Ako zvýšiť kvalitu softvéru cez automatizáciu testovania?

MARTIN PAŠMÍK

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

Abstrakt. Zložitosť softvérových systémov sa stále zvyšuje a tým narastá aj náročnosť udržiavania ich kvality. Jedným z najdôležitejších prvkov kvality softvéru je jeho správnosť, ktorá sa zvyčajne overuje prostredníctvom testovania. Testovanie je rozsiahla oblasť obsahujúca špecifické aj všeobecnejšie problémy, ktoré je potrebné riešiť pri každom projekte. Celkovo, pri vývoji softvéru chceme byť čo najefektívnejší, a aj preto by sme sa mali sústrediť na zefektívnenie testovacieho procesu. K tomu môže viesť práve automatizácia testovania. V tejto eseji sa teda venujem otázkam a problémom týkajúcich sa automatizovaného testovania, aké má požiadavky, výhody, či nevýhody a prečo je podľa môjho názoru dôležitou oblasťou, na ktorú by sme pri tvorbe softvéru nemali zabúdať.

Úvod

So zvyšujúcou sa komplexnosťou softvérových systémov sa zvyšujú aj nároky na zabezpečenie ich kvality. Kvalita je dôležitým ukazovateľom, ktorý by mal v čo najvyššej miere zohľadňovať každý návrhár a tvorca softvéru, ak chce aby výsledok bol úspešný. Kvalita výsledného softvérového produktu závisí od množstva faktorov, avšak medzi tie najdôležitejšie patria správnosť, použiteľnosť, spoľahlivosť, či bezpečnosť, ktoré sa overuje prostredníctvom testovania.

Testovanie dokáže zabrať množstvo času a prostriedkov, množstvo testov sa totiž opakuje po každej úprave softvéru a tento čas by sme mohli použiť na iné úlohy. Takisto je problémom ak sa začína testovať príliš neskoro a na odstránenie zistených problémov je potrebné vynaložiť väčšie úsilie ako keby sa odhalili skôr. Otázka teda znie, ako môže k riešeniu takýchto problémov prispieť práve automatizácia testovania?

Manažment projektov softvérových a informačných systémov, október 2008, s. 1-7.

Kvalita softvéru a testovanie

Kvalitou softvéru sa zvyčajne zaoberá manažment kvality, ktorý pokrýva celý v životný cyklus. Úlohou tohto manažmentu je zabezpečiť čo najkvalitnejší produkt s dodržaním termínov a rozpočtu. Podrobnejšie sa procesmi zabezpečenia kvality zaoberajú normy ISO 9000/ISO 14000.

V dnešnej dobe sa väčšina softvéru vyrába v spolupráci množstva ľudí, výslednému produktu môže ťažko kompletne porozumieť len jedna osoba, a aj preto je dôležité aby tento manažment pracoval efektívne. Zabezpečenie kvality softvérového produktu sa samozrejme týka nielen softvéru ale aj ľudí, ktorý ho vyvíjajú a ich organizácie, podrobnejšie sa však budem venovať práve testovaniu. Gill medzi postupy na zlepšenie softvéru zaraďuje [2]:

- Definovanie požiadaviek
- Predchádzanie chybám
- Detekcia chýb
- Odstránenie chýb

Pre testovanie sú dôležité hlavne dobre definované požiadavky, na základe ktorých sa vytvárajú testy a účelom testovania je práve detekcia chýb. Už notoricky známy je citát Dijkstru: „Testovanie nemôže preukázať, že v programe chyby nie sú. Môže iba ukázať, že tam chyby sú“. To je podľa môjho názoru spôsobené tým, že kvôli efektivite sa testy sústredia len na kritické a dôležité súčasti systému a takisto aj testy sú len softvérom a môžu obsahovať chyby.

