Zabezpečenie kvality softvéru a testovanie? O čom sa to tu bavíme?

GABRIEL PÁN

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava
pan[.]gabriel[zavináč]gmail[.]com

Abstrakt. Typické vývojové prostredie má určité charakteristiky, ktoré môžu negatívne ovplyvniť kvalitu softvéru. Vývojári majú väčšinou rôznu úroveň svojich profesionálnych schopností, zákazníci majú často nie príliš presne definované požiadavky, pričom ich zaujíma najmä výstup softvéru, a taktiež meniace sa skupiny ľudí pracujúcich na projekte prispievajú ku zníženiu kvality. Zabezpečenie kvality softvéru ako proces sú všetky aktivity vynaložené za účelom dosiahnutia kvality softvéru. Zahŕňa snahu predísť možným rizikám a dokončiť projekt presne podľa vopred danej špecifikácie a štandardov. Testovanie znižuje riziko, že vytváraný systém nebude v niektorých prípadoch fungovať správne a bezproblémovo, aj keď úplnú bezchybnosť zaručiť nedokáže. Pri procese testovania sa kriticky pristupuje ku softvéru a hľadajú v ňom chyby alebo sa porovnáva stav a správanie softvérového produktu oproti špecifikácii. Pri vývoji softvéru sa na testovanie vynakladá značné množstvo úsilia. Je veľmi dôležité, aký spôsob testovania v ktorej fáze vývoja použiť a koľko času stráviť testovaním. Preto sa v tomto článku snažím priblížiť niektoré prístupy k zabezpečeniu kvality a testovaniu.

Úvod

Zabezpečenie kvality (Quality Assurance - QA) nie je len testovanie alebo analýza. Aj keď dokáže byť tento proces nudný, ťažký a zdĺhavý je bezpochyby nevyhnutný. Zabezpečenie, že systém bude pracovať po doručení zákazníkovi vyžaduje mnoho plánovania a disciplíny. Presvedčiť druhých o tom, že aplikácia bude fungovať správne, vyžaduje ešte väčšie úsilie. Kvalitu musíme zabezpečovať počas celého času trvania projektu, nielen „zbúchať“ niečo na konci.

Čo patrí do zabezpečenia kvality? Samozrejme testovanie je v tomto procese kľúčovou aktivitou. Slovo proces by som chcel zvýrazniť lebo je jedným z hlavných aspektov QA. Pomocou QA dodávame produktu určitú hodnovernosť. Zabezpečujeme, že produkt bude fungovať správne a ľudia by mali veriť, že bude fungovať správne. [3]

Testovanie je jedným z hlavných nástrojov QA. Testovanie softvéru slúži na pomoc pri zisťovaní správnosti, úplnosti, bezpečnosti a kvality vyvinutého

počítačového softvéru. Má za úlohu overiť kvalitu softvéru v rámci kontextu, v ktorom má softvér slúžiť. Toto zahŕňa napríklad spúšťanie aplikácií so zámerom nájsť chyby. Musíme mať ale na zreteli, že testovanie nikdy nemôže dokonale zaručiť správnosť aplikácie. Ponúka nám možnosť porovnať softvér s tým, čo máme dané v špecifikácii a čo teda musí byť splnené. Od zabezpečenia kvality tento proces odlišuje fakt, že zabezpečenie kvality zahŕňa všetky business procesy pri tvorbe softvéru, nie len testovanie. [2]

Ako striktne musíme postupovať pri zabezpečovaní kvality?

V ideálnom svete by bola normou perfektná aplikácia. V skutočnom svete však musíme robiť kompromisy. Aj keď niektorí ľudia tvrdia, že kvalita je zadarmo, v skutočnosti to platí len málokedy. Po množstve vynaloženého úsilia a všemožných zlyhaniach možno nájsť proces, ktorý konečne prináša vysokú kvalitu, spoľahlivosť a výkonnosť. Kým sa ale dostaneme na takú úroveň, stojí to množstvo prostriedkov.

