Stručný obsah

Obsah

Předmluva

Jednou z oblíbených cukrovinek je u nás v Dánsku Ga-Jol, jejíž silné lékořicové aroma je výborným doplňkem našeho sychravého a mrazivého počasí. Kus šarmu těchto cukrovinek spočívá pro nás Dány také v moudrých a duchaplných rčeních, vytištěných v záhybu vrchní části každé krabičky. Dnes ráno jsem si koupil dvojité balení této lahůdky a našel jsem tam staré dánské pořekadlo:

Ærlighed i små ting er ikke nogen lille ting.

„Poctivost v malých věcech není malá věc." Bylo to dobré znamení, odpovídající tomu, co jsem chtěl říci již zde. Na malých věcech záleží. Tato kniha pojednává o prostých starostech, jejichž význam ale zdaleka malý není.

„Bůh spočívá v detailech," řekl architekt Ludwig mies van der Rohe. Tento citát připomíná současné argumenty ohledně role architektury ve vývoji softwaru a zvláště v oblasti, jako jsou „agilní techniky vývoje softwaru". Čas od času Bob i já zjišťujeme, že jsme do této bouřlivé debaty vtaženi. A také mies van der Rohe, který se zajímal o užitečnost a o nadčasové tvary staveb, což je základem dobré architektury. Na druhé straně také vybíral osobně každou kliku u dveří v domech, které navrhoval. Proč? Protože drobnosti jsou důležité. V naší pokračující „debatě" na téma vývoje řízeného testy jsme Bob i já zjistili, že souhlasíme s tím, že architektura softwaru hraje v jeho vývoji důležitou roli, ačkoliv máme různé představy o tom, co to vlastně znamená. Relativně není takové slovíčkaření důležité, protože můžeme přijmout jako samozřejmost fakt, že zodpovědní profesionálové věnují *nějaký* čas přemýšlení a plánování hned na počátku projektu. Pojetí návrhu, vedeného pouhými testy a kódem z pozdních devadesátých let, je dávno za námi. Ale pozornost věnovaná detailům je stále nepostradatelnějším základem profesionalismu, než je jakákoliv velkolepá vize. Za prvé je to praxe v malém, ve které profesionálové získávají zdatnost a důvěru pro praxi ve velkém. Za druhé, sebemenší kousek zanedbané konstrukce, dveří, které špatně doléhají, nebo křivá kachlička v podlaze, nebo dokonce jen nepořádek na stole dokáže zcela zmařit kouzlo celku. Právě o tom pojednává čistý kód.

Přesto je architektura jen jedním ze symbolů vývoje softwaru, což platí specificky pro tu jeho část, která vytváří počáteční *produkt* ve stejném smyslu, jako když architekt odevzdává bezchybnou budovu. V těchto dnech, patřících agilním metodám a metodám Scrum, je vše zaostřeno na rychlé uvedení *produktu* na trh. Chceme, aby továrna pracovala s maximální rychlostí a produkovala software. Tohle jsou lidské továrny: myslící a citliví kodéři, kteří zpracovávají nedostatky v produktu nebo sdělení uživatelů, na jejichž základě se má vytvořit *produkt*. V tomto pojetí se metafora výroby objevuje ještě silněji. Pohled na japonskou montážní automobilovou linku je velkou inspirací pro metodu Scrum.

Dokonce i v automobilovém průmyslu spočívá hlavní objem práce nikoliv ve výrobě, ale v údržbě – nebo v prevenci. V oblasti softwaru se 80 % toho, co děláme, nazývá „údržbou": opravováním. Místo abychom se typicky západním způsobem soustředili na *produkci* dobrého softwaru, bychom měli myslet více jako domácí opravář ve stavebnictví nebo automechanik v automobilovém průmyslu. Co nám k *tomu* může říci japonský management?

Kolem roku 1951 se v Japonsku objevil nový přístup ke kvalitě výroby, nazvaný „absolutně produktivní údržba" (Total Productive Maintenance – TPM). Zaměřuje se více na údržbu než na výrobu.