Existuje viacero typov testovania ako biela a čierna skrinka ako aj metodík, či spôsobov návrhu a prípravy testov. Takisto by sme mali rozlišovať aspoň testy modulov (unit tests), testy použiteľnosti, integračné, systémové, akceptačné a záťažové testy. Množstvo agilných metodík využívajú skoré testovanie na zabezpečenie kvality produktov, medzi najznámejšie asi patrí testami riadený vývoj (test-driven development), kde sa testy tvoria pred samotným začatím písania zdrojového kódu aplikácie. Zaujímavá bola podľa mňa informácia, že z ceny vývoja produktu testovanie môže dosahovať aj 50 % celkových nákladov[3], tým sa len potreba jeho efektívnejšieho vykonávania zvyšuje.

V štúdii testovania [4] Martin a kolektív ukázali, že v praxi, v malej skupine pracujúcej na projekte sa testovaniu venuje minimálny možný čas, sústredený len na skúšanie dôležitej a najčastejšie používanej funkcionality s tým, že ostatné chyby sa opravujú až neskôr, niekedy až po odhalení zákazníkom. Tento prístup má síce logický základ a to ušetriť prácu v aktuálnom čase, čo môže niekedy byť nevyhnutné kvôli nedostatku času, ale podľa môjho názoru to určite nie je správne. V budúcnosti sa totiž bude musieť opravám venovať viac času ako sa momentálne ušetrí. Softvér, ktorý by neobsahoval žiadne chyby hneď po vytvorení ani po doplnení alebo prepracovaní ďalšej funkcionality je utópiou.

Automatizované testovanie

Čo si máme pod týmto pojmom vôbec predstaviť? Existuje viacero pohľadov, ja by som činnosti spadajúce pod tento pojem rozdelil takto:

- Automatické vykonávanie testov
- Automatické generovanie jednotlivých testovacích prípadov
- Automatická validácia a vyhodnotenie výsledkov testov

Každá z týchto častí môže byť plne automatizovaná, ale v niektorých prípadoch môže byť dôležitá aj čiastočná automatizácia alebo podpora príslušných procesov. Generovanie testov je skoro vždy spojené aj s ich automatickým vykonávaním, a to zas býva veľmi často spojené s automatickým vyhodnocovaním. Generovanie testov je samostatná a zložitá disciplína, používa sa napríklad pri testovaní založenom na modeloch (model-based testing), k tomu je však potrebné vytvárať systém s vhodnou architektúrou. Ďalšou možnosťou je použitie softvéru, ktorý daný zdrojový kód zanalyzuje a na jeho základe vytvorí jednotlivé moduly a ich testy. Chakrabarti a Godefroid [3] opísali akým spôsobom by sa mal takýto systém správať. Ich návrh je zložený na formálnom opise prostredníctvom množín a grafov. Dôležitým prvkom v tomto prípade je realistickosť vygenerovaných vstupov, ktoré sa tvoria aj na základe prepojenia jednotlivých funkcií a ich rozhraní. Myslím si však, že väčšie možnosti uplatniť sa má skôr strojmi vykonávané testovanie, ktoré sa dá využiť vo viacerých prípadoch a tiež, že je vhodnejšie dopredu navrhnúť vhodnú architektúra s dobrou testovateľnosťou.

Použitím automatizovaného testovania v praxi sa zaoberali Berner, Weber a Keller. Vo svojej štúdii aplikácie automatizovaného testovania [1] na rôznych projektoch dospeli k výsledkom zobrazeným v Tab. 1, v ďalšom texte sa budem ich záverom venovať podrobnejšie.

Tab. 1. Tabuľka s výsledkami pozorovaní [1]

	Stratégia automatizácie testov je často nevhodná	Testy sú opakované častejšie ako sa očakáva	Možnosť spúšťať automatizované testy sa znižuje ak sa nepoužívajú	Automatizované testovanie nemôže nahradiť manuálne testovanie	Testovateľnosť je často zabúdanou nefunkcionálnou požiadavkou	Udržiavanie „testvéru“ je ťažké
Projekt A	Pozorované značne	Pozorované		Pozorované značne	Pozorované značne	
Projekt B		Pozorované	Pozorované značne	Pozorované značne	Pozorované značne	Pozorované
Projekt C	Pozorované značne	Pozorované značne	Pozorované značne	Pozorované značne	Pozorované	Nepozorované
Projekt	Pozorovaný	Pozorované		Pozorované	Pozorovaný	Pozorované

