

Obsah

Úvod	11
O autorech.....	13
O odborných korektorech.....	14
Poděkování	15
Předmluva	16

Část I

DDT vs. TDD

Kapitola 1

Někdo to má pozpátku	19
Problémy, které řeší testování řízené návrhem	20
Vědět, kdy je hotovo, je obtížné.....	20
Ponechání testování na později je dražší	21
Testování špatně navrženého kódu je těžké	21
Je snadné opomenout testy na úrovni zákazníka.....	22
Vývojáři bývají samolibí	22
Testy někdy postrádají smysl.....	22
Rychlý přehled metodiky DDT bez ohledu na nástroje.....	22
Struktura testování řízeného návrhem.....	23
Metodika DDT v akci.....	25
Jak se metodiky TDD a DDT odlišují	26
Ukázkový projekt: představení webové aplikace Mapplet 2.0	28
Shrnutí	29

Kapitola 2

Úvodní příklad s metodikou TDD.....	33
10 nejdůležitějších vlastností metodiky TDD	34
10. Testy řídí návrh	34

9. Celkový nedostatek dokumentace.....	34
8. Vše je test jednotky	35
7. Testy metodiky TDD nejsou tak docela testy jednotek (nebo snad ano?).....	35
6. Testy přijatelnosti poskytují zpětnou vazbu vůči požadavkům.....	35
5. Metodika TDD propůjčuje důvěru k provádění změn.....	35
4. Návrh se vyvíjí inkrementálním způsobem.....	36
3. Nějaký předběžný návrh je v pořádku.....	36
2. Metodika TDD produkuje velké množství testů	36
1. Metodika TDD je nehorázně obtížná.....	37
Přihlašování implementované pomocí metodiky TDD.....	37
Seznámení s požadavkem.....	37
Přemýšlejte o návrhu	40
Nejdříve se napíše první test psaný jako první test.....	41
Vytvoření kódu pro kontrolu přihlášení, aby test prošel	44
Vytvoření mockobjektu.....	46
Refaktorizace kódu ukazující rozvoj návrhu.....	48
Testy přijatelnosti s metodikou TDD.....	54
Závěr: metodika TDD je opravdu nehorázně obtížná	54
Shrnutí	56

Kapitola 3

Úvodní příklad s metodikou DDT..... 57

10 nejdůležitějších vlastností metodiky ICONIX/DDT	58
10. Metodika DDT zahrnuje testy obchodních požadavků	58
9. Metodika DDT zahrnuje testy scénářů.....	58
8. Testy jsou odvozené z návrhu	58
7. Metodika DDT zahrnuje testy řadičů	59
6. Metodika DDT testuje chytřeji, ne obtížněji	59
5. Testy jednotek metodiky DDT jsou „klasickými“ testy jednotek.....	59
4. Testovací případy metodiky DDT lze transformovat do testovacího kódu	59
3. Testovací případy metodiky DDT vedou k testovacím plánům	60
2. Testy metodiky DDT jsou užitečné pro vývojáře i členy QA týmů.....	60
1. Metodika DDT dokáže eliminovat nadbytečné úsilí.....	60
Přihlašování implementované pomocí metodiky DDT.....	60
Krok 1: Vytvoření diagramu robustnosti	62
Krok 2: Vytvoření testovacích případů.....	66
Krok 3: Přidání scénářů	68
Krok 4: Transformace testovacích případů testů řadiče do tříd.....	70
Krok 5: Generování testovacího kódu řadičů	72
Krok 6: Nakreslení diagramu posloupností.....	75

Krok 7: Vytvoření testovacích případů pro testy jednotek.....	78
Krok 8: Doplnění testovacího kódu	82
Shrnutí	86

Část II

Metodika DDT v praxi: Mapplet 2.0, web pro cestovatele

Kapitola 4

Seznámení s Mapplet..... 91

10 nejdůležitějších osvědčených postupů metodiky ICONIX Process/DDT.....	92
10. Vytvořte architekturu	93
9. Dohodněte se na obchodních požadavcích a testujte vůči nim	94
8. Návrh odvoďte z problémové domény	96
7. Podle storyboardů uživatelského rozhraní napište případy užití	98
6. Pro ověření, že případy užití pracují, napište testy scénářů.....	100
5. Testujte vůči konceptuálnímu a podrobnému návrhu	104
4. Model pravidelně aktualizujte	104
3. Testy přijatelnosti udržujte synchronizované s požadavky.....	111
2. Mějte automatizované testy stále aktuální	112
1. Kandidáta na finální verzi porovnejte s původními případy užití.....	112
Shrnutí	116

