

Efektivní testování softwaru

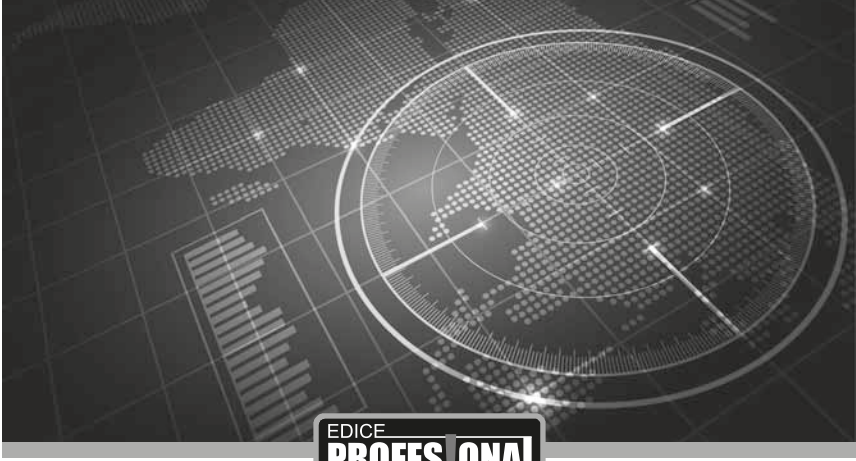
**Klíčové otázky
pro efektivitu
testovacího procesu**

**Miroslav Bureš, Miroslav Renda
Michal Doležel a kolektiv**

EDICE
PROFESIONAL

 **GRADA**[®]

- Organizace a řízení testování softwarových systémů
- Realistický plán testování a prevence možných rizik
- Příprava testovacího prostředí a testovacích dat
- Optimalizace regresních testů
- Statické testování a testování požadavků
- Odhadování testovacích aktivit
- Efektivní využití automatizovaných testů



EDICE
PROFES ONAL



Efektivní testování softwaru

**Klíčové otázky
pro efektivitu
testovacího procesu**

**Miroslav Bureš, Miroslav Renda
Michal Doležel, Peter Svoboda
Zdeněk Grössl, Martin Komárek
Ondřej Macek, Radoslav Mlynář**

Upozornění pro čtenáře a uživatele této knihy

Všechna práva vyhrazena. Žádná část této tištěné či elektronické knihy nesmí být reprodukována a šířena v papírové, elektronické či jiné podobě bez předchozího písemného souhlasu nakladatele. Neoprávněné užití této knihy bude **trestně stíháno**.

**Miroslav Bureš, Miroslav Renda, Michal Doležel, Peter Svoboda,
Zdeněk Grössl, Martin Komárek, Ondřej Macek, Radoslav Mlynář**

Efektivní testování softwaru

Klíčové otázky pro efektivitu testovacího procesu

Vydala Grada Publishing, a.s.
U Průhonu 22, Praha 7
obchod@grada.cz, www.grada.cz
tel.: +420 234 264 401, fax: +420 234 264 400
jako svou 6339. publikaci

Recenzovali:
doc. Ing. Valentino Vranič, Ph.D.
doc. RNDr. Jiří Barnat, Ph.D.
doc. Ing. Branislav Lacko, CSc.




Odpovědný redaktor Petr Somogyi
Sazba Jan Šístek
Fotografie na obálce fotobanka Allphoto
Počet stran 232
První vydání, Praha 2016
Vytiskly Tiskárny Havlíčkův Brod, a. s.

© Grada Publishing, a.s., 2016
Cover Design © Grada Publishing, a. s., 2016

Názvy produktů, firem apod. použité v knize mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

ISBN 978-80-271-9389-9 (pdf)
ISBN 978-80-247-5594-6 (print)

Obsah

Úvod	11
Pro jaké typy projektů a domény se kniha hodí?	11
Jak je kniha strukturovaná.....	11
Komu je kniha určena?	12
Použitá terminologie	12
O autorech.....	12
Poděkování.....	14
 1 Businessová hodnota testování.....	15
Kam mohou vyšplhat náklady za nedostatečnou kvalitu.....	15
Výpočet ceny za kvalitu	17
Prostředky pro argumentaci hodnoty testování z praxe	18
Fáze získání informací	18
Fáze přípravy argumentace	19
Fáze použití argumentace	20
Shrnutí	21
 2 Role testování ve firmě a v projektu	23
Testování neprobíhá ve vzduchoprázdnu.....	23
Faktory ovlivňující průběh a formu životního cyklu.....	26
Metodiky vývoje, testování a provozu.....	27
V model a W model: totéž, jen jinak?	28
Agilní přístup	31
DevOps jako spojení vývoje a provozu.....	33
Jak zapadá oddělení testování do organizace jako takové?	33
Role manažera testování	35
Interpersonální role.....	36
Informační role.....	36
Rozhodovací role.....	37
Shrnutí	38
 3 Strategie testování	39
Trocha chaosu na začátek.....	40
Co má obsahovat hlavní plán testování.....	41
Co má obsahovat detailní plán testování	42
Business Driven Test Management.....	43
Cíle testování.....	44