D	opak				opak	značne
Projekt E	Pozorované značne	Pozorované		Pozorované	Pozorované značne	Pozorované

Stratégia automatizácie testov je často nevhodná

To podľa môjho názoru súvisí s očakávaniami, že automatizované testovanie dokáže viac ako neautomatizované, lenže realita je opačná a testy ostávajú rovnako kvalitné, prípadne sa môžu vyskytnúť ešte ďalšie chyby pri ich automatizácii. Cieľom automatizácie je skôr odbremeniť ľudí, ktorí by trávili čas opakovaným vykonávaním tých istých testov, po každej zmene vo vnútri softvéru. Samozrejme úsilie vynaložené na automatizáciu testovania sa vráti až po viacerých vykonaniach testov. Následne sa môžu ľudia venovať zlepšovaniu kvality alebo množstva testov, respektíve inej práci súvisiacej s projektom.

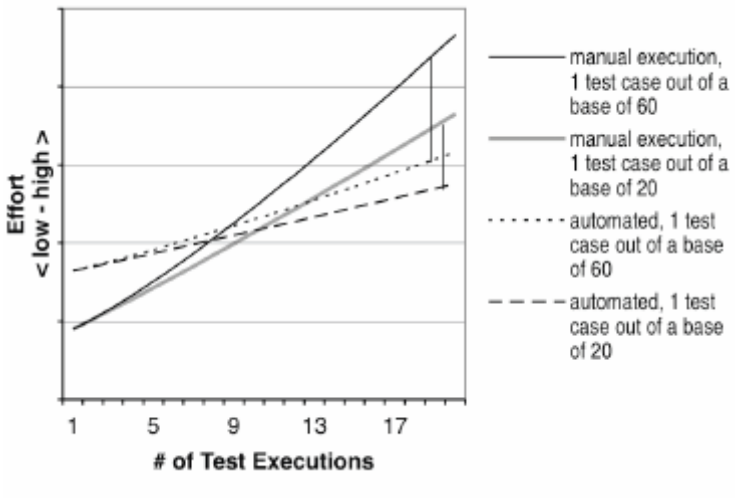
Dnes je už možné automatizovať aj testy týkajúce sa grafického rozhrania, avšak zaoberať sa tým by sme sa mali až keď má rozhranie ustálenú podobu, lebo zmeny dizajnu a presun jednotlivých ovládacích prvkov môžu testy znefunkčniť. Tiež si myslím, že výstupy takýchto automatizovaných testov sú väčšinou ťažko hodnotiteľné strojovo a musí sa im venovať človek.

Nemalo by sa zabúdať ani na stále kvalitnejšie nástroje, v tomto čase zabudované do každého pokročilejšieho vývojového prostredia, ktoré môžu testovanie uľahčiť.

Testy sú opakované častejšie ako sa očakáva

S týmto záverom sa stotožňujem a mám podobnú skúsenosť získanú počas implementačnej časti bakalárskeho projektu, kde som po doplnení každej zložitejšej funkcionality musel znova testovať aj základnú činnosť aplikácie, čím som sa oberal o čas. K názoru, že by bolo vhodné jednotlivé testy automatizovať som dospel až vo veľmi neskorom štádiu, kedy by sa úsilie vynaložené na tento úkon nenavrátilo.

Samozrejme, že testy, ktoré sa budú opakovane vykonávať veľakrát (pozri Obr. 1) sú vhodnými kandidátmi na automatizáciu. V prípade techník ako sú testami riadený vývoj alebo priebežné testovanie (continuous testing) v extrémnom programovaní je podľa mňa automatizácia testovania nutnosťou vždy.