Kapitola 5

Podrobný návrh a testování jednotek..... 117

10 nejdůležitějších „úkolů“ při testování jednotek.....	118
10. Začněte diagramem posloupností.....	119
9. Na základě návrhu identifikujte testovací případy.....	120
8. Pro každý testovací případ napište scénáře.....	122
7. Testujte chytreji: nepište překrývající se testy.....	124
6. Testovací případy transformujte do jazyka UML.....	126
5. Napište testy jednotek a doprovodný kód.....	130
4. Napište testy jednotek jako testy „bílé skříňky“	133
3. Použijte framework pro mock objekty.....	137
2. Algoritmickou logiku testujte pomocí testů jednotek.....	140
1. Napište samostatnou sadu integračních testů	141
Shrnutí	142

Kapitola 6

Konceptuální návrh a testování řadičů 143

10 nejdůležitějších „úkolů“ při testování řadičů	145
10. Začněte diagramem robustnosti	145
9. Z řadičů odvodte testovací případy	148
8. Pro každý testovací případ definujte jeden či více scénářů	151
7. Vyplňte pole Description, Input a Acceptance Criteria	154
6. Vygenerujte testovací třídy	155
5. Implementujte testy	159
4. Napište kód, který se snadno testuje	160
3. Napište testy řadičů jako testy „šedé skříňky“	162
2. Zfetezte testy řadičů dohromady	163
1. Napište samostatnou sadu integračních testů	165
Shrnutí	166

Kapitola 7

Testování přijatelnosti: rozvinutí scénářů

v případě užití..... 167

10 nejdůležitějších „úkolů“ při testování scénářů	169
Případy užití aplikace Mapplet	169
10. Začněte popisným případem užití	170
9. Proveďte transformaci na strukturovaný scénář	173
8. Ujistěte se, že všechny cesty mají kroky	174
7. Přidejte předběžné a následné podmínky	175
6. Vygenerujte diagram činností	175
5. Rozviňte „vlákna“ pomocí externích testů	176
4. Umístěte testovací případ do diagramu testovacích případů	178
3. Ponořte se do zobrazení testování nástroje Enterprise Architect	179
2. Do testovacích scénářů přidávejte podrobnosti	179
1. Vygenerujte dokument s plánem testů	180
Morální ponaučení	181
Shrnutí	184

Kapitola 8

Testování přijatelnosti: obchodní požadavky 185

10 nejdůležitějších „úkolů“ při testování požadavků	186
10. Začněte doménovým modelem	187
9. Napište testy obchodních požadavků	189
8. Požadavky vymodelujte a uspořádejte	189
7. Vytvořte testovací případy z požadavků	192

6. Projděte svůj plán se zákazníkem.....	194
5. Napište ruční popisy testů.....	198
4. Napište automatizované testy požadavků.....	198
3. Exportujte testovací případy.....	199
2. Testovací případy náležitě zpřístupněte.....	199
1. Zapojte svůj tým!.....	200
Shrnutí.....	200

Část III

Pokročilá metodika DDT

Kapitola 9

Antivzory při testování jednotek.....205

Chrám zkázy (neboli kód).....	206
Celkový pohled.....	207
Třída HotelPriceCalculator.....	208
Podpůrné třídy.....	209
Servisní třídy.....	210
Antivzory.....	212
10. Komplexní konstruktor.....	212
9. Stratosférická hierarchie tříd.....	213
8. Statický spouštěč.....	215
7. Statické metody a proměnné.....	217
6. Návrhový vzor Singleton.....	218
5. Těsně svázaná závislost.....	221
4. Obchodní logika v kódu uživatelského rozhraní.....	223
2. Servisní objekty deklarované jako finální.....	225
1. Nedomyšlené funkce dobromyslného programátora.....	225
Shrnutí.....	225

Kapitola 10

Návrh s ohledem na snazší testování.....227

Seznam deseti nejdůležitějších úkolů pro „návrh s ohledem na testování“.....	228
Chrám zkázy – důkladně pročištěný.....	229
Případ užití – pochopení toho, co vlastně chceme dělat.....	230
Identifikování testů radičů.....	231
Test výpočtu celkové ceny.....	232
Test načtení poslední ceny.....	233

