

Zdeněk M E Z N Í K prom. mat.

ZVT - OKR Ostrava

Organizace analytických a programátorských prací

Ve svém referátu se pokusím o částečné shrnutí více než desetiletých zkušeností pracovníků Závodu výpočetní techniky OKR a velmi rád se v diskusi podělím o zkušenosti další. Pro orientaci a úvod několik údajů:

ZVT zpracovává data pro VHJ OKR, které má cca 99 000 zaměstnanců v 27 národních podnicích. Zpracovávají se hlavně tyto agendy: mzdy a platy pro 66 000 zaměstnanců, účetnictví 350 000 položek měsíčně, MTZ 300 000 položek měsíčně, odbyt uhlí a koksu, statistika. Neuvádím řadu dalších neméně důležitých, ale objemem dat menších systémů.

Organizační členění závodu zahrnuje kromě jiných odborné útvary systémového inženýrství - cca 50 lidí a programování - cca 50 lidí. Práci těchto dvou útvarů se tento příspěvek bude zabývat.

Náš závod prošel celou historií vývoje výpočetní techniky. Od osmdesáti a devadesáti sloupcových děrnoštítkových systémů přes počítače ZUSE 223 a ICL 1905 a druhé počítačové generace až k počítačům EC 1040 a IBM 370/145, které představují špičku ve výpočetní technice u nás. Naši zaměstnanci měli tedy možnost projít vším, co tento, dnes již značně rozšířený, obor mohl nabídnout.

Nyní k tématu danému názvem příspěvku.

Na spolupráci analytiků a programátorů je řada názorů a vyskytují se oba dva možné způsoby. Tedy:

a) analytik a programátor pracují společně v jednom týmu - oddělení - odboru. Jejich pracovní náplně se značně prolínají, někdy jsou obě profese spojeny v jedné osobě, na jednom úkolu pracují společně od jeho zadání až po expedici výsledků, někdy si i obsluhují počítač.

Tento způsob práce je možné za určitých podmínek přijmout, avšak spíše výjimečně. Mělo by tomu tak být v případech, kdy jde o práce na malém počítači, tehdy, kdy existuje minimální počet pracovníků a práce na tomto počítači se teprve zavádí. Výpočetní středisko v této době nemá obvykle dostatek kvalifikovaných odborníků, musí si je teprve vychovat, což tento organizační přístup dovoluje. Analytiko-programátorské práce v této fázi jsou obvykle na kvalitativně nižší úrovni, což se částečně vyrovnává nadšením pracovníků pro novou výpočetní techniku. Avšak i ty nejlepší úmysly jsou málo platné, když neexistují standardní pracovní postupy, nejsou zkušenosti. Za současného a zřejmě i budoucího nedostatku pracovníků vůbec a obzvláště školených pracovníků v analytických a programátorských pracích nelze ovšem očekávat, že se tento stav změní.

Posuňme se ale na časové ose o kus dál. Pracovníci získají znalosti, zkušenosti, je jich víc, i když ne dostatek, utvoří si určité pracovní normy a standardizují si některé pracovní úkony. A právě nyní je nejvhodnější doba k tomu, aby došlo k nové organizaci a to

b) k rozdělení práce u počítače na část analytickou a část programátorskou a chápat příslušné pracovníky jako odborníky právě ve svém oboru. Tito pracovníci nemohou, pokud to nemá být na úkor jejich profesní zdatnosti, znát dobře analytický systém a k tomu ještě programovací jazyk, jazyk operačního systému, pracovní postupy atd. Toto vše

dohromady není v silách normálních průměrných zaměstnanců a tvrdím, že pokud se tento způsob práce používá, musí to být v naprosté většině případů práce neefektivní. Ještě na upřesnění - jde o práce u středního počítače, na kterém se zpracovávají hromadná data. Pokud je počítač využíván k vědeckotechnickým výpočtům nebo k řízení výroby, či k jiným speciálním pracím, jde o něco jiného, a tato, mnou uváděná tvrzení ve směr budou neplatná.

Dlouhodobá praxe ukazuje, že ten, kdo pracuje okolo výpočetní techniky (kromě pracovníků obsluhy, vstupní a výstupní kontroly), by měl začít jako programátor. Obzvláště to platí o analytících. Bohužel často jsem svědky toho, že analytickou přípravu systému provádí sice věcně odborně erudovaný pracovník, který však nemá ani vědomosti, ani zkušenosti s výpočetní technikou. Pro ilustraci předkládám tuto analogii: Jistě neexistuje projektant či konstruktér, produkující technickou dokumentaci pro výrobu, aniž by touto výrobou prošel, měl s ní zkušenosti a rozsáhlé vědomosti o ní. Vznikala by dokumentace pro stroje, podle níž by se nedalo pracovat. U výpočetní techniky, přesto, že jde o stroje značně nákladné (např. EC 1040 stojí 50-60 milionů Kčs), se tento stav přepouští. Dochází pak k velkým ztrátám na strojovém čase, na čase programátorů i analytiků. Tedy striktní požadavek na kvalifikaci analytiků by měl být vždy plněn. Za tohoto předpokladu není již žádný argument proti rozdělení profesí na analytika a programátora a to nejen profesní rozdělení ale i organizační. V našich podmínkách se zcela osvědčilo organizační rozdělení - odborný útvar systémového inženýrství, odborný útvar programování.

