Superprogramátoři... a ti druzí

Ladislav Dvořák

20:1

Každý z nás se již setkal s tím, že v produktivitě práce různých programátorů jsou značné rozdíly. Několik všeobecně známých studií uvádí, že poměr výkonnosti práce programátorů dosahuje až 20:1.

Prakticky ve všech uvedených studiích se jednalo o napsání a odladění malých „umělých" programů, takže je určitou otázkou, nakolik tyto studie odpovídají reálným poměrům, ale myslím si, že z praxe všichni dobře víme, jak velké rozdíly mezi programátory existují. Ačkoliv je tento fakt všeobecně známý, nenašel jsem nikde pokus o jeho rozumné vysvětlení.

(Jestli chcete, můžete si nyní zahrát hru: přestaňte číst a zkuste přijít na to, proč tomu tak je. Uvidíme, nakolik se budou naše poznatky shodovat.)

Domnívám se, že vysvětlení je následující: programátoři, podobně jako jiní řemeslníci, používají ke své práci nástroje.

V jejich případě se jedná o nástroje duševní povahy – různě silné a efektivní, a přitom „neviditelné". Programátoři si je vytvářejí sami a mohou proto vypadat velmi rozdílně.

Síla a zejména dostatečná paleta nástrojů (když vhodný nástroj neexistuje, musí se vytvořit a tady vznikají problémy) vede k rozdílům způsobujícím rozdíly ve výkonnosti. Jedním z cílů tohoto článku je proto „zviditelnit" nástroje, které při své práci programátoři používají.

Pokusme se o analýzu našeho problému z tohoto hlediska a zkusme stanovit základní oblasti, v nichž se uvedené nástroje nacházejí.

Základní oblasti programátorských dovedností

Práce programátora zahrnuje aktivity vztahující se k následujícím oblastem vědomostí:

a) znalost problematiky (tj. znalost reálného kontextu, který je analyzován a ve kterém mají vzniklé programy působit),

b) znalost způsobu řešení úloh (metodika a technologie programování). Tuto oblast můžeme dále rozdělit na dvě části:

1) řešení velkých úloh,

2) řešení úloh normální velikosti,

c) znalost programování (ovládání programovacího jazyka a počítače jako nástroje, pomocí něhož dokážeme napsat a odladit fungující program, resp.programový systém),

d) systémové znalosti (znalost počítače, operačního systému, způsobu organizace dat, jazyků a jejich různých vazeb, speciálních přístupů a triků).

Poznámka

Z určitého hlediska lze říci, že okruh znalostí uvedený pod bodem (c) je podmnožinou znalostí pod bodem (d). Z hlediska práce programátora je však mezi oběma oblastmi významný rozdíl. Oblast (c) zahrnuje to, co programátor používá ke své každodenní práci, zatímco znalosti z oblasti (d) zůstávají spíše v pozadí jeho rozhodnutí pro volbu určitého postupu nebo je pomocí nich řešen velmi úzký okruh nestandardních problémů (většinou zpracování speciálních rozhraní).

Pro jednoduchost jsem při dělení na jednotlivé oblasti použil slovo znalost. Ve skutečnosti jde pochopitelně současně i o schopnost své znalosti efektivně používat.

Konkrétní vědomosti a dovednosti z uvedených čtyř oblastí tvoří soubor duševních nástrojů, který má programátor k dispozici.

Současně se nám ukazuje poměrně podstatná skutečnost, že značná část práce programátorů, kterou bychom mohli nazvat *tvorba a modifikace duševních nástrojů*, je „skrytá". Jedná se právě o studium určitých znalostí a učení se jich rozumně používat.

Dovednost osvojovat si potřebné poznatky a umět je vhodně aplikovat bychom proto mohli také považovat za vlastnost, která významně ovlivňuje celkovou efektivnost programátora.

Z hlediska výkonnosti pak můžeme provést základní dělení programátorů na dvě skupiny: superprogramátoři a ti druzí.

Co umí superprogramátor

Obecná charakteristika:

- má nadprůměrné vědomosti v oblastech (c–d),
- programuje skoro strukturovaně (ovládá určité techniky z (b) a umí je používat bezpečně, rychle a přesně),
- obvykle zná dobře problematiku, v níž se pohybuje (a) – kontext, vazby, dynamický vývoj, a umí navrhnout kvalitní řešení problému,
- mívá období, kdy zdánlivě nic nedělá, jenom si hraje s počítačem a studuje nějaké hlouposti (to je období, kdy si „ostří" své duševní nástroje).

Výsledkem jeho práce je kvalitní analýza řešeného problému (dělá ji přes překážky, které jsou mu kladené) a vyvážený návrh řešení. Po krátké etapě „dobrušování pomocných nástrojů" (doplnění potřebných vědomostí a tvorbě podpůrných programových prostředků – například pro jednotný přístup k datům) poměrně rychle a s málo chybami program napíše a rychle odladí.

Program funguje ke všeobecné spokojenosti uživatelů a jeho doba životnosti je výrazně vyšší proti průměru.