Návrh s ohledem na snazší testování.....	234
10. Inicializační kód udržujte mimo konstruktor.....	235
9. S dědičností šetřete	236
8. Vyvarujte se statických inicializačních bloků.....	237
7. Používejte metody a proměnné na úrovni objektu	238
6. Nepoužívejte návrhový vzor Singleton	238
5. Udržujte své třídy oddělené	239
4. Nedávejte obchodní logiku do kódu uživatelského rozhraní	241
3. Použijte testování černé skříňky a šedé skříňky	246
2. Modifikátor „final“ používejte pro konstanty a obecně jím neoznačujte komplexní typy, jako jsou servisní objekty	246
1. Držte se případů užití a návrhu	247
Podrobný návrh pro případ užití Quote Hotel Price.....	247
Test řadiče pro výpočet celkové ceny	249
Test řadiče pro načtení poslední ceny	249
Předělaný návrh a kód	250
Shrnutí	251

Kapitola 11

Automatizované integrační testování..... 253

Seznam 10 nejdůležitějších „úkolů“ pro integrační testování.....	254
10. Testovací vzory hledejte ve svém konceptuálním návrhu.....	255
9. Nezapomeňte na testy zabezpečení	256
8. Rozhodněte, která „úroveň“ integračních testů se má psát	258
7. Integrační testy na úrovni řadičů či jednotek odvoďte z konceptuálního návrhu	259
6. Testy scénářů odvoďte ze scénářů případů užití.....	262
5. Napište celkové testy scénářů.....	263
4. Použijte testovací framework „přátelský k obchodnímu hledisku“	267
3. Kód grafického uživatelského rozhraní testujte v rámci testů scénářů	270
2. Nepodceňujte obtížnost integračního testování.....	270
1. Nepodceňujte hodnotu integračních testů	274
Hlavní body při psaní integračních testů.....	274
Shrnutí	276

Kapitola 12

Testování algoritmů 277

10 nejdůležitějších „úkolů“ při testování algoritmů	278
10. Začněte řadičem z konceptuálního návrhu	279
9. Rozvíňte řadiče do návrhu algoritmu	281
8. Diagram volně propojte s doménovým modelem.....	282
7. Rozdělte rozhodovací uzly obsahující více než jednu kontrolu	284
6. Pro každý uzel vytvořte testovací případ.....	284
5. Pro každý testovací případ definujte testovací scénáře	285
4. Vytvořte vstupní data z nejrůznějších zdrojů	288
3. Přiřadte logický tok do jednotlivých metod a tříd.....	289
2. Napište testy jednotek jako testy „bílé skříňky“	293
1. Aplikujte metodiku DDT na další návrhové diagramy	303
Shrnutí	304

Příloha

Alenka v říši případů užití..... 305

Úvod	306
Část 1	306
Alenka při čtení usnula.....	307
Příslib vývoje řízeného případy užití	307
Model analýzy spojuje text případů užití s objekty.....	308
Jednoduché a přímočaré.....	308
Stereotypy <<includes>> a <<extends>>	308
Máme zpoždění! Musíme začít programovat!	309
Alenka se podivuje, jak se dostat od případů užití ke kódu.....	309
Abstraktní... podstata	309
Až příliš abstraktní?.....	310
Teleocentricita... ..	310
Skutečně je nutné to vše specifikovat pro každý případ užití?	311
Část 2	312
Alenka dostala žízeň.....	312
Alenka pociťuje mdloby.....	313
Imagine... (s omluvou Johnu Lennonovi)	313
Párové programování znamená, že se nikdy nezaznamenají požadavky	314
Na zaznamenání požadavků není čas.....	315
Stejně tak můžete říci, že „kód je návrh“	315
Koho zajímají případy užití?	316
Projekt C3 ukončen.....	317

Jen a pouze jedinkrát?	318
Alenka odmítá začít programovat bez zapsaných požadavků	318
Spáchala jsi VPN... ..	320
Model CMM je mrtvý! Pryč s její hlavou!	321
Seriózní refaktorování návrhu	321
Část 3	321
Alenka se probouzí	322
Uzavření díry mezi „Co“ a „Jak“	323
Statické a dynamické modely jsou spojené dohromady	323
Přidělování chování se odehrává na diagramu posloupností	323
Ponaučení, které z toho plyne	324

Závěr

Byl smažno a lepě svyhlé testy.....	325
--	------------

Rejstřík	331
-----------------------	------------