Práce v těchto odborech je organizována v týmech. Programování je rozděleno na týmy dle zpracovávaných agend. V čele týmu je vedoucí - vysokoškolák s nejméně pětiletou praxí, jako druhý stabilní člen týmu zástupce vedoucího. Ostatní členové týmu jsou přidělováni dle momentální potřeby a možností. Tento organizační dochází k maximálnímu využití

lidí a dosahuje se maximální produkce.

A jak vypadá zpracování v týmech?

Vedoucí týmu obdrží od příslušného vedoucího analytického týmu programovou specifikaci, kterou s ním předběžně prodjednal. Ještě dříve však byl celý programový systém oponován pracovníky provozu a programování. Vedoucí týmu na základě obtížnosti programu a v závislosti na kvalitě programátorů, kteří jsou k dispozici, předá tuto specifikaci určitému členmu týmu. Tento o systému, jehož součástí je daný program, nemusí nic vědět. Ve specifikaci jsou jasně a jednoznačně popsány vstupy, výstupy a funkce programu. Programátor pak již použije metodu normalizovaného programování na vývojový diagram, dále daný programovací jazyk - který je pro všechny jednotný (u nás je to COBOL - ale o tomto se zmíním ještě dále), jazyk operačního systému a strojový čas na ladění. Ladění probíhá nejprve na vlastních datech - nečeká se, až jsou data vytvořena předchozím programy v systému, k tomu se použije program obecného užití pro generování ladících dat. Tento program na základě jednoduchých parametrů vygeneruje požadovaný soubor tříděný dle určených hledisek. Výsledky ladění pak jsou konzultovány s příslušným analytikem. S tímto se spolupracuje i při odsouhlasování výsledků při ladění systému, které se provádí na datech již dodaných analytikem.

Hovořil jsem o normalizovaném programování. V loňském roce byla podána na tomto fóru rozsáhlá informace o tomto tématu. Nicméně bych se stručně zmínil o hlavních výhodách, které nám tato metoda přinesla. Od poznání existence normalizovaného programování z časopisu MAA do rutinního zavedení jako závazného pracovního postupu při programování hromadných agend uplynulo zhruba 18 měsíců. Dlouho se musely překonávat konzervativní názory, muselo se argumentovat proti tvrzení, že se tímto bere individualita programátoru a podobně. Po poměrně dost dlouhém čase přesvědčování se nám tedy povedlo tuto metodu zavést do běžného užívání a nyní již nově školení programátoři ani jinak programovat neumějí.

Hlavní výhody vidím dvě:

- 1) Logicky náročné aktualizací programy, kdy vstupují dva i více souborů, u kterých jsou problémy s "doháněním" podle třídících klíčů a ještě větší problémy se zpracováním při dočtení jednotlivých souborů, přestávají být doménou starých a zkušených programátorů. I u nich se však mnohdy třeba po několikaletém používání programu v rutinně projeví chyba v takto složitém programu. Tento problém nyní prakticky neznáme, tyto dříve náročné a choulostivé programy zpracovávají zcela bezpečně středně zdatní programátoři.
- 2) Je zřejmě všeobecným zjevem, že v průměru 60-80% programovací kapacity je spotřebováno na údržbu programových systémů. Málokdy se dosáhne toho, že změnu do programu provádí právě autor tohoto programu. Před zavedením normalizovaného programování každý programátor vtlačil do programu své já, ovšem pro neautora bylo mnohdy nemožné, a to i v případě, že existovala rozumně provedená dokumentace, se v tomto programu vyznat a učinit správné zásahy. Metoda normalizovaného programování zcela odstraňuje tyto problémy. Programy se po formální stránce silně podobají, programátor ví, že např. pod symbolem začínajícím písmenem D je vždy test změny klíčů, který se v programu již nemůže nikde vyskytnout. Toto ovšem značně urychluje práci a rovněž bezpečnost provedení oprav je na mnohem vyšší úrovni.

Argument proti použití normalizovaného programování, že se tím bere individualita programátoru, že ho tato práce nebude lávit, nemůže obstát. Programátor má ještě vždy dostatek prostoru pro to, aby realizoval své osobní přístupy, aby v této práci nacházel zalíbení a uspokojení. Na druhé straně, v zájmu podniku je a musí být maximální efektivnost zpracování úloh a efektivní využívání počítače. Není první a poslední snahou podniku, aby každý jeho zaměstnanec odcházel denně spokojen s tím, co udělal, i když to udělal z objektiv-

ního hlediska zcela špatně.

