ZKUŠENOSTI S PROGRAMOVÁNÍM V DATABANKOVÉM A ON-LINE PROSTŘEDÍ

RNDr. Josef Bujnoch, Ing. Dominik Vymětal, CSc.
Třinecké železárny VŘSR, národní podnik Třinec

Úvod

Přednáška je pokusem shrnout a do určité míry zobecnit zkušenosti z programování a činností, které jej bezprostředně ovlivňují v Třineckých železárnách VŘSR pod databankovým (DB) a datakomunikačním (DC) systémem IMS. Obě části nejsou diskutovány zvlášť, spíše se zaměřujeme na jednotlivé problémy, které ovlivňují programování v prostředí DB/DC systému a snažíme se naznačit jejich řešení nebo alespoň náš názor na tento problém. Závěr shrnuje některé širší souvislosti, které mají vztah k programování a které DB/DC systém jen podtrhuje.

1. Centrální zabezpečení dat

DB systémy se vyznačují tím, že zabezpečení dat před ztrátou je organizováno centrálně a jednotně. Základem tohoto zabezpečení bývají většinou pravidelně pořizované kopie datových základen a logový (žurnálový) soubor se zachycením aktualizací jednotlivých datových základen. Zde si je zapotřebí uvědomit, že datové základny se stávají funkcí času. Tato skutečnost vystupuje nejzřetelněji do popředí u on-line aktualizovaných datových základen.

Zde je datová základna sdílena více programy či uživateli. Není tedy možné pořizovat si nějaké kopie pro jednu aplikaci a ty pak vracet do datové základny po případném

neúspěšném zpracování. Takovéto zásahy jsou zcela nepřípustné v rutinním zpracování a tuto zásadu je třeba dodržet a mít na zřeteli i při ladění na datových základnách, které jsou již v provozu.

Dalším problémem, který se ve vztahu ke způsobu práce a zabezpečení datových základen může objevit, je potřeba zachytit určitý konkrétní stav datové základny v určitém okamžiku. Například u on-line vedeného skladu je pro potřeby uzávěrky zapotřebí zachytit stav, kdy všichni skladníci ukončí výdeje a příjmy za daný měsíc. Problém řešíme OZ řídicích dat, která obsahuje stav, v jakém se příslušná datová základna nachází. Každý aktualizační program pak musí zkontrolovat, v jakém stavu datová základna je a zda smí do ní přistupovat. Pro výše naznačený problém zřejmě stačí dva stavy :
1. otevřeno pro příjmy a výdeje,
2. zavřeno pro příjmy a výdeje.
Přitom jsou-li příjmy a výdeje prováděny z více míst, je nutno evidovat uzavření OZ z každého místa zvlášť a uzavřít OZ až při uzavření ze všech míst.

V souvislosti s centrální údržbou DZ je třeba uvést ještě jednu skutečnost. Je nutné si promyslet vztah používaného programovacího jazyka a prostředků zabezpečení dat DB systému. Systémy řízení bází dat nívají prostředky pro vrácení DZ do stavu, který byl na DZ před započetím určitého aktualizačního programu, v případě, že tento program skončí "špatně". Právě toto "špatně" je třeba prozkoumat. Například v případě Třineckých železáren u programů v PL/1 při užití systému IMS je pro PL/1 "špatné" ukončení konec s CC=2000 a pro IMS pouze abnormální ukončení. Bylo zapotřebí zavést u všech aktualizačních programů v PL/1 jejich abnormální ukončení v případě neopravitelné chyby.