Mluvte jazykem, kterému investor rozumí.....	45
Přehled částí testovaného systému	46
Co naopak netestovat?.....	49
Určení priorit pro testování.....	49
Pravděpodobnost selhání.....	50
Možné dopady.....	51
Určení produktových rizik.....	51
Úrovně testování a intenzita testování	52
Sestavení projektového plánu	54
Vstupní a výstupní kritéria.....	55
Shrnutí	56



Projektová a produktová rizika.....	57
Pohled na rizika z výšky.....	57
Obecný přístup k řízení rizik	59
Jak řídit projektová rizika	60
Jak identifikovat projektová rizika	60
Jak na analýzu rizik.....	61
Jak s riziky naložit aneb plánování opatření	62
Realizace opatření pro řízení rizika	63
Jak evidovat projektová rizika.....	63
Jak projektová rizika ovlivňují produktová rizika	65
Jak řídit produktová rizika.....	65
Co znamená kvalita.....	65
Příklady neošetřených produktových rizik.....	66
Identifikace a analýza produktových rizik	66
Významné kvalitativní charakteristiky webových aplikací	68
Testování založené na rizicích (risk-based testing).....	71
Techniky pro testování založené na rizicích.....	71
Testování založené na rizicích v praxi	71
Prioritizace s ohledem na rizika	73
Reporting při testování založeném na rizicích	74
Shrnutí	75



Odhadování pracnosti testovacích aktivit.....	77
Co po nás chce management.....	77
Jaká je realita v odhadování – reálná přesnost.....	78
V čem je problém?	79
Kognitivní klamy	79
Metodické chyby	82
Rychlé tipy jak zpřesnit odhady	85
Základní metody pro odhadování testů	86
Odhadování pomocí podílu z jiné aktivity	86
Velikost testovaného objektu	87
Hierarchický rozpad práce (Work Breakdown Structure).....	88
Odhad na základě analogie	88

Planning poker.....	89
Optimistický, střední, pesimistický scénář (tříbodový odhad).....	90
Extrapolace na základě předchozího průběhu.....	90
Kombinované metody.....	91
Jak vybrat vhodnou metodu?.....	92
Jak kontrolovat odhady.....	92
Vytvořte a zpřesňujte vlastní systém pro odhadování.....	93
Výchozí nastavení odhadovací metody.....	93
Jaká data budeme potřebovat?.....	93
Zpřesňování odhadovací metody.....	94
Shrnutí.....	95



6 Statické testování.....	97
Testování dokumentace.....	97
Testování obsahu dokumentů.....	98
Testování formální stránky dokumentu.....	98
Metody testování dokumentace.....	98
Statické testování kódu.....	99
Co je kvalitní kód.....	100
Metriky kvality kódu.....	101
Formální stránka kódu – kódovací standardy.....	102
Metody statického testování kódu.....	102
Začlenění statického testování do procesu vývoje.....	103
Ověření přínosů statického testování v praxi.....	104
Shrnutí.....	105



7 Ověřování požadavků.....	107
Co je požadavek?.....	107
Úrovně požadavků.....	108
Kategorizace požadavků.....	109
Forma zachycení požadavků.....	110
Textové šablony požadavků.....	111
Atributy požadavků.....	112
Ověřování specifikace požadavků.....	112
Ověřování formálních náležitostí specifikace.....	112
Testování obsahové stránky specifikace.....	113
Způsoby ověřování požadavků.....	114
Shrnutí.....	114



8 Testovací prostředí.....	115
Co je testovací prostředí?.....	116
Životní cyklus testovacích prostředí.....	116
Kdy začít s plánováním testovacích prostředí?.....	116
Fáze životního cyklu testovacích prostředí.....	117
Stěžejní procesy podporující životní cyklus testovacích prostředí.....	118
Organizační zajištění životního cyklu testovacích prostředí.....	120

Úvod do strategie řízení testovacího prostředí.....	122
Typy testovacích prostředí.....	123
Obecné atributy testovacího prostředí	123
Pískoviště	125
Vývojové prostředí (DEV).....	125
Testovací prostředí pro systémové testy (SYS).....	125
Testovací prostředí pro integrační testy (INT).....	125
Testovací prostředí pro podporu produkce (PRS).....	126
Předprodukční testovací prostředí (PRE).....	126
Produkční a záložní testovací prostředí (PROD).....	126
Školící prostředí (EDU).....	127
Jak to při vývoji a provozu IT systému funguje	127
Využití testovacích prostředí v čase	129
Modelový případ pro analýzu využití prostředí v čase	129
Aktivity na prostředích	130
Způsob práce zvaný DevOps	131
Shrnutí	133



Příprava a správa testovacích dat.....	135
Proč jsou dobrá testovací data podstatná.....	135
Negativní jevy s testovacími daty, které komplikují testy.....	136
Proces správy testovacích dat.....	137
Sběr požadavků na data.....	137
Výroba testovacích dat	138
Přidělování dat ve sdíleném prostředí týmům a testerům.....	139
Monitoring kvality dat.....	139
Časování přípravy testovacích dat.....	140
Centralizovaně nebo každý sám?	140
Způsoby výroby dat a jejich srovnání.....	141
Ruční typování dat.....	141
Automatická výroba dat.....	141
Kopie produkčních beze změny dat	142
Částečná kopie produkčních dat.....	143
Anonymizace produkčních dat	144
Možné kombinace postupů	145
Srovnání jednotlivých způsobů.....	145
Shrnutí	147



