

## Slovo do pralice: Šest českých hlav reaguje

## Článek »Programátoři u pásu a na verpánku« vzbudil pozornost

Přinášíme ohlasy na příspěvek Alexandry Makovičkové, publikovaný v příloze Informační a komunikační technologie, která vyšla loni 19. 10. v Hospodářských novinách a 20. 10. v týdeníku Ekonom č. 42 (původní autorčin text lze najít na [www.makovickova.com](http://www.makovickova.com)).

**IVAN PILNÝ**  
nezávislý konzultant

Článek představuje zajímavou sumarizaci vývoje softwaru. Autorka se evidentně touto problematikou zabývá a souhrnu asi nelze věcně nic vytknout. Nepřekvapuje mne ani, že sumarizace je technologicky orientovaná, je pohledem zevnitř dovnitř. Tenhle pohled je u nás, počítačových odborníků, běžný. Maximálně konstatujeme obvyklá klišé: zákazníci nevědí, co chtějí, a občas poctivě přiznáme, že produkt jim dodáme později a draž. Nad rámeček tohoto obvyklého schématu jde poslední odstavec, který volá, aby rozhodovací pravomoci v moderních metodách řízení převzal ten, kdo je klientovi nejbližší, tj. analytik a programátor v jedné osobě. Když pomineme odvážnost a zkušenostmi ověřenou zřejmou nepravdivost tohoto tvrzení, obrací nám cosi naruby. Idea informačních technologií »per se« v roli zachránce společnosti je snad, doufám, už za námi. Nemám nic proti větší efektivitě a nižším nákladům na vývoj softwaru – v byznysu se tomu říká reengineering. Z byznysu už také víme, že na jeho úspěch tahle strategie zdaleka nestačí. Kdosi řekl, že »činnost některých manažerů připomíná přerovnávání křesel na palubě Titaniku«. Rozhodně si nemyslím, že softwarový průmysl je na tom jako Titanic, nicméně je užitečné připomenout si, odkud přišel a kam jde z poněkud jiného úhlu pohledu.

Poptávka po aplikacích historicky na začátku převyšovala možnosti technologií. Proto ty kreativní »housky« jednotlivých programátorů, šetřící každým bytem malé paměti. Spousta času potřebná na zvládnutí jazyka a obtížné kódování v primitivních jazycích předurčovalo téměř nemožnou údržbu a spoustu chyb. V téhle fázi software osvobozoval lidi od »opičích« činností, jako jsou spousty aritmetických operací, tvorby grafů a podobně. Cenou byl nízký uživatelský komfort a také z dnešního pohledu astronomické náklady a mizivá ochrana investic.

Technologie ovšem brzo dohnaly a předešly možnosti lidí v této oblasti. Často citovaný Moorův zákon o dvojnásobení rychlosti procesoru každých osmnáct měsíců platí. Jen se, doufám, už nepoužívá jako prodejní argument snažit se vnutit technologie zákazníkům. Softwarový průmysl se také snažil, nástroje přispívající k je-

ho větší efektivitě jsou v článku paní Makovičkové popsány. Co se podle mého názoru příliš nedaří, je přesvědčit zákazníky o rozumném poměru mezi přidanou hodnotou a cenou za aplikaci. Platí to i na českém trhu, přestože české firmy a Češi vůbec v poměru k ekonomickým indikátorům utrácí za informační technologie vysoce nadprůměrně. Iluze, že do informačního průmyslu jde více peněz, vznikají z toho, že mnohé, zejména velké počítačové firmy změny své zákaznické v oběti. Podnikatelská činnost i funkce státu v některých oblastech je bez počítačů už nepředstavitelná. A to je ideální past. Vydaje na další rozvoj se často změnilo ve výdaje mandatorní. V tom »vláží« vize typu »information anywhere, anytime to anybody«. Takové strategie občas skutečně někoho přimějí utratit jisté peníze. Ovšem jen do té doby, než zjistí, že lidé obvykle potřebují informace spíše »on demand« a v okamžité použitelné formě. »Kamenné« školství také zatím produkuje nové pracovníky spíše pro jinou planetu, takže se dovzdělávají praxí a zkušenostmi.