Existuje veľa požiadaviek, ktoré treba zahrnúť do QA. Niektoré z nich zahŕňajú splnenie základných funkčných požiadaviek – napríklad aby program robil to čo má pri očakávaných alebo neočakávaných vstupoch. Iné súvisia s výkonnosťou ako priepustnosť, odozva, spoľahlivosť a dostupnosť. Ďalšie môžu súvisieť s operačným systémom. Ak reálni používatelia budú mať nižšiu znalosť systému alebo schopnosť opraviť problémy, musí sa overiť na takýchto používateľoch.

V niektorých prípadoch je škoda spôsobená zlyhaním softvéru taká vysoká, že je prípustné oddialiť jeho odovzdanie zákazníkom, pokým všetky mysliteľné testy nie sú vykonané. V iných prípadoch sú prípustné zmeny a opravy aj počas samotného behu systému. Aj v iných sférach života sú v rôznych situáciách kladené rôzne požiadavky. Napríklad banka potrebuje iné záruky od klienta, ktorý si požičiava pár tisíc korún a iné od klienta ktorý si požičiava niekoľko miliónov. Najvyššiu záruku vyžadujú systémy vytvárané pre letectvo a telekomunikácie. Nestáva sa často, že by vypadli telefónne ústredne. [3]

Čo z toho vyplýva pre organizácie

Keďže zabezpečovanie kvality je proces, je normálne očakávať, že sa ním budú zaoberať zvláštne skupiny ľudí alebo celé organizácie. V jednoduchých, ľahko pochopiteľných aplikáciách, sa môže dizajnér starať aj o procesy súvisiace s QA rovnako, ako to robia pri tradičnom debugovaní alebo unit testovaní. Bohužiaľ ľudia často neradi míňajú veľa času na úlohy týkajúce sa QA. Vytváranie novej funkcionality je oveľa zábavnejší proces. Rovnako, ak niekto prehliadne určitú dôležitú vec pri analýze, zrejme tomu nebude inak ani neskôr pri procese testovania. [3]

Preto vo väčších organizáciách pre produkty s vyššími nárokmi sa o zabezpečenie kvality starajú špecializované tímy ľudí. Je ideálne ak sú tieto skupiny oddelené od tímu vývojárov a majú právomoc na príkaz prerobiť nefunkčnú časť aplikácie ak je to

potrebné. Títo nezávislí testeri sú väčšinou zodpovední za definovanie postupu pri QA a prihliadajú na jeho dodržiavanie.

Organizácia, ktorá sa zaoberá zabezpečovaním kvality, nemusí byť veľká aby bola efektívna. Relatívne malé skupiny dokážu dobre robiť svoju robotu, pokým sú nezávislé, majú znalosti a dobre rozumejú produktu. Potrebujú byť taktiež oboznámení s rôznymi možnosťami, ako je možné produkt zneužiť, obísť jeho bezpečnostné mechanizmy alebo iným spôsobom sa do neho nabúrať. Prideľovanie takýchto úloh neskúseným členom týmu môže ohroziť celý produkt a ľudí čo na ňom pracujú. [3]

Testovanie v praxi

Čitateľ tohto článku sa už zrejme stretol s pojmami testovanie metódou čiernej alebo bielej skrinky. Sú to pojmy označujúce spôsob, akým sa pozeráme na testovanie softvéru. Metóda čiernej skrinky pozerá na produkt zvonka, berie do úvahy výstupy produktu pri určitých vstupoch nehľadiac na vnútornú stavbu softvéru. Biela skrinka naopak prihliada aj na vnútornú stavbu softvéru.

Testovanie je z časti intuitívnym ale veľmi systematickým procesom. Dobré testovanie je viac ako len spúšťanie programu aby sme videli či pracuje správne. Hlboká analýza testovaného programu podporená širokými znalosťami o dostupných testovacích postupoch a nástrojoch sú základom pre systematické testovanie. [2]

Rozoznávame dve metódy testovania:

- Manuálne
- Automatické

Manuálne testovanie, ako samotný názov napovedá, je proces, pri ktorom jednotlivec alebo skupina jednotlivcov ručne testujú softvér. Toto môže znamenať orientáciu v používateľských rozhraniach, zadávanie rôznych informácií alebo dokonca pokusy o nabúranie sa do softvéru alebo databázy s ktorou pracuje. Je jasné, že takéto testovanie je náročné na čas a pomalé. Hodí sa najmä na:

- Testovanie používateľských rozhraní a použiteľnosti produktu
- Testovanie pri, ktorom testeri nepostupujú podľa vopred daného skriptu ale skôr skúmajú aplikáciu a používajú svoj inštinkt na nájdenie možných chýb
- Testovanie častí aplikácie, ktoré podliehajú častým zmenám

Jednou z hlavných nevýhod tejto metódy je jej náročnosť na čas. Čas potrebný na pretestovanie väčšej aplikácie sa môže pohybovať v týždňoch až mesiacoch. Ďalej, výsledky testovania často závisia od samotnej osoby, ktorá testovanie vykonáva. [1]

Automatické testovanie je proces vytvárania testovacích skriptov, ktoré potom môžu byť spustené automaticky v mnohých iteráciách. Ak sa správne použije, dokáže znížiť variabilitu výsledkov, zrýchliť testovací proces, zvýšiť počet testovaných častí a ponúka vierohodnejšie výsledky. Táto metóda sa však nehodí v niektorých prípadoch kedy:

- Testujeme finálnu použiteľnosť koncovým používateľom

- Nepotrebujeme niečo testovať veľakrát dokola, lebo výhodnosť automatického testovania sa ukazuje najmä pri opakovaných testoch
- Testujeme časti softvéru, ktoré podliehajú častým zmenám a vyžadujú pri tom úpravu aj testovacích skriptov

Z vlastnej skúsenosti môžem potvrdiť, že ani u nás sa na testovanie nezabúda. Každá časť aplikácie, ktorá má ponúkať určitú funkcionalitu, je dôkladne otestovaná. Pri testovaní rozhraní webových aplikácií (internetové stránky) sa využívajú ako živí testeri, tak aj automatizované systémy, ktoré však nie sú použiteľné vo všetkých prípadoch. Niektoré aplikácie majú zabudované mechanizmy, ktoré doslova cielene znemožňujú aby nejaký automatizovaný systém bol schopný pracovať s aplikáciou. V týchto prípadoch stále treba živých testerov. [1]

V ďalšej časti sa budem venovať jednému so spôsobov testovania prístupom biela skrinka. Podľa môjho názoru ide o jeden najefektívnejších spôsobov testovania, ktorý je vhodný pri tvorbe väčších aplikácii zložených z viacerých častí.

Unit testing

V texte budem používať tento anglický názov, ktorý označuje testovanie určitej samostatnej jednotky alebo modulu. Technika takéhoto testovania sa používa, keď máme väčší systém zložený z viacerých menších častí, ktoré zabezpečujú určitú funkcionalitu pre systém ako celok, ale nie sú s ním pevne zviazané a dajú sa ľahko nahradiť novšími, alebo inak stavanými modulmi ponúkajúcimi rovnakú funkcionalitu.

Takýto prístup je typický pre paradigmu objektovo orientovaného programovania, kde modulmi – unitmi môžu byť skupiny objektov pracujúcich, aby poskytli spolu určitú funkcionalitu, ba dokonca nimi môžu byť aj samostatné objekty. Od stupňa granularity akú zvolíme potom záleží množstvo a zložitosť vytváraných testov. Jednotlivé unity testujeme vždy ako samostatný celok.

Nato, aby mohli byť tieto jednotky samostatne otestované, je treba často veľké množstvo podporného kódu. Tento kód sa dá prirovnať ku lešeniu, ktoré používajú robotníci pri stavbe budov. Najprv sa postaví lešenie budovy a až keď sa stavba dokončí, lešenie sa rozoberie a odkryje samotnú budovu. Rovnako pri testovaní treba jednotlivým unitom vytvoriť prostredie, v ktorom môžu byť testované. Toto prostredie sa vytvára vždy pre potreby daného modulu a často má inú kvalitu ako samotný modul. Môže poskytovať napríklad prístup do falošnej databázy alebo poskytovať modulu vstupné testovacie údaje iným spôsobom. Rovnako môže poskytovať služby a funkcie ktoré modul volá a potrebuje ich pre svoju činnosť. Z tohto jasne vyplýva, že vytvorenie vhodného prostredia nie je vždy triviálne a často trvá pomerne dlho. Navyše pre iný typ testu treba zase iné prostredie a podobne. [3]