V další části bych se chtěl věnovat otázce programovacích jazyků. Všechny modernější počítače mají vesměs k dispozici překladače ke všem základním vyšším programovacím jazykům jako PLI, COBOL, FORTRAN, ALGOL. Mnohde bylo ponecháno na jednotlivcích, v jakém jazyce budou programovat, podle toho, který se jim více líbí, nebo který již dříve uměli. Předložený popis organizace práce v týmech předpokládá především existenci zásadně jednoho programovacího jazyka. Jedině za tohoto předpokladu lze provádět přesuny programátorů mezi týmy bez potíží. Vícejazyčnost je již silným omezením, ne-li znemožněním této práce. Dále, obvykle se při zpracovávání dat přistupovalo a mnohdy ještě přistupuje agendově a teprve později dochází k integraci těchto agend. Vícejazyčnost může tyto záměry značně ztížit, ne-li znemožnit. O tomto zřejmě nebude nikdo pochybovat. Z tohoto je nutno vyjmout programování speciálních úloh speciálními jazyky.

Mnohem častěji diskutovaná otázka je volba programovacího jazyka. Přes nedlouhou historii používání výpočetní techniky byly o tomto problému, o srovnávání jednotlivých jazyků, napsány mnohé úvahy, ba i knihy. Náš názor, názor programátorů, kteří programovali ve všech vyšších a v několika nižších jazycích je - což v zápatí budu obhajovat - COBOL pro zpracování hromadných dat.

Pro tento účel v podstatě jsou předurčeny jazyky PLI a COBOL. Proč jsme se rozhodli právě pro COBOL? Jsme toho názoru, že při zpracování většího počtu programů - naše roční produkce je cca 600-700 programů a všichni vlastníci středních počítačů dříve či později budou dosahovat nebo dosahují nejméně stejného počtu - je třeba mít k tomuto nástroj jednoduchý, avšak umožňující vše, co je třeba, ale na druhé straně neumožňující programátorům začátečnickům nebo méně zdatným, aby se utopili v příliš širokém spektru možností, které jazyk poskytuje. Jazyk PLI je nesporně v současné době nejlepším jazykem, který umožňuje tomu, kdo jej ovládá, na-

psat velmi efektivní program. Máme však neblahé zkušenosti, že v rukou řadového programátora, který své zaměstnání bere spíš jako existenční nutnost než jako potřebu k realizaci svého duševního potenciálu, může být jazyk PLI velmi nebezpečným nástrojem. Nevhodným použitím některých z mnoha možností (PLI např. má pro čtení 5 výroků - COBOL jeden), se dá napsat velmi neefektivní program. Musí se stále více počítat s tím, že nadšenců a fandů programátorů, kteří svou profesi zvládli do nejmenších podrobností, bude s rychlým rozšiřováním počítačů stále méně a že se postupně tato profese stane běžnou, bez exotické příchutí. Jazyk COBOL splňuje prakticky všechny požadavky i náročného programátora a přitom je poměrně jednoduchý, bez uváděné možnosti nepsání neefektivního programu. Dalším, i když ne příliš podstatným důvodem hovořícím pro COBOL je menší nárok na paměť přeloženého programu, často i kratší překlad. Z těchto důvodů jsme přešli po roce užívání jazyka PLI zpět ke COBOLu, který jsme užívali již dříve u ICL.

Při dnešním nedostatku pracovních sil vůbec je velmi obtížné provádět příliš přísný výběr nových programátorů. Obvykle jsou vedoucí pracovníci postaveni před rozhodnutí: buď získám tohoto člověka, i když o něm vím, že pro tuto práci se příliš nehodí, nebo nedostanu nikoho. Kvalitní, starší vyškolení programátoři na trhu pracovních sil bývají jen velmi vzácně; každý podnik si takového člověka umí udržet. Obvykle se lehce pouští ti, o které takový zájem není. Nicméně je a bude třeba podrobovat uchazeče o toto zatím ještě dost specifické a nepříliš rozšířené zaměstnání výběrovým testům, aby nedocházelo jak na straně pracovníka, tak na straně zaměstnavatele k problémům vedoucím k nepříjemným situacím. Radu let se nám osvědčuje výběrový test, jehož autorem je firma ICL. Protože však nepostihoval zdaleka všechny vlastnosti programátora (jako např. schopnost abstrakce, zobecňování, tvoření pojmů, praktické početní myšlení, kombinační schopnost, chápání vztahů, důkladnost myšlení, po-

hyblivost a proměnlivost v myšlení, teoretické myšlení, induktivní myšlení s číslly, tvoření úsudku, konkrétně praktické myšlení, emocionální stabilita, odolnost vůči psychické únavě aj.) byly zpracovány ve spolupráci s psychosociologickou laboratoří OKR Ostrava výběrové testy, které by s mnohem větší mírou určitosti měly dát vedoucím pracovníkům solidní podklady pro rozhodnutí o přijetí.