Jsem optimista a vidím, že popsaná realita už vede řadu počítačových firem ke změně strategie. Přestávají se orientovat jen na zvyšování efektivitu své výroby, ale přemýšlejí o skutečných potřebách svých zákazníků. Snaží se vytvářet přidanou hodnotu, která respektuje lidské zvyklosti a nemění je ve prospěch technologie. Největším problémem, naznačeným i ve výše zmíněném článku, jsou aplikace pro střední a malé firmy. Velké počítačové firmy se snaží o »downsizing« aplikací pro velké zákazníky, jiné postupně snižují počty nových »možností« v softwaru a snaží se »polidštit« tu důležité. Většina úspěšných aplikací, například CRM, vzniká stažením existujících aplikací z polic a doplněním interface. Počítačový průmysl stojí také před novými výzvami, dříve neexistujícími, které bytostně ohrožují jeho existenci. Nejde už zdaleka jen o pirátství, které přinesl vývoj vícenásobně použitelného softwaru, ale jedná se především o bezpečnost. Ochrana před »počítačovým terorisem« stojí nejen stále více peněz, ale směřuje zatím zřetelně proti uživatelskému komfortu. Jak by tedy měl vypadat »výrobní pás« dnešního aplikačního softwaru? Na začátku by měl být designér vytvářející hodnotu, pokrývající potřeby a respektující lidské chování. ▶



► Pak někdo, kdo efektivně zakóduje. Dále někdo, kdo otestuje. Nejen za účelem odstranění chyb, ale také kvůli splnění zadaných kritérií. Výsledný systém by měl být dynamický, tedy schopný dalšího vývoje podle potřeb zákazníka, aby ochránil jeho investici. Zbývá doufat, že zákazník, kteří takové aplikace dostávají, přibývá. Jen to zajistí optimální vývoj počítačového průmyslu, ale hlavně spoluplytí efektivnější ekonomiku a dodá prostor pro obohacení dalších lidských činností – zábavy, vzdělání... V počítačovém průmyslu se to jen hemží chytrými a vzdělanými lidmi, takže vše nemůže dopadnout jinak než dobře.

#### MARTIN KÁKONA ICZ a. s.

Rád bych reagoval na dva následující autorčiny výroky: »První generaci programovacích jazyků charakterizoval binární kód a velká náchylnost k vysokému procentu chyb. Jednoduchý algoritmus se zde měnil v sáhodlouhý kód.« K tomu podotýkám, že se zde zřejmě myslí strojový kód a ne binární. Dále se spíše myslí sáhodlouhý kódování a ne kód. Strojový kód u CISC procesorů je velmi efektivní pro zápis algoritmu. Výhody vyšších programovacích jazyků vyniknou v efektivitě zápisu teprve u RISC procesorů.

»Co se ale drasticky mění, jsou technologie a nástroje, pomocí nichž se jazyk dále vyvíjí.« V této souvislosti mě občas přepadá úzkost, ze-

jména když k nám přijde nějaký uchazeč o práci na pozici programátora. Skoro si myslím, že až vymře naše generace, zapomene se, jak vlastně funguje počítač a jak se programuje překladač. Možná, že to lidstvo bude muset zase znovu objevovat, pokud se ta informace neuchová někde v koutku Číny nebo Indie.

#### JIŘÍ KOUTNÍK IT Project Manager, Komerční banka, a. s.

Článek dobře shrnuje rozvoj technologií i metodologií vývoje softwaru a podtrhuje i problémy, se kterými se vývoj softwaru potýká. Chybí mi ovšem vlastní vize autorky, jak dané problémy řešit a kudy se vývoj bude ubírat. Čekal bych hlubší zamyšlení, jak budou nové (agilní) přístupy zohledněny v nabídce služeb dodavatelů a co naopak by autorka očekávala od zákazníků.

Veškeré změny ve výrobě softwaru, ať už v technologiích (cesta od binárního k objektovému a aspektovému programování) nebo v metodologiích (od vodopádu k agilním metodám), sledují a adresují jedině – potřeby trhu a vztah dodavatele se zákazníkem. Software je žádaná komodita a na rozdíl od počátků softwarového vývoje v minulém století podporuje v dnešní době téměř všechny činnosti. Zkracování doby uvádění výrobků a služeb na trh je výraznou konkurenční výhodou. Tlak na rychlost vývoje softwaru, který dané výrobky a služby podporu-

je, je proto několikanásobně větší. To vede na jedné straně k potřebě rozbití zavedené »pásově« orientované výroby (waterfall model) a k přechodu k silně paralelnímu a inkrementálnímu vývoji s vysokou mírou přizpůsobivosti podle měnících se potřeb zákazníka (agilní metody) a na druhé straně k »industrializaci« vlastního programování.