V praxi som zažil prednášku, na ktorej mi bol prezentovaný názor, že všetko by sa malo testovať na všetkých úrovniach. Znamená to, že by mali byť písané testy pre jednotlivé triedy, ktoré by testovali čisto len ich funkcionalitu. Ďalej na ďalšej úrovni by boli testy, ktoré by testovali funkcionalitu, ktorú ponúka viacero poprepájaných tried. Tieto spolu tvoria celok, aký je možné pokladať za samostatnú jednotku. Takéto jednotky pospájané dokopy potom tvoria ďalší už komplexnejší komponent na

nasledujúcej úrovni, ktorý by sa mal zase samostatne pretestovať (napríklad modul zodpovedný za komunikáciu aplikácie s databázou).

Podľa môjho názoru je takýto podrobný prístup k testovaniu síce na jednej strane dôkladný ale na druhej strane vyžaduje obrovské množstvo testov, ktoré ako som uviedol vyššie, vyžadujú nemalé množstvo podporného kódu, ktorý vôbec nepatrí ku samotnej aplikácii. Myslím si, že množstvo testov, ktoré je potrebné spraviť, by malo záležať od povahy projektu a malo by byť ponechané tak trošku aj na intuíciu. Tá zohráva pri testovaní podstatnú rolu. Väčšie moduly musia byť samozrejme dôkladne otestované a tie viac komplikované by mali byť pretestované aj po častiach. Pre moduly, ktoré očividne pracujú správne, lebo nijak inak pracovať ani nemôžu, by podľa môjho názoru nemuselo byť vytvárané toľko testov len kvôli tomu, aby spĺňali určitý formálny predpis. To samozrejme opäť závisí od povahy produktu.

Dôležité je nájsť dobrú strednú cestu tak, aby sme čo najviac znížili riziko, že neodhalíme nejakú chybu ale na druhej strane aby sme nemíňali čas, ktorého nikdy nie je dosť, na písanie zbytočne podrobných testov.

Záver

Proces zabezpečenia kvality softvéru nie je triviálna záležitosť. Detaily tohto procesu závisia od organizácie, ľudí a účelu produktu. Môže byť náročný, zdĺhavý a drahý ale je nevyhnutný. Je veľmi dôležité prispôsobiť ho potrebám daného projektu, aby dostatočne pokrýval všetky jeho potreby ale na druhej strane, aby zbytočne nezvýšil náklady na čas alebo ľudí. Zabezpečenie softvéru a testovanie treba robiť tak, aby sme minimalizovali to bolestivé a maximalizovali jeho prínos.

Použitá literatúra

1. Rob Pirozzi, LogiGear Corporation: *Introduction to Software Testing* http://www.logigear.com/newsletter/introduction_to_software_testing.asp (17.10.2008)
2. Software testing: http://www.testingbrain.com (17.10.2008)
3. Stuart Feldman, IBM research: *Quality Assurance: Much More than Testing* (2005)

Annotation

Software quality assurance and testing

The typical production environment has certain characteristics which may negatively affect software quality. The developers have widely varying level of skill. Customers have poorly defined but often complex objectives and are usually interested only in the software output. Also the frequent changes of the people in the team contribute to the decrease of software quality. Quality assurance as a process involves all activities aimed to assure software quality.

That means careful attention to the specification of the requirements to be satisfied, to ensure they accurately capture what is wanted or intended, and the formulation of the test cases that can be used to demonstrate their eventual satisfaction in code. Testing lowers the probability that the created software won`t work in some cases as desired. It cannot guarantee us flawless software though. A lot of money is spent on testing during the software development. It is crucial to choose the proper testing method and reserve the adequate amount of time in each phase of the development process. In this paper I would like to introduce some of the approaches to testing and quality assurance.