also are testing to determine the possible objections or observations by the customer. The aim of this essay is to briefly bring the reader to the attributes of quality evaluation, the importance of testing, and in particular to highlight and explain the methods that are best suited for use in a team project.