Jedním z hlavních pilířů TPM je několik tzv. zásad 5S. Zásady 5S jsou množinou pravidel – a zde používám termín „pravidlo" z důvodu snazšího pochopení. Zásady 5S tvoří ve skutečnosti základy metody *zeštíhlení* – další otravný slogan na západní scéně, který tvrdě proniká do softwarových kruhů. Tyto principy nejsou volbou. Jak ukazuje strýček Bob na stránkách úvodu, praktiky tvorby dobrého softwaru vyžadují tato pravidla: pozornost, zachování rozvahy a přemýšlení. Nejde jen o to uvést zařízení továrny do optimální výrobní rychlosti. Filozofie 5S v sobě zahrnuje tyto myšlenky:

- *Seiri* – neboli organizace: Zásadní věcí je vědět, kde se věci nacházejí – základem je například vhodné pojmenování. Myslíte si, že identifikátory proměnných nejsou důležité? Přečtěte si o tom v následujících kapitolách.
- *Seiton* – neboli pořádek: Existuje jedno staré americké rčení: *A place for everything, and everything in its place* – Mějme pro každou věc místo a vše na svém místě. Úsek kódu by měl být tam, kde jej budete hledat – a pokud ne, měli byste ho refaktorovat, aby tam byl.
- *Seiso* – neboli čistění: Nemějte na pracovišti zavěšené dráty, olej, kovový odpad a smetí. Co zde říkají autoři o zaneřádění kódu poznámkami a vykomentovanými řádky, které zachycují historii nebo přání do budoucna? Zbavte se jich.
- *Seiketsu* – neboli standardizace: Skupina souhlasí s tím, jak si udržovat pracovní místo čisté. Myslíte si, že tato kniha říká něco o konzistentním stylu kódování a sadě postupů v rámci skupiny? Kde se tyto standardy vzaly? Čtěte dále.
- *Shutsuke* – neboli disciplína. To znamená kázeň, dodržování postupů, věnovat často pozornost práci druhého a ochotu provádět změny.

Pokud přijmete výzvu – ano, výzvu – si přečíst tuto knihu a využívat ji, oceníte tento poslední argument a porozumíte mu. Zde se konečně dostáváme ke kořenům odpovědného profesionálního přístupu v povolání, které by mělo být spojeno s životním cyklem produktu. V průběhu údržby automobilů nebo jiných strojů metodou TPM je ošetřování havarijních stavů – kdy chyby vyplynou na povrch – výjimkou. Místo toho jsme se dostali na jinou úroveň: kontrolujeme stroje každý den a opravujeme opotřebované součásti ještě než budou mít poruchu, tedy provedeme ekvivalent příslovečné výměny oleje po 15 000 kilometrech, abychom zabránili opotřebení a zadření. V případě kódu refaktorujte bez slitování. Můžete se ještě o jednu úroveň zlepšit, tak, jak inovovalo hnutí TPM před padesáti lety: na prvním místě konstruujte stroje, které se snadno udržují. Čitelnost vašeho kódu je stejně důležitá jako skutečnost, že se kód bude provádět. Poslední pravidlo, které bylo v kruzích TPM zavedeno kolem roku 1960, je zaměřit se na zavádění zcela nových strojů nebo na výměnu starých. Jak nás napomíná Fred Brooks, možná, že bychom měli předělat hlavní části kódu od samého počátku přibližně každých sedm let, abychom vymetli špatně napsané partie. Možná bychom měli aktualizovat Brooksovu časovou konstantu řádově na týdny, dny nebo hodiny místo roků. V tom spočívá ten rozdíl.

V detailu spočívá mnoho síly, nicméně je v tomto přístupu k životu něco skromného a hlubokého, jak bychom mohli stereotypně očekávat od čehokoliv, co má původ v Japonsku. Ale nejedná se jen o východní přístup k životu. Lidová moudrost je plná podobných ponaučení. Citát v bodě Seiton vyšel z pera jednoho ministra státu Ohio, který doslovně viděl upravenost „jako lék pro špatnost každého stupně". A co Seiso? *Čistota je půl zdraví.* Při vší kráse, kterou budova oplývá, nepořádek v přijímací kanceláři jí bere veškerý třpyt. Co takhle Shutsuke v těchto malých záležitostech? *Kdo je věrný v malých věcech, je věrný i ve velkých.* Co žhavost po refaktorování v odpovídajících časech, posilující pozici pro následující „velká" rozhodnutí, raději než jejich odklad? *Jeden včasný steh ušetří devět dalších. Ranní ptáče dál doskáče. Co můžeš udělat dnes, neodkládej na zítřek.*. (To byl původní smysl

rčení „poslední důležitý okamžik“ v metodě zeštíhlení, dokud nepadlo do rukou softwarových konzultantů.) Co takhle nastavení prostoru pro malé, individuální snahy v rámci velkého celku? *Mohutný dub vyroste z malého žaludu.* Nebo co říkáte integraci jednoduché preventivní práce do každodenního života? *Gram prevence je lepší než tuna léků. S jedním jablkem denně nepotřebujete doktora.* Čistý kód spolu s pozorností vůči detailu ctí hluboké kořeny moudrosti, které se skrývají v naší obecné kultuře, nebo v její původní podobě nebo v podobě, v jaké by měla a *může* být.