Více o tomto zvláštním tvoru můžeme říci na základě rozlišení jeho jednotlivých odrůd.

Typy superprogramátorů

Podle záliby v oblastech (a–d) můžeme uvést následující typy superprogramátorů:

1) Velký analytik (má převahu znalostí v oblastech (a–b). Jeho znalosti z oblastí (c–d) převyšují znalosti průměrného programátora, ale v podstatě ho tato oblast nezajímá, a proto sám neprogramuje, pokud nemusí.

2) Velký manažer (těžiště jeho vědomostí leží v oblasti (b1) – řešení větších programových systémů – z ostatních oblastí zná to, co potřebuje.)

Pokud se věnuje své práci na 100% a má navíc hlubokou znalost oblasti (c) a trochu i (d), je ideálním typem pro vedení „chirurgického" týmu schopného podávat neuvěřitelné výkony.

Tento typ stojí v popředí zájmu amerických softwarových firem a příruček zabývajících se metodikou tvorby velkých programových systémů.

3) Ferda Mravenec, práce všeho druhu (umí všechno, má vyrovnané znalosti, otevřenost, schopnost komunikace (věnuje se své práci celé, ale nemá tak velkou ctižádost jako Velký manažer).

4) Osamělý vlk (má perfektní znalost problematiky, velmi kvalitní znalosti (c–d), ale má problémy se spoluprací a komunikací s druhými.

Je to typ ideální pro řešení složitých úloh, na které stačí jeden člověk.

5) Systémový programátor (hluboká znalost d) je specialista, který ví vše o nejrůznějších systémech. Dokáže se úspěšně ponořit do hlubin operačního systému, aby nalezl řešení úlohy speciálního typu. Normální programy jej nezajímají.

Jeho nevýhodou je, že se často baví s ostatními na „své úrovni" – když to přežene, je výsledek styku s ním stejný jako s níže popsaným „Doktorem Ovachem".

Je vidět, že se jedná o značně odlišné typy s různými možnostmi použití. Pro všechny však platí, že mohou být k práci motivováni nebo naopak být frustrováni. Většinou platí spíše druhý případ.

Časový stress vyplývající ze špatného odhadu doby potřebné na realizaci programu nebo programového systému (který je svým způsobem zákonitý), vede okolí superprogramátorů k vynalézavosti, pokud jde o hledání zkratkovitých řešení. Naprostá většina těchto řešení se ukáže jako blbost nebo, v lepším případě, jako něco, co značně komplikuje úspěšné řešení.

„Nepochopitelné" a nesplnitelné požadavky superprogramátora se po čase ukazují jako lepší řešení. Ale valící se sudy už nejde zastavit.

Superprogramátoři mají většinou pravdu, ale často ji nedokážou prosadit!

Ostatní programátoři

I ostatní programátory můžeme podle jejich znalostí rozdělit na jednotlivé typy. Spíše než o přesnou systematiku, kterou lze na základě definice základních oblastí snadno udělat, se pokusím o vystižení určitých výrazných a dobře známých typů.

1) Ženská od Cobolu (oblasti (a,b,d) ji nezajímají, v (c) má vědomosti, které jí umožní aby na základě přesného zadání jednoduchého programu v potu tváře napsala a odladila cosi složitého).

Ačkoliv má pocit, že je přetěžována, patří k nejméně efektivním programátorům a hodí se na řešení rutinních, jednoduchých úloh. (Tento typ je převážně doménou žen, ale najdou se i muži, které lze do této skupiny úspěšně zařadit.)

2) Malý odborník (ví něco o (a–b), ostatní zná spíše nedostatečně) navrhuje dokonalá, bohužel však nerealizovatelná řešení (možná vůbec nerealizovatelné a možná to jenom neumí).

To je situace, ve které se ocitá prakticky každý začátečník. Aby se z něj stal slušný programátor, musí prožít období, kdy se musí „podusit". Toto období trvá běžně kolem půlroku, někdo v něm však zůstane na celý život.

3) Doktor Cvach – teoretický znalec počítačového systému. Když má udělat analýzu jednoduchého problému, tak se ztratí.

4) Malý analytik (převaha jeho vědomostí je v oblasti (a), ostatní ho nebaví). Má zájem o styk s lidmi a o hledání řešení jejich problémů.

5) Malý manažer – umí zorganizovat práci pro více lidí, tak, aby vzniklo něco rozumného, a vyjít přitom s nimi.

6) Normální programátor (zná ode všeho něco a od ničeho moc). Umí průměrně analyzovat problémy – rozumně chápe oč jde (smysl a účel programu). Prakticky ovládá způsob, jak navrhnout rozumný a realistický program, a zná nepříliš rozsáhlou, ale přitom dostatečně velkou množinu operací s počítačem, aby dokázal program napsat a odladit.

7) Tvrdý programátor (jeho znalosti se vztahují převážně k (c) a trochu k (d)). Bere, co se mu přinese a řešení spolehlivě vysedí.

