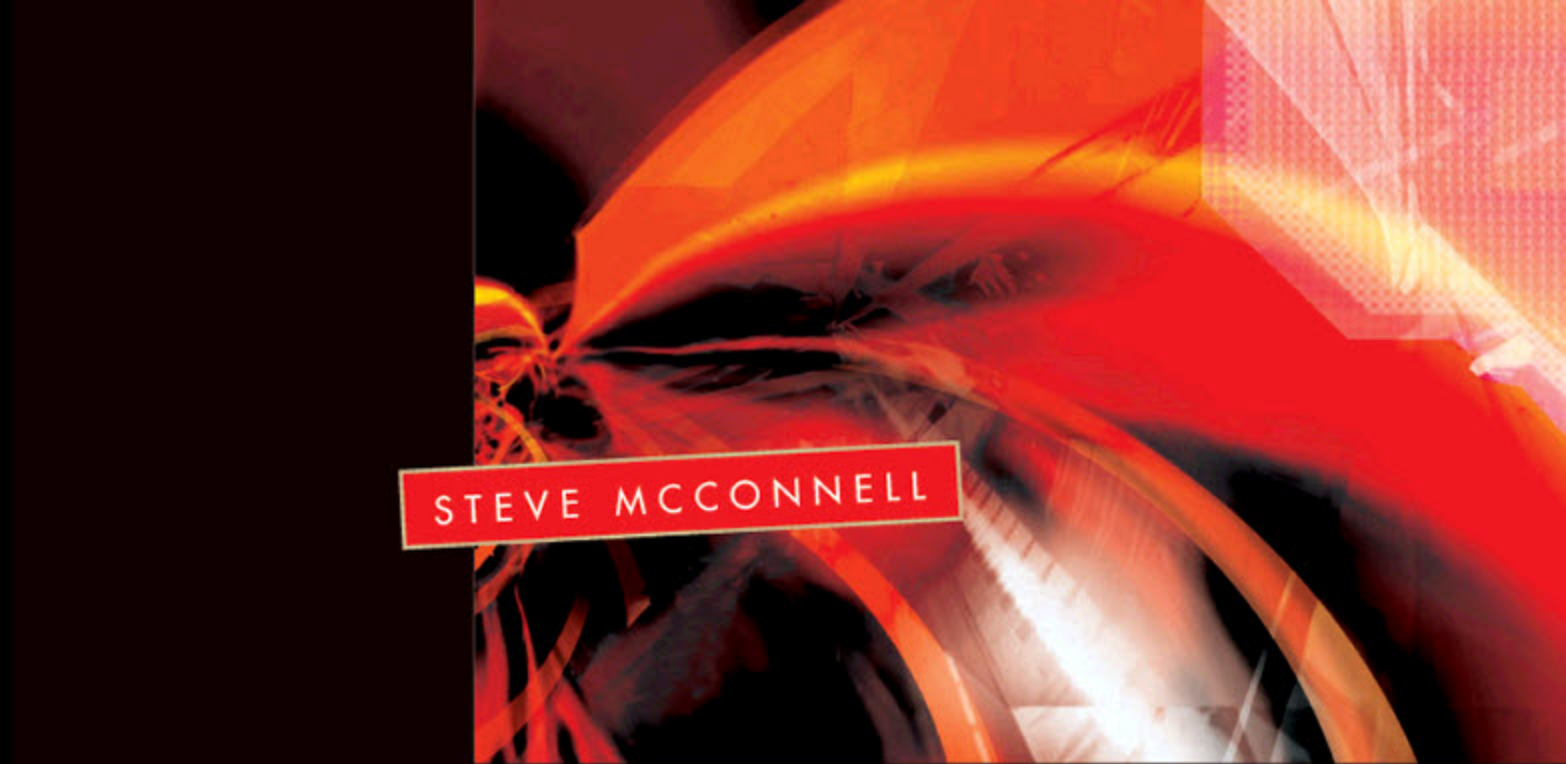




STEVE MCCONNELL

DOKONALÝ KÓD

UMĚNÍ
PROGRAMOVÁNÍ
A TECHNIKY
TVORBY
SOFTWARE

LEGENDÁRNÍ KNIHA
CODE COMPLETE
V ČEŠTINĚ!