Dokonce i v literatuře o velké architektuře můžeme nalézt rčení, která vycházejí z těchto domnělých detailů. Přemýšlejte o klikách miese van der Roheho. To je *seiri.* Znamená to věnovat pozornost každému jménu proměnné. Proměnnou byste měli pojmenovat se stejnou pečlivostí, s jakou dáváte jméno prvorozenému dítěti.

Jak jistě ví každý vlastník domu, podobná starost a neustálé vylepšování nikdy nekončí. Architekt Christopher Alexander – otec vzorů a jazyků vzorů – vidí v každém kroku vlastního návrhu malý, lokální čin opravy. A v dovednosti udělat dobrou stavbu vidí výhradně kompetenci architekta. Větší formy mohou být ponechány vzorům a jejich využití obyvatelům. Návrh je nekonečný nejen tím, že do domu přidáváme další pokoje, ale i tím, že věnujeme pozornost jejich vymalování, výměně opotřebovaných koberců nebo modernizaci kuchyňského dřezu. Většina dovedností vede k podobným názorům. V našem pátrání po jiných lidech, kteří věří, že boží dům se nachází v detailech, se ocitáme v dobré společnosti francouzského autora z 19. stolení Gustava Flauberta. Francouzský básník Paul Valery nám radí, že báseň není nikdy hotová a je třeba ji neustále předělávat. Když s tím přestanete, je to, jako kdybyste se jí zřekli. Takové zaujetí pro detaily je společné všem snahám o dokonalost. Možná, že tu najdete něco nového, ale možná že se během čtení této knihy rozhodnete přijmout pravidla, nad kterými již dávno zvítězila apatie nebo sklon k živelnosti, a prostě „zareagujete na změnu“.

Bohužel, podobné snahy běžně nechápeme jako základní kameny umění programovat. Kód brzy opouštíme ne proto, že je hotový, ale proto, že se náš hodnotový systém zaměřuje více na vnější vzhled než na podstatu toho, co poskytujeme. Cena za nedbalost se ukáže až na závěr: *Pravda vždy vyplave.* Ani v průmyslu, ani ve vědeckých kruzích se nikdo nesnižuje na takovou úroveň, aby se staral o čistý kód. Během mé minulé práce v Bellových laboratořích v oddělení výzkumu produkce softwaru (Software Production Research – skutečně produkce, výroba!) jsme přišli na několik stručných propočtů, které naznačovaly, že konzistentní způsob odsazování textu byl statisticky jedním z nejvýznamnějších ukazatelů nízkého počtu chyb. Chceme, aby architektura nebo programovací jazyk nebo nějaký jiný vyšší pojem byly záležitostí kvality. Tak, jako lidé, jejichž profesionalismus má daleko do mistrovství nástrojů a grandiózních návrhových metod, i my cítíme pohaněni důležitostí, kterou sem vnášejí tovární stroje, kodéři, za pomocí konzistentního používání odsazování. Abych citoval svou vlastní knihu, vydanou před 17 lety, takový styl odlišuje dokonalost od pouhé obratnosti. Japonský pohled na svět chápe zásadní význam každodenního pracovníka a k tomu i význam systému vývoje, jímž jsme zavázáni běžné a každodenní činnosti těchto pracovníků. Kvalita je výsledkem milionu obětavých kroků péče – nejen nějaké grandiózní metody, která sestoupila z nebe. Že jsou tyto kroky jednoduché, to ještě neznamená, že jsou zjednodušující, a už vůbec ne, že by byly snadné. Nicméně jsou základní kostrou dokonalosti a také krásy v rámci každého lidského snažení. Ignorovat je znamená nebýt plně lidským.