Užitečnost jednotlivých typů

K čemu je nám popsaná typologie dobrá?

Je důležité uvědomit si, s kým máme co do činění! Různé typy, kromě zcela nepoužitelných (ale i pro ně se možná najde nějaká práce – třeba v oblasti úzké specializace, nebo organizace dokumentace), lze využít poměrně efektivně v oblastech, které jim dobře vyhovují. Nemá cenu nutit je do něčeho, co jim nesedí.

I v případě, že nemáme možnost ovlivnit rozdělení práce, nám typologie může pomoci lépe pochopit, co od svých kolegů můžeme očekávat (ať už pracovně nebo i lidsky) a lépe se s tím vyrovnat.

Konečně – není špatné umět sám sebe zařadit.

Rozdíly ve výkonnosti

Vraťme se ještě jednou k otázce, kterou jsme se původně začali zabývat. Proč je ve výkonnosti programátorů takový rozdíl?

Pochopíme to, když se podíváme blíže na způsob práce Normálního programátora. Na rozdíl od frází, které se píší do kvalifikačních předpokladů různých platových stupnic („samostatně a iniciativně řeší náročné úlohy atd."), je realita poněkud jiná. Ve skutečnosti zná něco málo z analýzy problémů, jeho návrh řešení programu je na průměrné úrovni, program napíše s řadou chyb, s kterými dost těžce zápasí.

V případě, že narazí na něco neobvyklého mimo okruh svých vědomostí a pokouší se to sám vyřešit – zvládne to, ale stojí ho to hodně času. Jeho obzor to příliš nerozšíří – pokud nejde o zcela zásadní věc, tak na to zase zapomene. (Z toho plyne jedno poučení – nemá cenu příliš rozšiřovat vědomosti Normálních programátorů v oblasti (d), zato je třeba dbát o to, aby měli určité funkční minimum v oblasti (c).)

Komu připadá, že je uvedený způsob práce nedostatečný, je třeba připomenout, že Normální programátor je poměrně ideální pracovník a že existuje řada daleko horších typů.

Je třeba, abychom si zvykli vidět věci takové, jaké jsou. Realistické vidění vede k tomu, že si člověk uvědomí, co má vlastně od lidí požadovat a jakým způsobem jim může efektivně pomoci.

Zveřejnění duševních nástrojů

Víme-li o tom, že pracujeme duševními nástroji, je rozumné starat se o to, aby tyto nástroje byly – v rámci možností – co nejkvalitnější.

Protože jde o skryté nástroje, je užitečné zkoumat, jakými nástroji naši kolegové disponují, a učit se od nich. V případě, že jejich nástroje přesahují naše možnosti, je dobré o nich vědět, abychom své kolegy mohli v případě potřeby využít. Nejde o žádné para-

zitování, ale o smysluplnou spolupráci. Domnívám se, že je vhodné schopnosti jednotlivých pracovníků vhodným způsobem zpřístupnit a snažit se, aby to pro všechny strany bylo užitečné. (To pak umožní například i to, aby ten, kdo je lepší, byl za všeobecného souhlasu lépe placen.)

Z toho vyplývá i efektivnost různé „neproduktivní" práce, jako jsou konzultace, veřejná oponentura programů, přednášky a školení (i tady však existují omezení – za určitou hranicí informace ztrácejí význam, protože přijímající je nedokáže zpracovat).

To vše vyžaduje určité klima otevřenosti, které u nás minulých letech bylo prakticky nedosažitelné, a proto s podobným stylem práce máme málo zkušeností.

Jak pomoci jednotlivým typům

Práci superprogramátora můžeme zlepšit, když jej budeme efektivně využívat, tj. přidělíme mu práci příslušného typu, přeneseme na něj příslušný díl zodpovědnosti a také ho příslušně oceníme.

Práci ostatních programátorů zkvalitní, když se jim pomůže získat příslušné minimum vědomostí (a současně se jim položí nůž na krk, že budou muset odejít, když jej nebudou umět). To by mělo i těm slabším pomoci zařadit se do skupiny Normálních programátorů.

To je naprostý základ. Normálnímu programátorovi nejlépe pomůže, když bude mít snadný přístup k informacím, které jsou „za jeho obzorem", a když bude mít možnost snadno získat pomoc v nestandardních situacích.

Bude-li moci využít služeb specialistů, kteří pro něho udělají, co potřebuje, nebo mu aspoň poradí (práce se systémem, s knihovnami speciálních programů, pomoc při ladění, dokumentace, údržba), překonávání překážek při práci pro něj pak nebude tak stressující a časově náročnou záležitostí a bude moci pracovat daleko plynuleji a efektivněji. (Všimněte si, že to co by měl mít Normální programátor k dispozici, je vlastně jakási obdoba „chirurgického týmu" superprogramátora.)

Autor: RNDr. Ladislav Dvořák, CSc.
Ústav pro informatiku a vzdělávání
Gorkého nám. 26
P. O. BOX 562
111 21 Praha 1
tel. (02) 267 041 - 9, kl. 399