OPTIMÁLNÍ STRUKTURA PROGRAMU
SROZUMITELNOST A OPAKOVANÁ POUŽITELNOST KÓDU
MINIMALIZACE CHYB, LADĚNÍ A VYLEPŠOVÁNÍ KÓDU
TÝMOVÉ PROGRAMOVÁNÍ A VYUŽITÍ OSOBNÍCH DISPOZIC
JAK UDRŽET KVALITU VE VŠECH FÁZÍCH PROJEKTU


press

Steve McConnell

Dokonalý kód

Umění programování a techniky tvorby software

Computer Press, a.s.

Brno

2005

Dokonalý kód

Umění programování a techniky tvorby software

Steve McConnell

Copyright © Computer Press, a.s. 2005. Vydání první. Všechna práva vyhrazena.
Vydalo nakladatelství Computer Press, a.s. jako svou 1214. publikaci.

Vydavatelství a nakladatelství Computer Press, a.s.,
nám. 28. dubna 48, 635 00 Brno, knihy.cpress.cz

ISBN 80-251-0849-X

Prodejní kód: K1148

Překlad: Bogdan Kiszka

Jazyková korektura: Josef Novák

Vnitřní úprava: Petr Klíma

Sazba: Bogdan Kiszka

Rejstřík: Bogdan Kiszka

Obálka: Martin Sodomka

Komentář na zadní straně obálky: Martin Domes

Technická spolupráce: Jiří Matoušek

Odpovědný redaktor: Martin Domes

Technický redaktor: Jiří Matoušek

Produkce: Petr Baláš

Copyright 2004 by Microsoft Corporation. Original English language edition Copyright ©2004
by Steve McConnell.

Translation: © Computer Press, 2005.

Autorizovaný překlad z originálního anglického vydání Code Complete, Second Edition.

Originální copyright: © Microsoft Corporation Inc./Steve McConnell, 2005.

Překlad: © Computer Press, 2005.

**Žádná část této publikace nesmí být publikována a šířena žádným způsobem
a v žádné podobě bez výslovného svolení vydavatele.**

Computer Press, a.s., nám. 28. dubna 48, 635 00 Brno
tel.: 546 122 111, fax: 546 122 112

Objednávejte na: **knihy.cpress.cz**
distribuce@cpress.cz

Bezplatná telefonní linka: **800 555 513**

Dotazy k vydavatelské činnosti směřujte na: **knihy@cpress.cz**

Máte-li zájem o pravidelné zasílání informací o knižních novinkách do Vaší e-mailové schránky,
zašlete nám zprávu, obsahující váš souhlas se zasíláním knižních novinek, na adresu
novinky@cpress.cz.



Novinky k dispozici ve dni vydání, slevy, recenze,
zajímavé programy pro firmy i koncové zákazníky.

Stručný obsah

Část 1: Základy

Kapitola 1:	Vítejte při stavbě softwaru	31
Kapitola 2:	Metafora pro rychlejší pochopení vývoje softwaru	37
Kapitola 3:	Dvakrát měř, jednou řež: Vstupní opatření	49
Kapitola 4:	Klíčová stavební rozhodnutí	83

Část 2: Tvorba vysoce kvalitního kódu

Kapitola 5:	Návrh během stavby	95
Kapitola 6:	Pracovní třídy	143
Kapitola 7:	Vysoce kvalitní rutiny	177
Kapitola 8:	Defenzivní programování	201
Kapitola 9:	Proces programování v pseudokódu	227

Část 3: Proměnné

Kapitola 10:	Obecně o užití proměnných	249
Kapitola 11:	Význam názvů proměnných	271
Kapitola 12:	Základní datové typy	301
Kapitola 13:	Neobvyklé datové typy	329

Část 4: Příkazy

Kapitola 14:	Sekvenční uspořádání kódu	357
Kapitola 15:	Práce s podmínkovými příkazy	365
Kapitola 16:	Řídící cykly	379
Kapitola 17:	Neobvyklé řídicí struktury	403
Kapitola 18:	Metody řízené tabulkami	425
Kapitola 19:	Obecná témata spojená s řízením	445

Část 5: Vylepšení kódu

Kapitola 20:	Kvalita softwaru	477
Kapitola 21:	Stavba ve spolupráci	491
Kapitola 22:	Vývojářské testování	509
Kapitola 23:	Ladění	543
Kapitola 24:	Restrukturalizace kódu (refaktorování)	571
Kapitola 25:	Strategie ladění výkonu	593
Kapitola 26:	Techniky ladění výkonu	615