Bohužel připravenost dodavatelů i zákazníků, jak autorka správně připomíná, je na takový způsob vývoje zatím velmi malá. Služby, které drtivá většina dodavatelů nabízí, jsou obvykle striktně založeny na kontrakčních dodávkách více či méně monolitických aplikací a systémů a vývoj se pak silně soustředí na jejich výrobu a akceptaci, nebo je zákazníkovi nabízena lidská kapacita po jednotlivých činnostech (jako na pásu). Jen velmi málo dodavatelů je ochotno dodávat a vázat odměnu (vzít na sebe riziko) za své dodávky na spokojenost koncového zákazníka, což je mimořádně hlavním heslem agilních metodologií.

To se týká jak velkých, tak malých dodávek. Stále bohužel většinou panuje přesvědčení, že zákazník je schopen od nejmenších podrobností specifikovat zadání, které nebude v čase měnit, a když, tak za cenu vysoké penalizace (zajištění dodavatele). Zákazník pak (obvykle mylně) očekává, že na základě tohoto zadání po několika týdnech, měsících nebo i letech dostane s minimálním přispěním přesně ten produkt, který očekával, a svazuje (k zajištění svých investic) dodavatele sítí časových, funkčních a kvalitativních omezení pod hrozbou sankcí.

Realita je pochopitelně odlišná. Jen málokdo je schopen přesně specifikovat, co chce, v okamžiku, kdy jde nakupovat (to platí i pro mnohem hmatatelnější výrobky, než je software). Svět se navíc velmi dynamicky mění a jen máloco je stabilní více než několik dnů. Představa, že bez aktivní spolupráce na vývoji dostane klient přesně produkt, jaký si tiše představoval, je zcela mylná. Heslem by mělo být: flexibilita softwaru, spokojenost uživatele a aktivní přístup ze strany zákazníka. Jen ti dodavatelé a zákazníci, kteří si uvědomují nutnost těsné týmové spolupráce, připravenosti na změnu a plnou orientaci na potřeby koncového uživatele, budou nakonec spokojeni a plně využijí výhod, které moderní softwarové technologie přináší. Je potěšitelné, že uvedené poznání není jen na straně zákazníků, ale, jak je z článku patrné, i dodavatelů. Nezbyvá, než se těšit, že se brzy odrazí i na trhu, v poptávce a nabídce služeb.



**DAN TOTUŠEK**  
KOMPASS Czech Republic, a.s.

Rád bych se vyjádřil za stranu »retail«. Ideálem obou stran, tedy klienta a dodavatele, jsou rozhodně schopní lidé. Je proto úplně jedno, zda software vyvíjí »obývací one man company« nebo velký softwarehouse. Pokud totiž konkrétní »one man« v obýváku dokáže pochopit, co po něm klient chce, následně vše převést do praxe a zdokumentovat tak, aby mohl být případně nahrazen jiným »one manem«, pak mohou být všichni spokojeni. Takový člověk musí nezbytně být i ve velkém softwarehouse, jinak se s klientem dodavatel nikdy uspokojivě nedohodne. Pro programátory je často nejobořlivější si přiznat, že jsou dělníky, kteří vyrábí produkt podle zadání, jež má vytvářet co možná nejmenší počet lidí na straně dodavatele a vývojáře, což představuje další »konflikt«. Jde totiž o to, najít tak dobrého programátora, který napíše takřka vše, svou práci umí po sobě alespoň rámcově zkontrolovat a zároveň nemá ambice prosazovat vlastní »vylepšení«. K poslední větě v článku Alexandry Makovičkové bych rád dodal, že analytik a programátor v jedné osobě, splňující výše popsané, je prostě k nezaplacení.