Správa defektů.....	149
Rozdíl mezi chybou, defektem, selháním a incidentem.....	149
Role a odpovědnosti při správě defektů	150
Příčiny defektů.....	150
Připravte se na defekty	150
Atributy defektů	151

Jak defekt nahlásit.....	152
Životní cyklus defektu	152
Řízení změn.....	153
Komunikační mechanismy na projektu	154
Nástroje pro řízení testování.....	154
Závažnost, urgentnost a prioritizace defektů.....	155
Oprava defektů a přetestování.....	157
Metriky týkající se defektů a reporting.....	158
Příprava na reporting	158
Metriky a reporting defektů.....	159
Co lze vyčíst ze statistik defektů?.....	161
Lze předem odhadovat počet defektů?.....	162
Doporučené postupy a tipy z projektů	162
Kontrolní seznam (test execution checklist).....	162
Více dodavatelských týmů na jednom projektu	162
Shrnutí	163



11 Regresní testování.....	165
Co je regresní testování	165
Typy regresních chyb a jejich příčiny.....	166
Testovací strategie regresních testů	166
Regresní testy v různých úrovních.....	167
Kritéria volby typů regresních testů.....	167
Regresní testování vs. testování změn	168
Co ovlivňuje rozsah regresních testů	169
Optimalizace sady regresních testovacích případů.....	170
Plán vykonávání regresních testů.....	171
Příprava regresních testovacích případů.....	172
Na co se zaměřit při přípravě (regresních) testovacích případů	172
Testovací data pro regresní testy a jejich specifiká.....	173
Údržba regresních testovacích scénářů	173
Tým pro regresní testy a jeho motivace	174
Smoke a sanity testy.....	174
Shrnutí	175



12 Automatizované testování.....	177
Vyplatí se automatizovat testy?.....	178
Manuální vs. automatizovaný test – najdi osm rozdílů.....	178
Jak efektivně využít automatizované testy?	180
Jak to vlastně funguje?	181
Front-end testy: simulace aktivity manuálního testera	181
Jednotkové testy	183
Integrační testy.....	185
Kam patří automatizované testy ve V-modelu?.....	186

Jaké největší problémy nastávají při automatizovaném testování?....	188
Jaké základní vlastnosti má mít dobrý automatizovaný test?	190
Jak testy efektivně udržovat?	192
Jak vytvářet odolné automatizované testy?.....	193
Jak dobře strukturovat automatizované testy?	194
Jaký rozsah testů je vhodné automatizovat?	195
Co je obtížné automatizovat?	196
Jak automatizované testování zařadit do ostatních procesů?	197
Jak zapojit automatizované testy do procesu testování?.....	197
Inkrementální nebo „big-bang“ přístup?	197
Na co při nastavování procesů nezapomenout?.....	198
Průběžná integrace a související koncepty	199
Ekonomika a návratnost automatizovaných testů.....	200
Shrnutí	201



Outsourcing a další strategie dodávky testování.....	203
Komoditizace testování: pomůže nám testovací továrna?.....	204
Co je to centrum excelence v testování?	205
Jak strukturovat centrum excelence v testování?	206
Outsourcing a další strategie sourcingu testování.....	209
Jak sourcing ovlivňuje centra excelence v testování?	211
Izolované testování	211
TCoE interní (onshore insourcing)	213
Řízená testovací služba poskytovaná lokálně (onshore outsourcing).....	213
A co když jsou testeři za mořem (offshore outsourcing)?.....	216
Důvěra, kontrola a moc – jako roli hrají v outsourcingu?	219
Shrnutí	220

Literatura.....	221
-----------------	-----

Abstract.....	229
---------------	-----



Úvod

Držíte v rukou knihu, která se zabývá vybranými tématy z oblasti testování softwaru. Věříme, že vám informace a zkušenosti obsažené na následujících stránkách pomohou zefektivnit práci, nastavit nový úhel pohledu na věc nebo vyvolat následnou diskuzi.

Testování představuje průměrně 20 % až 40 % pracnosti v projektech vývoje softwaru. Pro vývoj řídicího softwaru s vysokými požadavky na spolehlivost může pracnost testování často přesáhnout pracnost vývoje. Pokud má být testování efektivní, musíme k němu přistupovat jako k samostatné odborné disciplíně – stejné, jako je analýza, návrh, architektura, vývoj nebo projektové řízení. Klíčem k úspěchu je vyvážená kombinace metodiky a praktických zkušeností.

Proč jsme se rozhodli o testování softwaru napsat další knihu? Na toto téma je k dispozici řada odborných publikací v anglickém jazyce, ale v češtině jich je pomálu. A právě místní kontext je podle našeho názoru podstatný – praktické zkušenosti a rady, které najdete na dalších stránkách, jsou pochopitelně ovlivněné projekty, z nichž pocházejí. Autoři jednotlivých kapitol sice pracovali na řadě zahraničních nebo mezinárodních projektů, ale velká část našich zkušeností pochází z českého prostředí.