Část 6: Systémové úvahy

Kapitola 27:	Jak velikost programu ovlivňuje jeho stavbu	653
Kapitola 28:	Řízení stavby	665
Kapitola 29:	Integrace	693
Kapitola 30:	Programovací nástroje	713

Část 7: Softwarové mistrovství

Kapitola 31:	Rozvržení a styl	731
Kapitola 32:	Dostatečně výmluvný kód	777
Kapitola 33:	Povahové vlastnosti	815
Kapitola 34:	Motivy softwarových dovedností	831
Kapitola 35:	Kde najdete další informace	849
Bibliografie		859
Rejstřík		877



Obsah

Část 1: Základy

Kapitola 1

Vítejte při stavbě softwaru 31

- 1.1 Co je to stavba softwaru? 32
- 1.2 Proč je stavba softwaru tak důležitá? 35
- 1.3 Jak číst tuto knihu 36

Kapitola 2

Metafory pro rychlejší pochopení vývoje softwaru 37

- 2.1 Jak je důležité mít metafory 38
- 2.2 Jak používat softwarové metafory 40
- 2.3 Běžné softwarové metafory 41

Kapitola 3

Dvakrát měř, jednou řež: Vstupní opatření 49

- 3.1 Význam vstupních opatření 50
- 3.2 Jak určit povahu softwaru, na němž pracujete 56
- 3.3 Příprava na definici problému 61
- 3.4 Příprava definice požadavků 63
- 3.5 Příprava architektury 68
- 3.6 Čas věnovaný přípravám 78

Kapitola 4

Klíčová stavební rozhodnutí 83

- 4.1 Volba programovacího jazyka 84
- 4.2 Programovací konvence 88
- 4.3 Vaše místo na technologické vlně 89
- 4.4 Volba hlavních stavebních postupů 91

Část 2: Tvorba vysoce kvalitního kódu

Kapitola 5

Návrh během stavby 95

- 5.1 Návrhové úkoly 97
- 5.2 Klíčové návrhové koncepce 99
- 5.3 Návrhové stavební bloky: Heuristika 108
- 5.4 Návrhové postupy 130
- 5.5 Interpretace oblíbených metodik 138

Kapitola 6

Pracovní třídy **143**

6.1 Základy tříd: Abstraktní datové typy	144
6.2 Dobrá rozhraní tříd	151
6.3 Problematika návrhu a implementace	161
6.4 Důvody pro tvorbu třídy	169
6.5 Jazykové otázky	173
6.6 Nad rámec tříd: Balíčky	174

Kapitola 7

Vysoce kvalitní rutiny **177**

7.1 Rozumné důvody pro tvorbu rutin	180
7.2 Návrh na úrovni rutiny	184
7.3 Dobré názvy rutin	187
7.4 Jak dlouhá může rutina být?	189
7.5 Jak používat parametry rutin	190
7.6 Zvláštní úvahy na téma užití funkcí	195
7.7 Makra a vložené rutiny	197

Kapitola 8

Defenzivní programování **201**

8.1 Ochrana programu před zadáním neplatných informací	202
8.2 Aserce	204
8.3 Techniky ošetřování chyb	208
8.4 Výjimky	212
8.5 Jak zabezpečit program, aby zvládl škody napáchané chybami	217
8.6 Podpora ladění	218
8.7 Jak určit, jaký podíl defenzivního programování nechat v ostrém kódu	222
8.8 Hlavně to s defenzivním programováním nepřehánět	223

Kapitola 9

Proces programování v pseudokódu **227**

9.1 Shrnutí jednotlivých kroků tvorby tříd a rutin	228
9.2 Co hovoří pro pseudokód	229
9.3 Stavba rutin pomocí procesu programování v pseudokódu (PPP)	232
9.4 Alternativy k PPP	245

Část 3: Proměnné

Kapitola 10

Obecně o užití proměnných **249**

10.1 Datová gramotnost	250
10.2 Snadná deklarace proměnných	252

10.3 Jak inicializovat proměnné	253
10.4 Rozsah	257
10.5 Persistence	263
10.6 Kdy vzniká vazba?	264
10.7 Vztah mezi datovými typy a řídicími strukturami	265
10.8 Každá proměnná má mít jeden účel	267