Obr. 1. Pomer potrebného úsilia k počtu vykonávaní testov[1]

Možnosť spúšťať automatizované testy sa znižuje ak sa nepoužívajú

K tomuto by som dodal len toľko, že tak ako sa mení zdrojový kód a vlastnosti softvéru počas vývoja, treba udržiavať aj testy, tie by mali byť takisto prehľadné a opísané, aby sa nestali nezrozumiteľnými a tým pádom pri úprave testovanej oblasti nepoužiteľnými. Automatizované testy sú predsa takisto len softvérom a prenášajú sa na nich všetky jeho možné nedostatky.

Automatizované testovanie nemôže nahradiť manuálne testovanie

Bolo pozorované, že väčšina chýb sa objaví práve pri tvorbe testov a nie až pri ich spustení. Automaticky opakované testy už ťažko odhalia zásadné chyby, tie sa podľa mňa môžu vyskytnúť až pri veľkých zásahoch do štruktúry programu alebo nepozornosťou programátora. Skúsení programátori testov tiež môžu využívať poznatky o tvorcoch testovanej súčiastky a skúsenosti z predchádzajúcich projektov. Tieto vlastnosti sa len v malej miere dajú preniesť do automatizovaných testov. To je síce pravda, ale podľa môjho názoru je určite vhodnejšie keď majú títo ľudia viac času na analýzu, návrh a implementáciu testov, čiže zvyšuje sa kvalita testov, prípadne sa môže zvyšovať ich kvantita a otestovaných bude viacero súčastí systému. Samozrejme si nemyslím, že všetky úlohy spojené s testovaním sa dajú automatizovať, napríklad testy nefunkcionálnych požiadaviek ako sú použiteľnosť, internacionalizácia a lokalizácia alebo čiastočne aj bezpečnosť softvéru sa dajú vykonávať a vyhodnocovať počítačom veľmi ťažko.

Testovateľnosť je často zabúdanou nefunkcionálnou požiadavkou

Miera testovateľnosti systému je dôležitým faktorom pri každom testovaní. Táto požiadavka na kvalitu softvéru je často opomínaná, čím sa spôsobuje množstvo

problémov. Spomedzi ďalších identifikovaných požiadaviek na architektúru systému sú podľa mňa najdôležitejšie:

- Neviditeľné alebo nepoužiteľné vrstvenie, ktoré nedovoľuje testovať časti systému samostatne a nezávisle
- Nezrozumiteľné alebo chýbajúce chybové výpisy [1]

Časti systému by teda mali byť vhodne štruktúrované a rozložené. Ak totiž nie je možné vytvoriť dostatočne podrobné testy, automatizáciou ich vykonávania sa nemusíme ani zaoberať. Ak sa plánuje aj automatizácia validovania výsledkov testov, tak aj výpisy a chybové hlášky by mali ideálne byť spracovateľné strojom.

Udržiavanie „testvéru" je ťažké

Za testvér považujeme všetky potrebné časti pre automatizáciu testovania ako sú jednotlivé testy, vstupné a výstupné údaje, či dodatočné nástroje na ich spravovanie. Nezabúdajme, že testy sú stále len softvérom a preto by mali tiež prejsť životným cyklom a mali by byť aj otestované. Testovanie testov by však už malo prebiehať manuálne a v ideálnych prípadoch by sa mala tvorbe testov venovať vyššia pozornosť, aby sa chybám predišlo, ináč by sme sa mohli testami testov testov a tak ďalej bezvýsledne zacykliť. Pri správnom návrhu sa tiež dajú rôzne komponenty testov viacnásobne využiť, keďže testovacie činnosti sú si príbuzné.

Vhodnosť automatizovaného testovania

Automatizovať môžeme rôzne druhy testov, testy modulov, integračné testy aj akceptačné testy, dôležitou otázkou podľa môjho názoru je skôr akú námahu potrebujeme na zautomatizované vkladanie vstupov a následné ohodnotenie výstupov. Ak totiž testujeme jednoduchý matematický program, ktorý má číselné vstupy a k nim vieme zistiť príslušné výstupy, tak automatizácia je v zásade jednoduchá. Horšia je situácia pri systémoch so zložitým grafickým rozhraním, ktorého ovládanie by sme chceli zautomatizovať. To je však veľmi závislé od prípadu k prípadu, od premenlivosti rozhrania a použitých podporných nástrojov.