Samozřejmě že jsem stále obhájcem myšlení v širším slova smyslu a zvláště obhájcem hodnoty architektonického přístupu, hluboce zakořeněného v oblasti znalostí a softwarové upotřebitelnosti. O tom tato kniha není – alespoň ne zjevně. Tato kniha obsahuje delikátnější sdělení, jehož hloubka by neměla být podceněna. Odpovídá aktuálnímu současnému rčení lidí, jako jsou Peter Sommerlad, Kevlin

Henney a Giovanni Asproni, kteří se kódu věnují na plný úvazek: Jejich mantrou je „kód je návrhem" a „jednoduchý kód". Zatímco my musíme mít na paměti, že rozhraní je program a že jeho struktury mají co říci k jeho struktuře, zásadní je natrvalo přijmout jednoduchý postoj, že návrh žije v kódu dále. A zatímco předělávání v přeneseném smyslu výroby vede k nákladům, předělávání v oblasti návrhu vede k vyšším hodnotám. Na náš kód bychom měli pohlížet jako na skvělé vyjádření ušlechtilého úsilí vytvořit návrh – návrh jako proces, ne jako statický cíl. Je to kód, kde se uplatní metrika vazeb a soudržnosti v architektuře. Pokud budete poslouchat, jak Larry Constantine popisuje vazby a soudržnost, mluví v termínech kódu – ne v abstraktních a velkolepých koncepcích, které lze nalézt v UML. Richard Gabriel nám ve své eseji „Abstraction Descant" (chvála abstrakce) tvrdí, že abstrakce pochází od ďábla. Kód je proti ďáblovi a čistý kód je snad božský.

Když se vrátím zpět ke své malé krabičce Ga-Jol, myslím, že je důležité si všimnout, že dánská moudrost nám radí, abychom věnovali pozornost nejen malým věcem, ale také abychom v malých věcech byli poctiví. To znamená být poctivý ke kódu, poctivý ke svým kolegům ohledně stavu kódu a především být poctivý v otázce kódu vůči sobě. Udělali jsme *to nejlepší*, abychom „zanechali tábořiště čistější, než bylo předtím"? Refaktorovali jsme svůj kód před začátkem práce? To nejsou okrajové záležitosti, ale věci, které leží přímo v ohnisku hodnot agilních metod. Doporučený postup v metodě Scrum je, aby refaktorování bylo součástí koncepce fáze „hotovo". Ani architektura, ani čistý kód netrvá na perfektnosti, ale jen na poctivosti a nejkvalitnější práci, které jsme schopni. *Chybovat je lidské, odpouštět božské.* V rámci metody Scrum děláme vše viditelné. Větráme naše špinavé prádlo. Jsme poctiví ohledně stavu našeho kódu, protože ten není nikdy perfektní. Staneme se lidštějšími, hodnotnějšími v dokonalosti a staneme blíže této velikosti v detailech.

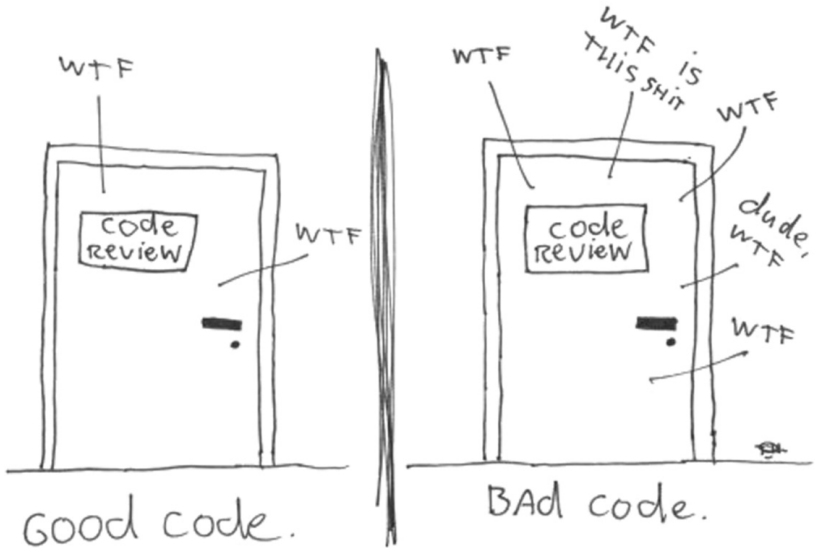
V naší profesi zoufale potřebujeme veškerou pomoc, kterou můžeme získat. Pokud čistá podlaha sníží v obchodě počet úrazů a pokud dobře zorganizované pracovní prostředky zvýší v dílně produktivitu, pak jsem pro ně. Pokud jde o tuto knihu, jde o nejlepší praktickou aplikaci zásad „zeštíhlení" v oblasti softwaru, kterou jsem kdy viděl v tisku. Od této malé a praktické skupiny přemýšlejících intelektuálů, která po léta spolupracovala nejen na tom, aby se zlepšili, ale také aby obdarovali svými vědomostmi lidi z průmyslové sféry dílem, které právě držíte v rukách, jsem nečekal nic menšího. Svět to dělá o něco lepším než jsem jej viděl dříve, dokud mi strýček Bob neposlal svůj rukopis. Když jsem skončil tohle cvičení v ušlechtilém pronikání do podstaty věci, mohu odejít, abych si uklidil svůj psací stůl.