Kapitola 11

Význam názvů proměnných **271**

11.1 Obecné úvahy na téma volby dobrých názvů	272
11.2 Pojmenování jednotlivých datových typů	277
11.3 Význam konvencí pojmenování	282
11.4 Neformální konvence pojmenování	284
11.5 Standardizované předpony	291
11.6 Tvorba krátkých, srozumitelných názvů	293
11.7 Jaké názvy nepoužívat	296

Kapitola 12

Základní datové typy **301**

12.1 Čísla obecně	302
12.2 Celá čísla	304
12.3 Reálná čísla	305
12.4 Znaky a textové řetězce	308
12.5 Booleovské proměnné	312
12.6 Výčtové typy	313
12.7 Pojmenované konstanty	318
12.8 Pole	320
12.9 Tvorba vlastních typů (přezdívání)	322

Kapitola 13

Neobvyklé datové typy **329**

13.1 Struktury	330
13.2 Ukazatele	333
13.3 Globální data	345

Část 4: Příkazy

Kapitola 14

Sekvenční uspořádání kódu **357**

14.1 Příkazy, které musí být vykonány v určitém pořadí	358
14.2 Příkazy, které mohou být vykonány v libovolném pořadí	361

Kapitola 15

Práce s podmínkovými příkazy 365

- 15.1 Příkazy if 366
- 15.2 Přepínače 372

Kapitola 16

Řídicí cykly 379

- 16.1 Volba správného typu cyklu 380
- 16.2 Řízení cyklu 385
- 16.3 Snadná tvorba cyklů – vezmeme to z opačného konce 397
- 16.4 Souvislost mezi cykly a poli 399

Kapitola 17

Neobvyklé řídicí struktury 403

- 17.1 Více návratových cest z rutiny 404
- 17.2 Rekurze 406
- 17.3 Příkaz goto 410
- 17.4 V kontextu neobvyklých řídicích struktur 422

Kapitola 18

Metody řízení tabulkami 425

- 18.1 Obecné úvahy na téma užití metod řízených tabulkami 426
- 18.2 Tabulky s přímým přístupem 427
- 18.3 Tabulky s indexovaným přístupem 438
- 18.4 Tabulky se schodovým přístupem 440
- 18.5 Další příklady prohledávání tabulek 443

Kapitola 19

Obecná témata spojená s řízením 445

- 19.1 Booleovské výrazy 446
- 19.2 Složené výrazy (bloky) 457
- 19.3 Prázdné příkazy 458
- 19.4 Jak zabránit příliš hlubokému vnoření 459
- 19.5 Základy programování: Strukturované programování 468
- 19.6 Řídicí struktury a složitost 471

Část 5: Vylepšení kódu

Kapitola 20

Kvalita softwaru 477

- 20.1 Atributy kvality softwaru 478
- 20.2 Techniky zlepšení kvality softwaru 480
- 20.3 Relativní efektivnost kvalitních technik 484

20.4 Kdy zajišťovat kvalitu	487
20.5 Obecné zásady kvality softwaru	487

Kapitola 21

Stavba ve spolupráci **491**

21.1 Přehled metod vývoje ve spolupráci	492
21.2 Párové programování	495
21.3 Formální inspekce	497
21.4 Jiné metody společného vývoje	503

Kapitola 22

Vývojářské testování **509**

22.1 Vliv vývojářského testování na kvalitu softwaru	511
22.2 Doporučené postupy pro vývojářské testování	513
22.3 Tipy pro zlepšení testování	515
22.4 Typické chyby	526
22.5 Nástroje pro podporu testování	531
22.6 Jak zlepšit testování	536
22.7 Ukládejte výsledky testování	537

Kapitola 23

Ladění **543**

23.1 Obecný pohled na ladění	544
23.2 Hledáme chyby	548
23.3 Oprava chyby	558
23.4 Psychologické úvahy o ladění	561
23.5 Ladicí nástroje – samozřejmé i ty méně samozřejmé	564

Kapitola 24

Restrukturalizace kódu (refaktorování) **571**

24.1 Kudy se ubírá evoluce softwaru	573
24.2 Úvod do refaktorování	573
24.3 Charakteristické restrukturalizace	579
24.4 Jak bezpečně restrukturalizovat	586
24.5 Strategie restrukturalizace	588

Kapitola 25

Strategie ladění výkonu **593**

25.1 Výkon obecně	594
25.2 Úvod do problematiky ladění výkonu	598
25.3 Různé formy oběžných a těžkopádných programů	603
25.4 Měření	609
25.5 Iterace	610