Kniha poskytuje průřez oblastí, které jsou podle našeho názoru na současných testovacích projektech problematické, nebo představují rezervu z hlediska efektivity testování. Kniha ani zdaleka nepokrývá kompletní životní cyklus testování. Vědomě zanedbáváme zejména témata týkající se analýzy testování. Ta byla rámcově pokryta v knize [1] a jejich detailní popis by vydal na samostatnou publikaci.

Pro jaké typy projektů a domény se kniha hodí?

Testování softwaru je velmi různorodá disciplína. Jedno ze sedmi základních pravidel testování prezentovaných ve standardu ISTQB [2] říká, že testování je závislé na kontextu. Mobilní aplikaci elektronického bankovníctví vyvíjenou agilním způsobem budeme testovat podstatně jinak než řídicí software dopravního letadla.

Tato kniha je do jisté míry ovlivněna doménovými znalostmi a zkušenostmi jejích autorů. Většina projektů, na kterých jsme pracovali, byla řízena stylem vývoje typu vodopád (*waterfall*). Agilnímu testování se proto v knize systematicky nevěnujeme. Téma je pokryté v řadě jiných publikací, např. [3]. Nejvíce zkušeností máme z oblasti vývoje webových nebo desktopových aplikací. Projekty, které ovlivnily naše zkušenosti, byly realizované zejména v oblastech bankovníctví, pojišťovnictví a telekomunikačních služeb. Méně zkušeností máme v oblasti testování vestavěných systémů (*embedded systems*) a softwaru, kde jsou kladené vysoké požadavky na spolehlivost. Pro tyto oblasti nemusí všechny popsané metody stoprocentně odpovídat.

Jak je kniha strukturovaná

Kniha je uspořádána do 13 kapitol, každá z nich se snaží pokrýt jednu tematickou oblast bez větších závislostí na jiné kapitole. Můžete tedy pracovat s jednotlivými kapitolami, „na přeskáčku“ tak, jak se vyskytne potřeba pro rozšíření znalostí konkrétní problematiky. Jedinou výjimkou je kapitola 2, *Role testování ve firmě a v projektu*, která vysvětluje základní pojmy o testování, z nichž vychází výklad v ostatních kapitolách. Tuto kapitolu doporučujeme přečíst vždy.

Každá kapitola obsahuje závěrečné shrnutí, které rekapituluje nejdůležitější informace k danému tématu. Pokud byste nenašli čas na přečtení všech kapitol, doporučujeme prolistovat knihou a přečíst alespoň tato shrnutí.

Díčí kapitoly jsou psané různými autory, kteří jsou experty na probíranou oblast. Nevýhodou tohoto „sborníkového přístupu“ může být mírně odlišný jazykový styl jednotlivých kapitol. Věříme, že kvalita informací tuto nevýhodu výrazně převáží.

Komu je kniha určena?

Knihu jsme psali pro začínající manažery testování, nebo pro zkušené testery či analytiku testování, kteří by se časem chtěli do role manažera testování profilovat. Kromě toho samozřejmě doufáme, že nové poznatky nebo nové úhly pohledu na jednotlivé problémy najdou v knize i zkušenější manažeři testování. Pro získání základního přehledu knihu doporučujeme i projektovým manažerům.

Použitá terminologie

Při psaní knihy jsme se snažili používat odborné termíny ve překladu použitým v glosáři ISTQB [4]. V několika málo případech jsme se od tohoto zdroje uchýlili, protože jsme buď považovali za vhodnější termín ustálený v testerské praxi, nebo glosář daný termín neobsahuje.

O autorech

Miroslav Bureš

Testováním softwaru se zabývá již přes deset let. V současné době působí na katedře počítačů ČVUT FEL, kde se zabývá výzkumem a inovacemi v oblasti testovacích metodik, efektivní automatizace testování a automatizace návrhu testovacích scénářů. Zde také vyučuje předměty týkající se testování softwaru a vede řadu experimentálních projektů s cílem zvýšit efektivitu současných testovacích metod. V předchozím období vedl testovací oddělení české a slovenské pobočky Capgemini, spolupracoval s tvůrci testovací metodiky TMAP a působil v řadě projektů z oblasti financí jako manažer testování a projektový manažer.



Miroslav Renda

Od roku 1995 prošel většinou projektových rolí v oblasti dodávky softwaru. Nejdéle působil v roli projektového manažera. Je držitelem mezinárodních certifikátů ISTQB, Prince 2, ITIL a dalších. V poslední době vedl testovací oddělení v konzultační společnosti Capgemini, v současnosti řídí testování nebo dodávky IT projektů klientům ve finanční sféře. Několikrát přednášel na mezinárodní konferenci Czech-Test. Ve volném čase se věnuje rodině, sportu a práci viceprezidenta odborné neziskové organizace CaSTB.



Michal Doležel

Má za sebou více než desetiletou technicko-manažerskou praxi v oblasti testování softwaru, především z odvětví telekomunikací a finančnictví. Dnes pracuje v pražském IT hubu farmaceutické společnosti Merck Sharp & Dohme, kde vede Testing Center of Excellence pro region EMEA. Rovněž spolupracuje s katedrou informačních technologií VŠE FIS, kde vyučuje v magisterském studiu a zabývá se výzkumem. Je autorem několika příspěvků publikovaných v odborných časopisech a na zahraničních konferencích. Mimo testování jsou předmětem jeho zájmu organizační koncepty pro vývoj informačních systémů a mezioborová problematika sociálních a kulturních faktorů v IT.