Niekedy sa však automatizácii testovania nedá vyhnúť. Za takéto prípady považujem napríklad testy zaťaženia systému. Zvyčajne nie je možné vyskúšať správanie sa v takýchto podmienkach, keďže niektoré dnešné systémy zvládajú tisíce aj desaťtisíce požiadaviek súčasne. Preto je nutné vytvoriť komplexný automatizovaný testovací systém, ktorý je schopný takúto záťaž simulovať.

Ďalšou otázkou je vhodnosť pre malé projekty a úlohy, tá podľa môjho názoru súvisí s tým, či sa vôbec nejaké testovanie vykonáva. Napríklad pri školských projektoch, ktoré sa raz vytvoria a odovzdajú je tvorba automatizovaných testov zbytočná, zaujímavá môže byť až pri celosemestrálnych alebo dlhších projektoch ako sú bakalárska či diplomová práca alebo tímový projekt, kedy sa vytvára viacero verzií a prototypov. Pri projektoch, ktoré však plánujeme v budúcnosti udržiavať a vylepšovať je však určite vhodné venovať čas na automatizáciu.

Záver

Testovanie je samozrejmosťou pre každý väčší projekt a myslím si, že je súčasťou vývoja softvéru, ktorej by sme mali venovať rovnakú pozornosť ako samotnej jeho tvorbe. Keďže všetci chceme byť čo najefektívnejší, musíme sa venovať aj optimalizácii testovacieho procesu, ktorý môže byť časovo veľmi náročný. Hlavnou výhodou automatizácie testovania je, že pri dostatočnom opakovaní sa testov šetrí čas človeka, ktorý by musel testy manuálne vykonávať. Tým sa umožňuje venovať sa iným úlohám a vytvoriť kvalitnejší produkt. Samozrejme, že je potrebné dávať dôraz na testovateľnosť systému a venovať čas príprave a údržbe automatizovaných testov, ale z dlhodobého hľadiska je to určite výhodné a niekedy dokonca nevyhnutné. Ďalším krokom vo vývoji by mohlo byť automatizované generovanie testov, ktoré by viedlo k ešte väčšiemu zefektívneniu práce.

Použitá literatúra

1. Berner, S., Weber, R. and Keller, R.K.: Observations and Lessons Learned from Automated Testing. *International Conference on Software Engineering - Proceedings of the 27th international conference on Software engineering*, (2005) 571-579, dostupné cez http://portal.acm.org/citation.cfm?id=1062556 [13-10-2008].
2. Gill, P: Factors affecting effective software quality management revisited. In: *ACM SIGSOFT Software Engineering Notes, Volume 30, Issue 2 (March 2005)*. http://portal.acm.org/citation.cfm?id=1050862 [13-10-2008].
3. Chakrabarti, A., Godefroid, P.,: *Software Partitioning for Effective Automated Unit Testing*. In: *International Conference On Embedded Software - Proceedings of the 6th ACM & IEEE International conference on Embedded software* (2006), dostupné cez http://portal.acm.org/citation.cfm?id=1176925 [13-10-2008].
4. Martin, D., Rooksby J., Rouncefield, M., Sommerville I.: *'Good' Organisational Reasons for 'Bad' Software Testing: An Ethnographic Study of Testing in a Small Software Company*. In: *International Conference on Software Engineering - Proceedings of the 29th international conference on Software Engineering* (2007) 602-611, dostupné cez http://portal.acm.org/citation.cfm?id=1248890 [13-10-2008].

Annotation

How to improve software quality through test automation?

Complexity of software systems is rising and so is the difficulty to keep their good quality. One of the most important parts of software quality is its correctness, which is usually verified through testing. Testing is an area, which has large range containing specific, but also more general problems, which need to be solved by every project. In general when developing

software, we want to be as effective as possible and because of that, we should concentrate on making the testing process more effective. That is where test automation might help. In this essay I am addressing problems and questions about test automation, what are its requirements, advantages or disadvantages and why is it in my opinion important thing on which we should never forget when creating new software.