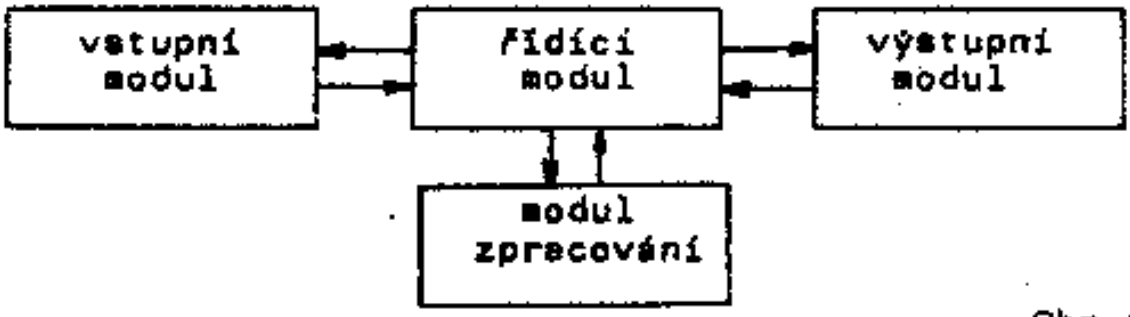
2. Odtržení vstupu a výstupu od výpočtu

Zásada odtržení vstupu a výstupu od výpočtu vystoupí

zřetelně v systému s DC částí pracujícím v režimu on-line. Terminálová síť umožňuje prodloužit výpočetní středisko až k uživateli, dává možnost operativního poskytování informací a sběru dat. Za tyto výhody se většinou platí v oblasti programování, neboť nelze předpokládat, a ani by to nebylo účelné, že terminálové síť bude vybavena jednotnými terminály.

Je nutné, aby program pracující pod určitým datakomunikačním systémem "rozpoznal", z jakého terminálu se na něho uživatel obrátil a věděl, na jaký terminál odpovídá. U nás jsme to vyřešili ve standardu pro logická jména terminálů, která jsou v programu přístupná.

Nejvhodnější struktura programu je uvedena na obr.1.



Obr.1

Řídící modul řeší otázky začátku a ukončení zpracování a sekvence volání ostatních modulů. Řídící funkce mohou být zahrnuty i v modulu zpracování, ale je výhodnější tvořit speciální řídící moduly tak, aby moduly zpracování se skutečně zabývaly jen zpracováním.

Moduly zpracování oproštěné od řídících funkcí jsou pak použitelné ve více různých aplikacích. Modulů zpracování může být i více, sekvence jejich volání je zakotvena v řídícím modulu.

Vstupní a výstupní moduly jsou svázány s řídícím dohodnutou strukturou vstupních či výstupních dat. Řeší otázky neformátovaného vstupu z některých typů terminálů, otázky

ukončení vstupu apod., otázky související se zvláštnostmi výstupu na ten který druh terminálu.

3. Práce s terminály

Významným faktorem ovlivňujícím programování (a celý návrh aplikace) při práci s terminály, pokud nejde o lokální terminály, je rychlost odpovědi. Čas T, za který dostane uživatel u terminálu odpověď, se skládá z dílčích časů podle vzorce :

$$T = T_p + T_z + T_d$$

Čas T_p označuje dobu přenosu vstupní zprávy z terminálu do počítače a dobu přenosu výstupu z počítače na ter-minál. Tato doba, kterou ovlivníme pouze minimalizací délky vstupu a výstupu, se na celkové době T podílí dosti podstatně. Např. přenos plné obrazovky (1920 znaků) do počítače a zpět při použití modemů o rychlosti 2400 bitů/s je 12 s.

Čas T_z označující dobu zpracování bez I/O operací sice programátorsky ovlivnit můžeme, ale podíl T_z na celkovém čase T je tak zanedbatelný, že kromě vyloučení třídění souborů přesahujících 100 vět, není se třeba touto částí zabývat.

Čas T_d označující přístupy do datové základny dobu odezvy ovlivňuje poměrně značně a lze jej při návrhu a programování aplikace ovlivnit. Tyto úvahy se budou týkat zejména programů poskytujících výsledky a budou ovlivněny frekvencí užití a počtem "základních" položek údajů, z nichž vznikají agregované informace.

Problémy přináší údržba agregací současně s aktualizací základních položek. Nejlepším řešením se ukázalo vytvořit pro aktualizaci určitého typu agregací modul, jehož parametry bude adresa (klíč) agregace, starý obsah aktualizované položky a nový obsah aktualizované položky. V případě vstupu nové položky bude starý obsah aktualizované položky nulový.