**DANIEL KUTÁČ**  
Sales Engineer InterSystems B.V. ČR

Třebaže autorka zevrubně popisuje různé metody tvorby programů a různá programovací prostředí, je podle mého názoru ale opomenut fakt, že většina dnešních aplikací je založena na použití databáze pro ukládání dat a nad ní umístěné aplikační vrstvy. Tato klasická vícevrstvá architektura již sama o sobě ovšem představuje zdroj problémů bez ohledu na to, jakou technologii vývojáři použijí pro vývoj aplikační logiky (objektově orientované prostředky, tj. Javu či .Net nebo něco jiného). Pro přístup k databázím pak musí »přehodit mentální výhybku« a přejít do SQL, tedy do světa relačních vztahů. (V mnoha textech lze narazit na pojem »impedanční nesoulad«, který vystihuje rozdíl mezi objektovým a relačním pohledem na svět.) Makovičková tento aspekt vývoje softwaru nezmiňuje, pravděpodobně proto, že ho považuje za

samozřejmý. Nicméně tak tomu být nemusí. Pomocí některých technologií, například postrelační databázové platformy Caché, lze docílit toho, že obě zmíněné vrstvy lze »mentálně« (a třeba i fyzicky) spojit do jedné. Zjednoduší se tak celý vývojový cyklus, protože vše, co analytik navrhne, může poměrně jednoduše přímo přetvořit do hotového kódu (pomocí průvodců atd.). Nemusí se tedy spoléhat na někoho dalšího znalého SQL, »pouhého« programátora, jak pochopí či nepochopí jeho analýzu. V tomto případě hovoří programátor stejným jazykem jako analytik – objektově. Výhodou použití postrelační platformy, která podporuje objekty i SQL, je zpravidla kratší potřebná doba pro napsání aplikace, a tím i nižší náklady. Díky úspoře v délce kódu, která může být i v řádu desítek procent, se snižuje i riziko ztráty integrity, výkonu aplikace a výskytu chyb. Dalším možným místem úrazu je, nakořlí analytik pochopí požadavky zákazníka. Nicméně i zde prokazují objektové technologie svoje přednosti, protože se více blíží reálnému světu. Zákazník hovoří přirozeným jazykem, který má bezprostřední odezvu v objektech, a analytik pak nemusí přemýšlet, jak zákaznickovy představy přetvořit do relačního modelu. Je nutné zdůraznit, že využití postrelační architektury při vývoji aplikací nikterak nenarušuje různé metodiky vývoje softwaru. Naopak – je s nimi v souladu a ještě více je zefektivňuje.

**PATRIK PLHOŇ**  
konzultant ve společnosti Et netera

V souvislosti s článkem bych chtěl upozornit na pochopitelný trend zaměřit se na zákaznický přizpůsobený, respektive přizpůsobitelný vývoj aplikací (systémů) »na klíč«, tzn. k obrazu klienta, oproti dříve upřednostňované hromadné velkosériové výrobě »krabicových« produktů. Malé a střední zákaznický ze zmíněného tržního segmentu maloobchodu (retail) dnes už opravdu obvykle nezajímá, jak bude řešení postaveno, ale v jakém rozsahu pokryje jejich požadavky na funkcionalitu, termín dodání a návratnost investice. Tlak na maximální efektivnost a hospodárnost je možná nejvíce patrný právě v oblasti

pořizování infrastruktury informačních a komunikačních technologií.

Rovněž bych rád poukázal na potřebu využít standardní postupy (např. prostřednictvím v článku rovněž zmíněných webových služeb). Cílem je tady jak propojitelnost stávajících aplikací a systémů podniku s těmi nově nasazovanými, tak také rozšířitelnost řešení (jeho další rozvoj). Při současném, často hodně dynamickém a hektickém trendu růstu potřeby, a tím i vývoje aplikací může úspěšně fungovat jen taková společnost (dodavatel), která je schopna kromě kvalitního projektového řízení (správné rozdělení projektů na fáze i dílčí úkoly) a týmové spolupráce vhodných lidských zdrojů (na straně dodavatele i zákazníka) nabídnout buď velmi rychlý a dostatečně kvalitní vývoj drobných levných aplikací či z již vyvinutých (koupených) komponent stavět nová neotřelá komplexní řešení. Ve druhém jmenovaném případě může být znovu-použití naprogramovaných částí či celých aplikací velmi výhodné, ale v souvislosti s vývojem »na míru« ne vždy aplikovatelné. Efektivní může být dostupnost dostatečně robustní a škálovatelné stavebnicové technologické platformy.

V praxi se jako jeden z nejkritičtějších článků řetězce zákazník (uživatel) – dodavatel (projektový manažer – konzultant – analytik – programátor – tester) ukazuje právě testování, resp. vhodný model a simulace používání vyvíjené aplikace i související ošetřování výjimek (nestandardních stavů – např. při ovládní aplikace). Fatální může být také absence příslušné dokumentace v případech změny vývojáře.

Připravil ht