Zdeněk Grössl

Téměř patnáct let testuje software, prošel pozicemi od testera až po manažera testování, převážně v oblasti financí a utilit. Jeho hlavní doménou je oblast analýzy testování, což podporuje i fakt, že je jedním z nemnoha českých držitelů certifikátu ISTQB CTAL Test Analyst. V současné době vede testovací oddělení ve společnosti Ness Czech a současně přednáší na VŠE analýzu testování. Pravidelně také vede tutoriály o analýze testování na mezinárodní konferenci CzechTest a účastní se zájmových setkání proTest.



Martin Komárek

Specialista v oblasti business analýzy a specifikace požadavků na software. V současnosti pracuje ve firmě Cactoo Software, kde působí především jako lektor a konzultant. Kariéru začal jako analytik bezpečnostně-kritických aplikací v AŽD. Od roku 2007 přednáší na ČVUT FEL tematiku softwarového inženýrství.



Ondřej Macek

Jako konzultant a školitel se podílí na řadě projektů z oblasti business analýzy a procesního řízení IT. Působil na ČVUT FEL, kde kromě realizace softwarových projektů vyučoval i předměty spojené s objektovým návrhem kódu a s řízením vývoje softwaru.



Radoslav Mlynář

V oblasti informačních technologií pracuje od roku 2001. Na začátku kariéry se zabýval implementací a administrací nástrojů pro řízení testů. Tři a půl roku strávil na projektech, kde měl na starosti implementaci technických požadavků od jejich analýzy, přes testování až po nasazení. Pět let vedl tým řízení kvality softwaru v České spořitelně. V současné době se v České spořitelně podílí na budování nové organizační jednotky – profesionálního testovacího centra v oblasti řízení testovacích prostředí a v oblasti řízení testů souhrnných celobankovních releasů.



Peter Svoboda

Testováním softwaru se zabývá od roku 2006, jeho specialitou je ekonomika testů a business pohled na testování. V minulosti působil v testovacím oddělení společnosti AVG a jako manažer testování a test konzultant ve společnosti Capgemini. V současné době buduje slovenskou větev společnosti tesena a věnuje se rozšiřování povědomí o nových trendech v testování na slovenském a českém trhu. Zároveň působí v roli manažera testování na IT projektech, především ve finančních institucích.



Poděkování

Na tvorbě knihy se podílela nejen osmička autorů. Na konečnou kvalitu díla měla velký vliv řada expertů z oblasti testování, kteří revidovali dílčí kapitoly. Za podnětné připomínky a náměty bychom chtěli poděkovat (v abecedním pořadí): Ladislavu Doubkovi, Karlovi Frajtákoví, Karolu Frühaufovi, Antonínu Gabrielovi, Anně Havlíčkové, Davidu Janotovi, Davidu Kristlovi, Štěpánu Květenkému, Tomáši Lisému, Martinovi Mičkalovi, Lubomíru Michalčákovi, Petru Neugebauerovi, Štěpánu Pelcovi, Slavoji Pískovi, Gáboru Puhallovi, Blance Rabhi, Zdeňkovi Samuelovi, Jaroslavu Strharskému, Pavlu Strnadovi, Milanovi Šterbovi, Romanu Štroblovi, Ivo Zelenkovi a Janě Zientkové. V neposlední řadě pak děkujeme recenzentům knihy doc. Ing. Valentinu Vraničovi, Ph.D., doc. RNDr. Jiřímu Barnatovi, Ph.D. a doc. Ing. Branislavu Lackovi, CSC.



Businessová hodnota testování

Peter Svoboda

Téměř každý IT projekt doprovází diskuze mezi sponzorem, businessovým vlastníkem zaváděného řešení, dodavatelem a manažerem testování o tom, jaká hloubka testování a potažmo jako množství vynaložených prostředků je žádoucí, potřebné a možné.

Cílem této kapitoly je poskytnout čtenáři argumentaci k obhájení potřeb testování vypočtených pomocí metod v kapitole 5, *Odhadování pracnosti testovacích aktivit*. Vysvětlíme si termín cena za kvalitu, podíváme se na její výpočet a postupně projdeme jednotlivými fázemi přípravy a použití argumentace nákladů na testování. Celý výklad bude proložen modelovými situacemi, které mohou manažerovi testování pomoci najít inspiraci při prosazení potřeb testování v organizaci.

1.1 Kam mohou vyšplhat náklady za nedostatečnou kvalitu

V této kapitole se nebudeme podrobně věnovat odhadování pracnosti testů, protože detaily k tomuto tématu najdete v kapitole 5, *Odhadování pracnosti testovacích aktivit*. Předpokládejme tedy, že pracnost stanovit umíme a dokážeme si tuto pracnost obhájit i před experty z oblasti testování. V praxi často narážíme na problém při obhajování ceny za testování u sponzora projektu a potažmo u dalších organizačních složek zapojených do projektu (business). Otázka, se kterou se každý manažer testování v takové chvíli potýká nejčastěji, zní: „**Proč máme tolik platit za testování? Cena za testy je příliš vysoká!**“ Vezměme si pro další výklad na pomoc událost, která se před nějakým časem skutečně stala.