V neposledni řadě napomáhá efektivitě běhu programů dobře zpracovaná dokumentace, která uživatele vede ke správnému a účelnému využívání aplikace. K tomu mohou přispět vhodné nápovědi pokračování přímo ve výstupních zprávách. Vytváření tzv. "jídelniček" či dokonce zavedení funkce "help" je třeba pečlivě zvažovat s ohledem na narůstající časy T_p. Dají se zavést tam, kde lze předpokládat práci s menší frekvencí, ale významnou. To bude zejména u dotazových systémů provrcholové řídící pracovníky. Jsou zbytečné u aplikací pro každodenní práci referentů. Zde by měly působit zase standardy a zásady jednotné komunikace v systému. Jde o to, aby všechny aplikace používaly co možná nejjednotněji např. funkční klíče, formální úpravu vstupu, zásady pokračování apod. Formální sjednocení aplikací se pozitivně projeví na pracovištích, kde se prolínají dvě nebo více aplikací.

4. Ladění programů

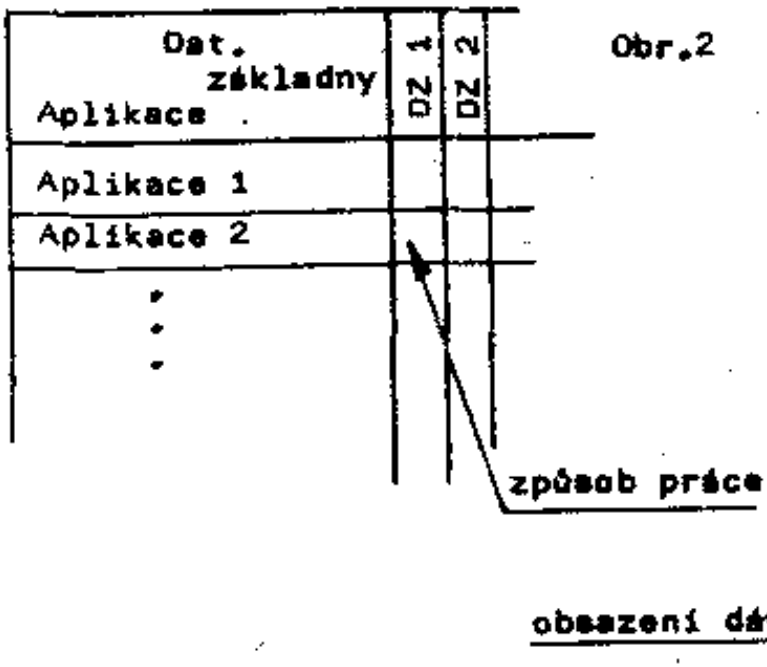
Vliv OS/OC systému na ladění programů se projeví nejzřetelněji v momentech ladění nových programů, zejména aktualizačních, na již provozovaných bázích dat. Zde je zapotřebí dobře zvážit již výše zmíněnou skutečnost, že datové základny se stávají funkcí času a není možné jejich "vrácení" do původního stavu před laděním.

Držíme-li se schematu on-line programu znázorněného na obr.1, může nám to přinést výhody i v ladění. Nejaložitější částí na odladění bude zřejmě modul zpracování, který provede všechny zásahy do datové základny. Máme-li po ruce vhodné vstupní a výstupní moduly, je možno připravit soubor ladicích dat takový, že jádro zpracování mohou dobře a pohodlně odladit v dávce. Řídicí modul může v ladění obsahovat i pomocné výpisy apod. Převedení programu z ladicího stavu do provozního pak při dobře definovaných zásadách spolupráce modulů nemůže být problémem.