25.6 Shrnutí přístupů k ladění výkonu	611
---------------------------------------	-----

Kapitola 26

Techniky ladění výkonu **615**

26.1 Logika	617
26.2 Cykly	622
26.3 Transformace dat	630
26.4 Výrazy	635
26.5 Rutiny	644
26.6 Přepis v jazyku nižší úrovně	644
26.7 Čím více usilujete o změnu, tím méně se vám to daří	648

Část 6: Systémové úvahy

Kapitola 27

Jak velikost programu ovlivňuje jeho stavbu **653**

27.1 Komunikace a velikost	654
27.2 Spektrum velikostí projektu	655
27.3 Vliv velikosti projektu na chyby	656
27.4 Vliv velikosti projektu na produktivitu	657
27.5 Vliv velikosti projektu na vývojové aktivity	658

Kapitola 28

Řízení stavby **665**

28.1 Podpora dobrých programátorských postupů	667
28.2 Systém řízení konfigurace	669
28.3 Odhad harmonogramu stavby	675
28.4 Měření	681
28.5 Programátoři jsou také lidé	684
28.6 Jak řídit svého vedoucího	689

Kapitola 29

Integrace **693**

29.1 Význam způsobu integrace	694
29.2 Integrovat fázově, nebo přírůstkově?	695
29.3 Strategie přírůstkové integrace	698
29.4 Každodenní překlad a odzkoušení	705

Kapitola 30

Programovací nástroje **713**

30.1 Návrhové nástroje	714
30.2 Nástroje pro práci se zdrojovým kódem	715
30.3 Nástroje pro práci se spustitelným kódem	720

30.4 Na prostředí orientované na nástroje	724
30.5 Tvorba vlastních programovacích nástrojů	725
30.6 Pohádka plná nástrojů	726

Část 7: Softwarové mistrovství

Kapitola 31

Rozvržení a styl 731

31.1 Základy rozvržení	732
31.2 Techniky rozvržení	738
31.3 Styly rozvržení	740
31.4 Rozvržení řídicích struktur	747
31.5 Rozvržení jednotlivých příkazů	754
31.6 Rozvržení komentářů	764
31.7 Rozvržení rutin	767
31.8 Rozvržení tříd	769

Kapitola 32

Dostatečně výmluvný kód 777

32.1 Externí dokumentace	778
32.2 Programovací styl jako dokumentace	779
32.3 Komentovat, či nekomentovat?	782
32.4 Klíč k účinným komentářům	785
32.5 Techniky tvorby komentářů	791
32.6 Standardy IEEE	810

Kapitola 33

Povahové vlastnosti 815

33.1 Jakou mají povahové vlastnosti souvislost s tématem této knihy?	817
33.2 Inteligence a skromnost	817
33.3 Zvědavost	818
33.4 Duševní upřímnost	822
33.5 Komunikace a spolupráce	824
33.6 Kreativita a disciplína	824
33.7 Lenost	825
33.8 Vlastnosti s menším významem, než by se mohlo zdát	826
33.9 Návyky	828

Kapitola 34

Motivy softwarových dovedností 831

34.1 Jak zvítězit nad složitostí	832
34.2 Zvolte vhodný postup	834
34.3 Pište programy především pro lidi, teprve pak pro stroje	835

34.4 Programujte do jazyka, nikoli v něm	837
34.5 Soustřeďte svou pozornost pomocí konvencí	838
34.6 Programujte z hlediska problémové domény	839
34.7 Pozor, padající kamení	841
34.8 Opakujte znovu a znovu	843
34.9 Oddělte software od náboženství	844

Kapitola 35

Kde najdete další informace **849**

35.1 Informace o stavbě softwaru	850
35.2 Témata přesahující rámec stavby	851
35.3 Časopisy	853
35.4 Plán doporučené četby vývojáře softwaru	854
35.5 Staňte se členy profesní organizace	856

Přílohy

Bibliografie **859**

Rejstřík **877**

Další chvála na knihu Dokonalý kód (v originále Code Complete)

„Je to excelentní průvodce programovacím stylem a stavbou softwaru.“ – Martin Fowler, autor knihy *Refactoring (Refaktoring – zlepšení existujícího kódu)*, Grada Publishing 2003).