V květnu 2015 vydal americký Úřad pro letectví (FAA) bezpečnostní bulletin pro všechny modifikace Boeingu 787 Dreamliner.¹ Bulletin se týkal softwarového problému v řídicí jednotce generátorů palubního napětí, ve kterém dojde po 248 dnech nepřetržitého běhu řídicí jednotky k přetečení rozsahu datového typu a následnému pádu softwaru do nouzového režimu. V praxi by to pak znamenalo okamžité vypnutí obou hlavních palubních generátorů a ztrátu napětí, což by znamenalo ohrožení bezpečnosti letu. Největší nebezpečí hrozí ve fázích vzletu a přistání, kdy by automaticky spuštěná náporová turbína díky nízké rychlosti letounu nezačala vyrábět dostatečné množství energie do šesti sekund od vypnutí generátorů, po kterou budou energii dodávat palubní baterie.

¹ FAA: Bulletin 2015-NM-058-AD, 2015 Dostupné také zde: [http://rgl.faa.gov/Regulatory_and_Guidance_Library/rgAD.nsf/0/584c7ee3b270fa3086257e38004d0f3e/\\$FILE/2015-09-07.pdf](http://rgl.faa.gov/Regulatory_and_Guidance_Library/rgAD.nsf/0/584c7ee3b270fa3086257e38004d0f3e/$FILE/2015-09-07.pdf).



Bezesporu se jedná o závažný defekt. K dubnu 2015 je v provozu 269 letadel typu Boeing 787 všech variant.

Podívejme se na detaily vnitřní implementace této funkcionality:

- V řídicím softwaru generátorů je jednoduché počítadlo, které narůstá každou setinu sekundy o 1.
- Počítadlo se vynuluje při vypnutí řídicí jednotky.
- Pro počítadlo byl použit datový typ o šířce 31 bitů. Rozsah počítadla je tedy 2^{31} milisekund.

To znamená, že po cca 248 dnech dojde k přetečení rozsahu počítadla a chybě v softwaru. V komerčním letectví je běžné, že se systémy letadla při zastavení úplně nevypínají (hlavně ty, které mají na starost napájení, což je i tento případ). K úplné odstávce dochází zpravidla při servisních zásazích. Čas běhu 248 dnů je sice nepravděpodobný, nikoliv však vyloučený. Z tohoto důvodu FAA vydala nařízení k okamžitému restartu všech jednotek ve všech variantách modelu Boeing 787, aby se čas na odstranění defektu v softwaru prodloužil o (minimálně) 248 dnů.

Co to znamená z pohledu ekonomiky (ne)testování?

Nařízení FAA je pro aerolinky (minimálně americké) závazné, tudíž jej musí provést všechny, jež provozují daný typ stroje. Jeden restart byl vyčíslen na jednu pracovní hodinu technika hodnocenou 85 USD, takže pouhá **aplikace tohoto „workaroundu“ stála americké provozovatele aerolinek 22 865 USD.**

Nepřímé finanční dopady, zejména nutnost stáhnout letadlo kvůli zásahu z provozu a ztráta reputace díky popularitě této zprávy v médiích, **mohou být o řád až dva vyšší.**

Vyřešení defektu bude nejspíše spočívat ve hloubkové analýze kódu a nahrazení všech nenulovaných proměnných s datovým typem integer 32 bit proměnnou typu integer 64 bit. Následovat budou rozsáhlé regresní testy, zátěžové testy a simulované testy dlouhodobého provozu (pokus o znovunalezení defektu). Jelikož systémy Dreamlineru obsahují přibližně 110 milionů řádků kódu, **tyto náklady se pravděpodobně vyšplhají na několik milionů dolarů.**

Zavedení testování dlouhodobé odolnosti (*soak testing*) ve vhodný okamžik přitom při podobně rozsáhlých projektech představuje náklady ve výši několika desítek tisíc USD. Jinými slovy – **investicí desítek tisíc USD do vhodného typu testu bylo možné předejít ztrátě několika milionů USD vyvolané defektem.**

Tabulka 1.1: Typické přímé náklady na opravu defektu v závislosti na fázi, ve které byla nalezena (zdroj: Capgemini)

	Analýza	Design	Vývoj	Vývojářské testy	Testování	Produkční provoz
Relativní počet zanesených defektů v dané fázi	10 %	40 %	50 %			
Relativní počet detekovaných defektů v dané fázi	3 %	5 %	7 %	25 %	50 %	10 %
Normalizované náklady na opravu defektů (EUR)	250	250	250	1 000	3 000	12 500

I v případě, že aktuálně nepracujete na projektu vývoje softwaru pro letectví, základní principy jsou obecně platné napříč odvětvími. Před několika lety provedla společnost Capgemini průzkum ceny za opravu defektů v různých fázích projektu,² zahrnující několik stovek integračních projektů z celé Evropy.