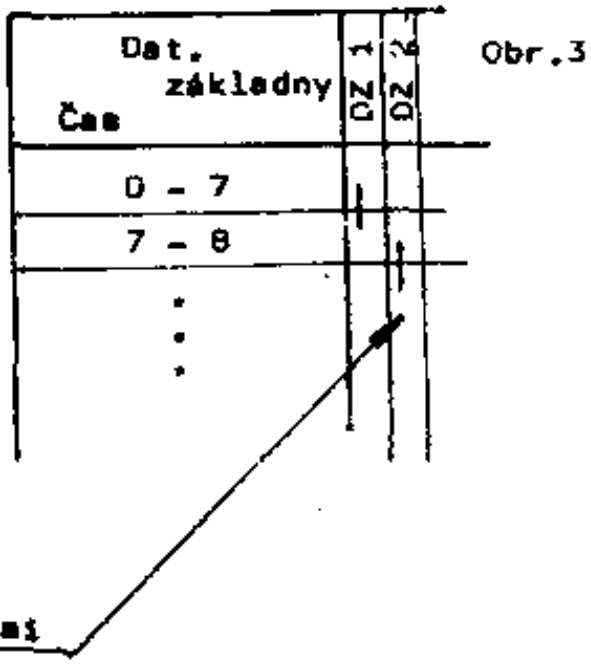
Zdá se přirozené, že do provozu by měly jít programy odladěné. Považujeme však za nutné upozornit, že toto platí u on-line systémů dvojnásob. On-line program by měl možné situace řešit co nejúplněji a nestandardní a neřešitelné situace řešit co nejzřetelněji, tj. abnormální ukončení. Zde se opět projevuje nevýhoda ukončení PL/1 programu s CC=2000 při chybě.

5. Návrh systému jako celku

Nadpis této části se zdá samozřejmý. Jestliže však v systému s dávkovým zpracováním neúplný návrh přinese nepříjemnosti pouze výpočetnímu středisku, může neúplnost návrhu DB/DC systému značně snížit jeho efektivitu a účinnost. Problémy totiž mohou vyvolat delší dávkové zpracování na datových základnách sdílených více aplikacemi v režimu on-line. Dávky mohou totiž značně zpomalit nebo zcela zastavit on-line zpracování s nepříjemnými důsledky u uživatele. Je vhodné v průběhu návrhu sestavovat dvě následující tabulky.



Obr.2



Obr.3

V tabulce na obr.2 ehromáždíme všechny aplikace a jejich způsob práce s jednotlivými datovými základnami. Na jejím základě sestavíme tabulku na obr.3, kde zachytíme obsazení jednotlivých datových základen dávkami. Porovnáním obou tabulek můžeme zjistit, které on-line aplikace a kdy mohou mít problémy.

Obecně se dá vyslovit doporučení k dávkovým zpracováním v on-line systému v tom smyslu, že je třeba omezit na nejúspornější možnou míru aktualizační dávky a dávkové tiskové výstupy pořizovat z kopií datových základen ve formě klasických souborů pořizovaných v pevných, potřebami zpracování daných intervalech.

Závěr

Systémy řízení bází dat dávají programátorům a analytikům do rukou mohutné, kvalitativně vyšší nástroje. Vyžadují však i kvalitativně vyšší přístup k návrhům aplikačních systémů, jednotlivých programů a způsobu jejich práce. Nutnost opakovaných standardních programátorských činností například při ošetřování výsledků požadovaného přístupu na datovou základnu, odtržení řízení výpočtu od vstupních a výstupních operací a vlastního zpra-cování, možnosti využití stejných postupů v různých typech programů, nutnost důsledného dořešení všech chybových stavů, to jsou všechno skutečnosti, které podtrhují výhody modulárního přístupu v databankovém a datakomunikačním prostředí.

Je třeba upozornit na stoupající význam slovníků a adresářů dat, a to v pojetí informačních systémů o informačním systému. Tyto systémy schopné zachytit strukturu ASŘ se stanou významným pomocníkem a zdrojem informací i pro programování, a to zejména ve fázi údržby aplikačních programových systémů. To proto, že používáme-li modulárního přístupu, pak v mnohých případech udržet aktuální informace o tom, kde je který modul použit, je značně pracné.

Trendy, které přinesly databankové a datakomunikační systémy do zpracování dat a které se promítají i do oblasti programování, jsou v Třineckých železárnách rozpracovávány a zobecňovány do filosofie "otevřené architektury ASŘ", která se vyznačuje integrací subsystémů řízení v horizontálním směru, zabezpečením hierarchických řídicích a informačních vazeb v prostředí distribuovaných datových a technických prostředků počítačových sítí.