„Kniha *Code Complete* Steva McConnella... nabízí programátorům nejrychlejší cestu k moudrosti... Jeho kniha se čte velmi příjemně. Ani na okamžik nezapomenete, že k vám promlouvá skrze těžce nabyté zkušenosti.“ – Jon Bentley, autor knihy *Programming Pearls, 2nd ed.*

„Je to jednoduše nejlepší kniha o softwarové stavbě, jakou jsem kdy četl. Každý vývojář by měl mít její výtisk a měl by si každý rok knihu přečíst. Přestože ji čtu každý rok alespoň jednou, stále se z ní učím i po devíti letech!“ – John Robbins, autor knihy *Debugging Applications for Microsoft .NET and Microsoft Windows*.

„Dnešní software musí být robustní a odolný, a zabezpečený kód začíná u disciplinované softwarové stavby. Po deseti letech od jejího vydání stále není lepšího pramene než *Code Complete*.“ – Michael Howard, inženýrství zabezpečení v Microsoft Corporation, spoluautor knihy *Writing Secure Code*.

„Zevrubné posouzení taktických problémů, jež přechází do tvorby dobře navrženého programu. McConellovo dílo zahrnuje tak odlišná témata, jako jsou architektura, standardy kódování, testování, integrace a povaha softwarového řemesla.“ – Grady Booch, autor knihy *Object Solutions*.

„Nepostradatelnou encyklopedií je pro softwarového vývojáře kniha *Code Complete*, kterou napsal Steve McConnell. Podtitul *Praktická příručka softwarové stavby* přesně říká, čím tato 850stránková kniha přesně je. Jejím cílem je zmenšit propast mezi znalostmi špičkových odborníků a profesorů (například Yourdona a Pressmana) na jedné straně a běžnou programátorskou populací na straně druhé a pomoci vývojářům psát rychleji lepší programy, jež jim nebudou způsobovat zbytečné bolesti hlavy... Výtisk této knihy by měl mít ve své knihovničce každý vývojář. Její styl a obsah je praktický po všech stránkách.“ – Chris Loosley, autor knihy *High-Performance Client/Server*.

„Původní kniha Steva McConnella nazvaná *Code Complete* je jedním z nejpřístupnějších děl pojednávajících podrobně o metodách softwarového vývoje...“ Erik Bethke, autor knihy *Game Development and Production*.

„Zlatý důl užitečných informací a rad týkajících se široké škály problémů v oblasti návrhu a tvorby dobrého softwaru.“ – John Dempster, autor knihy *The Laboratory Computer: A Practical Guide for Physiologists and Neuroscientists*.

„Máte-li skutečně zájem o zkvalitnění svých programátorských dovedností, musíte mít knihu Steva McConnella *Code Complete*.“ – Jean J. Labrosse, autor knihy *Embedded Systems Building Blocks: Complete and Ready-To-Use Modules in C*.

„Steve McConnell napsal jednu z nejlepších knih týkajících se softwarového vývoje, jenž je nezávislý na počítačovém prostředí... *Code Complete*.“ – Kenneth Rosen, autor knihy *Unix: The Complete Reference*.

„Asi tak dvakrát za století se setkáte s knihou, která zkrátí dobu nutnou pro sběr praktických zkušeností a ušetří vám celé roky skutečného očistce... Neumím dostatečně expresivně vyjádřit, jak dobrá tato kniha doopravdy je. Název *Code Complete* dost pokulhává za významem tak brilantního díla.“ Jeff Duntermann, *PC Techniques*.

„Nakladatelství Microsoft Press vydalo knihu, kterou považuji za rozhodující publikaci o softwarové stavbě. Tato kniha patří do knihovničky každého softwarového vývojáře.“ – Warren Keuffel, *Software Development*.

„Tuto vynikající knihu by si měl přečíst každý programátor.“ – T. L. (Frank) Pappas, *Computer*.

„Toužíte-li se stát profesionálním programátorem, může být oněch 35 dolarů nejlépe investovanou částkou ve vašem životě. Nezůstaňte u četby této recenze. Jednoduše vyběhněte z domu a rychle si tuto knihu kupte. McConnellovým cílem je zmenšit propast mezi znalostmi špičkových odborníků a běžnou programátorskou populací... Skvělý počín, navíc úspěšný.“ – Richard Mateosian, *IEEE Micro*.

„*Code Complete* by si měli přečíst všichni..., jež mají s vývojem softwaru něco společného.“ – Tommy Usher, *C Users Journal*.