Přímé náklady na defekt, který je nalezen až v produkci, jsou tedy podle tohoto průzkumu v průměru 12,5x vyšší než náklady na odstranění defektu nalezeného v průběhu vývojářských testů. **Pokud do testování zapojíme např. revize analytických dokumentů, můžeme opravu části chyb zlevnit až 50x.**

1.2 Výpočet ceny za kvalitu

Větší IT projekty jsou ve většině případů financované z peněz, nad nimiž mají kontrolu výkonní manažeři společnosti. Mnozí manažeři testování a manažeři projektů si stěžují, že komunikace a jednání s výkonnými manažery je složité. V těchto stížnostech zaznívá, že výkonní manažeři nechtějí slyšet argumentaci a jediné, co je zajímavé, je co nejnižší cena. To nemusí být pravda. Je nutné si uvědomit, že manažeři testování a manažeři IT projektů jsou většinou technicky uvažující lidé, upřednostňující rozdílnou argumentaci a vyznávající jiné profesní hodnoty než lidé ve výkonném managementu. Výkonný management často neslyší na argumenty typu „testování sníží rizika“, „testování ušetří peníze“ (něco nedělat je přeci 100% sleva) nebo na obsáhlou argumentaci popisující technickou složitost řešení a z toho plynoucí technické komplikace. Je třeba argumentovat jinak. Výkonný management zajímá hodnota, kterou za své peníze dostanou.

Kolik tedy má sponzor zaplatit za testy? Tato otázka je v principu špatně položená. **Lepší je ptát se, co chceme pomoci testů dosáhnout? Jaká je hodnota, kterou testy přinesou?**

Odověď na tuto otázku bude dána namapovaním produktů testování na businessové cíle při zohlednění fází životního cyklu produktu. Toto mapování je přehledně zobrazené na obrázku 1.1. Zkratka MŽP znamená minimální životaschopný produkt (angl. *minimal viable product*).

	Business cíle	Business rozhodnutí	Stav produktu	Projektová aktivita	Dotazy adresované testováním	
Koncept	Zisk			Nápad, výzkum, vytvoření a dodání draftu	Identifikace závažných překážek pro realizaci	Existuje životaschopný trh? Jaká jsou businessové rizika?
Prototyp1 (Validace myšlenky)	ROI, potenciál, rizika	Množství prostředků	Draft	Vytvoření MŽP* (rychle a levně)	Kontrola, že MŽP je v souladu s vizí, je použitelný a prodejný	Konzistence s vizí? Vyhovuje použití? Nejdůležitější funkce jsou prodejné/použitelné?
Funkční prototyp	Trh, potenciál, pokrok, rizika	Více prostředků?	Beta	Proměna MŽP na finální produkt (rychle a levně)	Zhodnocení a nastoupení nejkratší cesty k zisku	Vyhovuje legislativě? Jsou businessová a technická rizika ošetřena?
Produktizace	Celková cena za nasazení < Pokračování ve zlepšování	Nasazení?	Kandidát na nasazení	Udržení nákladů na údržbu a podporu co nejnižší	Předpovědět a zabránit ztrátám (mitigace rizik)	Proč zákazníci odcházejí ke konkurenci? Která oblast podpory nás stojí nejvíce?
Nasazení	Celková cena za opravu < Ztracený čas / ušlý zisk	Oprava?	Oprava	Vyřešit rychle (a levně)	Zastavení ztrát bez způsobení nových ztrát	Dopady změn? Jaká jsou rizika? Existuje jednodušší/levnější/bezpečnější varianta?
Podpora	Ziskový potenciál	Nová verze?	Nerze N+1	Draft -> MŽP -> Kandidát na nasazení (lépe než poprvé)	Zvážení nových příležitostí k zisku bez ohrožení stávajících	Jakým způsobem můžeme vydat opravu rychle a levně?
Aktualizace	Celková cena za podporu > Potenciál zisku	Ukončení provozu?		Upgrade, upsell, alternativy	Zabezpečení hladkého přechodu pro zákazníky / uživatele	Jak ochránit data za co nejnižší cenu? Loajalita? Soukromí?

Obrázek 1.1: Mapování produktů testování na businessové cíle při zohlednění fází projektu

² Capgemini Europe: Cena za opravu defektů podle fáze, ve které jsou nalezeny, 2010. Dostupné na: <http://www.capgemini.com>.



*

V praxi se osvědčil při přípravě strategie testování následující postup:

1. Diskuze očekávané hodnoty testování se sponzory – tato aktivita by měla proběhnout jako první, jelikož očekávaná hodnota testování do určité míry definuje cíle testování.
2. Odhad pracnosti testování – jako vstup pro odhadování pracnosti jsou kromě jiného použity výstupy diskuze se sponzorem o hodnotě testování.
3. Validace odhadů se sponzorem – zde už můžete použít pro argumentaci výstupy z úvodní schůzky. Tedy domluvené businessové cíle a náklady na produkty testování, jež je pomáhají naplnit.

A jak můžeme popsat a kvantifikovat cenu za kvalitu (angl. *Cost of Quality*, COQ), zmíněnou v úvodu? Podle standardu ISTQB vstupují do ceny za kvalitu (či přesněji „ceny za nekvalitu“) následující kategorie nákladů:

- Náklady využití na prevenci defektů – sem patří např. školení vývojářů, náklady na zavedení kódovacích standardů.
- Náklady na nalezení defektů – zahrnují veškeré testovací aktivity od plánování testů, přípravy testovacích případů, revize specifikací, přípravu testovacích prostředí, jejich vykonávání až po reporting.
- Náklady vyvolané řešením defektů v době testování (angl. *cost of internal failure*) – sem patří náklady na opravu defektů ve specifikacích, opravy defektů v kódu a na ně nabalené náklady dalších rolí (nasazování do prostředí, přetestování atd.).
- Náklady vyvolané selháními v produkčním prostředí (angl. *cost of external failure*) – zahrnují nejen přímé náklady na veškeré aktivity související s opravou **případných** produkčních defektů a distribucí opraveného systému uživatelům. Velkou položkou v této škatulce mohou být náklady na sankce (např. smluvní pokuta účtovaná odběratelem), náklady na ztrátu reputace atd.