James O. Coplien

Mørdrup, Denmark

Úvod

Jediná skutečná míra kvality kódu: Počet průšvihů za minutu



Reprodukováno s laskavým svolením Thoma Holwerdy. http://www.osnews.com/story/19266/WTFs_m
(c) 2008 Focus Shift

Do kterých dveří patří váš kód? Kde se nachází váš tým a vaše společnost? Proč jsme v této místnosti? Jde o obyčejné přezkoumání kódu nebo jsme hned po startu nalezli moře příšerných problémů? Ladíme v panice a zíráme do kódu, o kterém jsme si mysleli, že funguje? Opouštějí nás hromadně zákazníci a manažeři nám dýchají na záda? Jak si můžeme být jisti, že skončíme za pravými dveřmi, když postup začíná být obtížný? Odpověď zní: *mistrovská práce*.

Cesta k mistrovství má dvě části: znalosti a práci. Musíte získat znalosti principů, vzorů, postupů a heuristiky, kterou mistr v oboru ovládá, a musíte tuto znalost vstřebat do svých prstů, očí a do své osobnosti tím, že budete tvrdě pracovat a cvičit.

Mohu vás naučit fyziku řízení bicyklu. Opravdu, klasická matematika je relativně jednoduchá. Přitažlivost, tření, moment hybnosti, těžiště atd. lze všechno ukázat na méně než jedné stránce popsané rovnicemi. S těmito vzorečky mohu dokázat, že jízda na kole je praktická, a poskytnout vám veškerou znalost, kterou potřebujete, abyste jezdili. A přesto spadnete na zem ihned, jakmile si na kolo sednete.

Psaní kódu je úplně stejné. Mohli bychom sepsat všechny zásady správného postupu při psaní čistého kódu a pak vám tuto práci svěřit (jinými slovy, nechat vás spadnout z kola ihned, jakmile na něj nasednete), ale jaké by to z nás dělalo učitele a jaké žáky z vás?

Ne. Takhle tato kniha postavena není.

Učit se psát čistý kód je *dřina*. Vyžaduje to víc než jen znalost principů a vzorů. Musíte se nad tím *potit*. Musíte to sami procvičovat a sledovat své neúspěchy. Musíte sledovat pokusy ostatních i jejich neúspěchy. Musíte je vidět klopýtat a vracet se zpět. Musíte je vidět lámat si hlavu nad svými rozhodnutími a vidět cenu, kterou platí za rozhodnutí, jež jsou chybná.

Buďte připraveni na to, že čtení této knihy bude dřina. Tohle není kniha pro dobrou pohodu, kterou můžete číst na palubě letadla a již skončíte před přistáním. Tato kniha vás přinutí pracovat, *a to tvrdě*. Jaký druh práce budete vykonávat? Budete číst kód – mnoho kódu. A budete vedeni k přemýšlení, co je na tomto kódu dobrého a co špatného. Budete žádáni, abyste sledovali, jak jednotlivé moduly rozebíráme a pak je opět dáváme dohromady. To zabere čas i úsilí, ale myslíme si, že se to vyplatí.

Rozdělili jsme tuto knihu na tři části. Několik prvních kapitol popisuje principy, vzory a postupy, jak psát čistý kód. V těchto kapitolách je poměrně dost kódu, který vás bude lákat ke čtení. Zde budete připraveni pro druhou část, která bude následovat. Pokud tuto knihu po přečtení první části odložíte, pak hodně štěstí!