„Rozhodl jsem se rozhlédnout poněkud dále než obvykle a bez výhrad mohu doporučit knihu Steva McConnella *Code Complete*... Výtisk, který jsem si zakoupil, nahradil vedle mé klávesnice referenční manuál API.“ Jim Kyle, *Windows Tech Journal*.

„Tato velmi dobře napsaná, ale objemná kniha je pravděpodobně nejlepším dílem na téma praktických aspektů softwarové implementace.“ – Tommy Usher, *Embedded Systems Programming*.

„Je to nejlepší kniha o softwarovém inženýrství, kterou jsem kdy četl.“ Edward Kenworth, *EXE Magazine*.

„Tato kniha zasluhuje, aby se stala klasikou. Měla by být povinnou četbou nejen pro všechny vývojáře, ale také pro všechny, jež jsou odpovědní za jejich řízení.“ – Peter Wright, *Program Now*.

Mé ženě Ashlie, jež sice nemá s počítači mnoho společného, ale která má mnoho společného se zbytkem mého života, jež obohacuje více způsoby, než jsem schopen popsat.

Předmluva

Odstup mezi nejlepšími postupy v softwarovém inženýrství a průměrnou praxí je velký – snad ještě větší než v jakékoli jiné inženýrské disciplíně. Nástroj, jenž šíří dobré postupy, by mohl být velmi důležitý.

– Fred Brooks

Napsáním této knihy jsem chtěl především zaplnit mezeru mezi znalostmi skutečných autorit oboru či profesorů na straně jedné a běžnou komerční praxí na straně druhé. Mnohé velmi výkonné programovací techniky se celá léta, než po kapkách stečou do programátorského povědomí, ukrývají v útrobách deníků či akademických elaborátů.

Přestože praxe určující směr vývoje softwarových produktů pokročila v posledních letech rapidně kupředu, o obecné praxi to bohužel říci nelze. Spousta programů stále obsahuje mnoho chyb. Nemálo projektů není dokončeno v plánovaném čase, narůstají rozpočty a výsledky často neuspokojují potřeby cílových uživatelů. Vývojáři, kteří se pohybují v softwarovém průmyslu, jakož i ti, kteří se pohybují v akademickém prostředí, už přece objevili účinné postupy, jež eliminují většinu programátorských problémů, s nimiž jsme se potýkali do sedmdesátých let. Protože se tyto postupy kromě vysoce specializovaných technických časopisů nikde neobjevují, většina organizací zabývajících se programováním je prostě dodnes nepoužívá. Studie hovoří o tom, že cesta od výzkumného vývoje až ke komerční praxi trvá zhruba 5 až 15 let (Raghavan and Chand 1989, Rogers 1995, Parnas 1999). Tato příručka popsaný proces zkracuje, neboť zpřístupňuje klíčové objevy průměrnému programátorovi.

Komu je tato kniha určena

Výzkum a programovací zkušenosti shromážděné v této příručce vám pomohou vytvářet velmi kvalitní software. Vaše práce bude rychlejší a nebudete muset řešit tolik problémů jako dosud. Kniha vám pomůže proniknout do podstaty věci, pomůže vám odpovědět na otázku, proč jste se s problémy potýkali v minulosti. Pomůže vám však také zjistit, jak se jim vyhnout v budoucnu. Programovací postupy popisované v této knize vám pomohou udržet pod kontrolou dokonce i velké projekty. Pomohou vám úspěšně udržovat a upravovat software, změní-li se požadavky na některý z vašich projektů.

Zkušeným programátorům

Tato příručka slouží zkušeným programátorům, jež hledají zevrubný a snadno použitelný průvodce vývojem softwaru. Protože se kniha soustřeďuje na konstrukci jako na nejznámější část životního cyklu softwaru, jsou popsané techniky výkonného vývoje softwaru srozumitelné pro samouky, stejně jako pro programátory s formálním vzděláním.

Technickým vedoucím

Mnozí techničtí vedoucí používali knihy řady *Code Complete* k výuce méně zkušených programátorů, které měli ve svých vývojových týmech. Knihu můžete využít rovněž k zacelení případných mezer ve vašem vzdělání. Jste-li zkušenými programátory, je možné, že s určitými závěry nebudete souhlasit (byl bych ostatně překvapen, kdyby tomu tak nebylo). Pokud se při čtení jednotlivých případů zamyslete, zřídka přijdete na řešení, o němž již dříve nebyla řeč.