Náklady na zajištění kvality před nasazením do produkčního prostředí jsou dané součtem první tří položek:

Cena zajištění kvality = Náklady na prevenci + Náklady na nalezení defektů + Náklady na opravu

Přínos těchto investic do kvality před uvedením systému do produkčního provozu vůči možným nákladům, které by bylo nutné vynaložit, kdybychom tyto investice/aktivity nerealizovali, pak ukazuje cena za kvalitu:

Cena za kvalitu = Náklady vyvolané možným produkčním selháním – Cena zajištění kvality

Pro argumentaci opodstatnění investic do aktivit pro zajištění kvality je tedy pochopitelně nutné, aby byla cena za kvalitu kladným číslem. Zjednodušeně řečeno: má smysl testovat jen do té míry, pokud jsou náklady na testování nižší než náklady na řešení případných produkčních incidentů.

1.3 Prostředky pro argumentaci hodnoty testování z praxe

Pro lepší ilustraci výše uvedeného uvádíme smyšlený příklad inspirovaný příběhem prezentovaným v úvodu kapitoly.

Projekt: Vývoj softwaru řídicí jednotky generátorů proudu v novém typu letounu.

1.3.1 Fáze získání informací

Při diskuzi s architektem řešení, vedoucím vývoje a analytickým týmem **se snažíme získat co nejvíce podrobností o technickém řešení, způsobu vývoje a případných specifikách**. Z těchto diskuzí získáme následující výstupy:

Očekávané parametry vývoje:

- 4,3 milionu řádků kódu,
- vývoj formou „Cleanroom Development“;³

³ Cobb & Mills: Engineering Software Under Statistical QualityControl, Science Alliance, 1990.



- statistická chybovost 3 defekty na 1000 řádků kódu (zohledňuje způsob vývoje a zkušenosti z minula),
- odhad počtu chyb přítomných v softwaru řídicí jednotky: 12 900,
- celkové množství kódu ve všech systémech: 110 milionů řádků,
- odhad celkového počtu chyb přítomných v softwaru letounu: 330 000.

Další podklady:

- průměrný nálet na stroj: 25 000 hodin za 10 let.

Při prvotní diskuzi se sponzory **procházíme business cíle projektu a vyjasníme si případné nedorozumění**, abychom byli schopni správně uchopit cíle. Z diskuze získáme následující výstupy:

Business cíle:

- ROI po 1000 prodaných letadlech daného typu,
- splnění regulace **DO-178B, DO-178C**,
- dodání 3000 strojů do deseti let.

1.3.2 Fáze přípravy argumentace

Argumentaci postavíme na **požadavcích regulátora na maximální počty selhání** dílčích závažností na letovou hodinu, které jsou prezentované v druhém sloupci tabulky 1.2.

Při zohlednění vstupních informací získaných v úvodní fázi dostáváme, že v prvních 10 letech bude souhrnný nálet **75 000 000 hodin**. Po vynásobení maximálních regulátorem stanovených přípustných počtů selhání v důsledku dílčích závažností defektů získáváme maximální počty defektů uvedené ve třetím sloupci tabulky. Protože vyvíjíme pouze kód pro řídicí jednotku, je třeba násobit maximální počty přípustných defektů poměrem mezi řádky zdrojového kódu řídicí jednotky (4,3 milionů) a celkovými řádky zdrojového kódu (110 milionů). Tato výsledná hodnota je pak uvedena v posledním sloupci.

Tabulka 1.2: Maximální počty přípustných defektů podle požadavků regulátora

Závažnost defektu	Max. počet selhání za letovou hodinu	Max. počet přípustných defektů pro celé letadlo	Max. počet přípustných defektů pro řídicí jednotku
Katastrofické defekty	0,000000001	0,075	0,00293
Nebezpečné defekty	0,0000001	7,500	0,29318
Závažné defekty	0,00001	750,000	29,31818
Drobné defekty	0,001	75000,000	2931,81818

Z podobných historických projektů známe statistické rozložení pravděpodobnosti výskytu defektů pro jednotlivé závažnosti. Tato hodnota je uvedena v druhém sloupci tabulky 1.3. Prostým součinem očekávaného procenta defektů dílčí závažnosti s celkovým počtem odhadovaných defektů v řídicí jednotce (12 900) získáme hodnoty uvedené ve třetím sloupci. Pro kategorii katastrofických a nebezpečných defektů nám předchází tabulka ukazuje, že v softwaru řídicí jednotky nesmí být žádný defekt (čtvrtý sloupec). Nižší závažnosti připouští nenulový počet defektů. Ve čtvrtém sloupci tabulky 1.3 tedy vypočítáme, kolik defektů musíme odhalit pro dílčí závažnosti, abychom splnili požadavky regulátora.