Druhá část knihy je těžší. Skládá se z několika rozborů problémů, jejichž složitost postupně narůstá. Každý rozbor je cvičením v čistění nějakého kódu – transformování problematického kódu do jiného tvaru, který tolik problémů nemá. V této sekci se intenzivně věnujeme detailům. Budete muset listovat sem a tam mezi doprovodným textem a výpisy kódu. Budete muset analyzovat kód, se kterým pracujeme, porozumět mu a projít si naši argumentaci pro každou změnu, již provedeme. Rezervujte si na to nějaký čas, protože *tohle by vám mělo trvat několik dnů.*

Třetí část knihy je za odměnu. Je to jediná kapitola obsahující seznam heuristik a intuitivních postupů, sebraných v průběhu tvorby našich rozborů. Když jsme tyto případy procházeli a čistili jejich kód, zdokumentovali jsme veškeré důvody svých zásahů buď jako heuristiku nebo intuitivní postup. Pokusili jsme se porozumět svým vlastním reakcím na kód, který jsme četli a opravovali, tvrdě jsme pracovali a snažili se podchytit, proč jsme se cítili právě takto a proč jsme dělali to, co jsme dělali. Výsledkem je databáze znalostí, která popisuje způsob našeho myšlení při psaní, čtení a čistění kódu.

Tato databáze znalostí má omezenou hodnotu, pokud si nedáte práci a nebudete pečlivě studovat rozbory ve druhé části této knihy. V těchto studiích jsme pečlivě popsali každou změnu, kterou jsme provedli, pomocí dopředných odkazů na heuristiku. Tyto odkazy jsou umístěny v hranatých závorkách, např. [H22]. To vám umožní vidět *souvislosti*, jak byly tyto heuristiky použity a napsány. Nejsou to samotné heuristiky, které jsou tak cenné, ale *souvislost mezi těmito heuristikami a samostatnými rozhodnutími, jež jsme během čistění kódu v našich rozborech udělali.*

Abychom vám s těmito souvislostmi pomohli, umístili jsme na konec knihy křížové odkazy, které pro každý dopředný odkaz uvádějí příslušné číslo stránky. Takto lze najít všechna místa, kde byla nějaká konkrétní heuristika použita.

Pokud si přečtete první a třetí sekci a přeskočíte rozbory problémů, pak jste četli další „oddechovou" knihu o psaní dobrého softwaru. Ale pokud věnujete čas a propracujete se těmito studiemi krok za krokem a s každým krátkodobým rozhodnutím, pokud to vezmete z našeho pohledu a přinutíte

se přemýšlet tím stejným způsobem jako my, získáte mnohem bohatší pochopení těchto principů, modelů, postupů a heuristik. Pak už to nikdy nebude nějaká „oddechová“ znalost. Bude to něco, co jste dostali do krve, do prstů a do srdce. Bude to něco, co se stalo vaší součástí stejným způsobem, jako se kolo stane součástí vašeho já, jakmile zvládnete jízdu na něm.

Poděkování

Výtvarné dílo

Děkuji mým dvěma umělkyním, Jeniffer Kohnke a Angele Brooks. Jennifer se zasloužila o senzační a originální obrázky na začátku každé kapitoly a rovněž o portréty lidí, jako jsou Kent Beck, Ward Cunningham, Bjarne Stroustrup, Ron Jeffries, Grady Booch, Dave Thomas, Michael Feathers a já.

Angela se zasloužila o znamenité obrázky, které zdobí vnitřní části každé z kapitol. Také pro mě v několika minulých letech namalovala nemalé množství obrázků, včetně těch, které se nacházejí uvnitř knihy „*Agile Software Development: Principles, Patterns, and Practices*“. Je také mým prvorozeným dítětem, ze kterého mám velkou radost.

Poznámka redakce českého vydání

Nakladatelství Computer Press, které pro vás tuto knihu přeložilo, stojí o zpětnou vazbu a bude na vaše podněty a dotazy reagovat. Můžete se obrátit na následující adresy:

Computer Press
redakce PC literatury
Holandská 8
639 00 Brno

nebo

knihy@cpress.cz

Další informace a případné opravy českého vydání knihy najdete na internetové adrese *http://knihy.cpress.cz/k1646*. Prostřednictvím uvedené adresy můžete též naší redakci zaslat komentář nebo dotaz týkající se knihy. Na vaše reakce se srdečně těšíme.