D I S T R I B U O V A N O S T (na cestě k páté generaci ?)

Zd. Rusín

Přestože denně přicházíme do styku s profesionálními počítačovými systémy pracujícími s využitím prvků distribuovanosti (asynchronnost či paralelismus logických i fysikálních procesů, komunikace formou výměny zpráv), přestože i z toho mála ne vždy aktuálních odborných publikačních příležitostí jsme se seznámili s východisky i cíli zahraničních projektů tzv. páté generace, nacházíme skutečnou distribuovanost v uživatelské sféře jen zřídka.

Je-li základní objektivní technickou překážkou bezesporu neexistence kvalitních komunikačních služeb, pak v subjektivní oblasti narážíme na neznalost zcela běžných organisačních principů v počítačových systémech používaných.

Cílem následujícího textu samozřejmě není žádná dalekosáhlá reforma všeobecné úrovně počítačové gramotnosti v naší vlasti, nýbrž prostá snaha upozornit alespoň jistou část profesionálních tvůrců programového vybavení na aktuálnost výkonných a spolehlivých distribuovaných uživatelských aplikací i v našich podmínkách.

Probírané koncepty jsou ilustrovány praktickými realisacemi jediného výrobce, britské firmy ICL, s jejichž zařízeními autor trvale pracuje.

Ve shodě s realitou můžeme charakterisovat dnešní centrální výpočetní systémy jako síť hierarchisovaných účelových (mikro)procesorů, spolupracujících na základě vzájemné výměny signálů při exekuci uživatelských úloh. Tato situace je nám natolik samozřejmá, že jsme ochotni takovýto systém s jedním procesorem řídicího kódu považovat za typickou realisaci klasického schematu sekvenčního stroje, když například asynchronnost periferních operací a víceprogramové sdílení procesorového času si vpodstatě neuvědomujeme.

Tím méně zřetelná je tato situace 'klasickému' uživateli výpočetního střediska, jenž na konci řetězu dávkových úloh očekává svůj balík sestav. A po pravdě řečeno, ani pozice programátora u terminálu se ještě nemusí zmíněné uživatelské pozice příliš vzdalovat,

zvláště je-li odezva systému slušná a scénář práce nevybočuje z kolovrátku input/edit-kompilace-testování. Nicméně však již obvykle vstupují na scénu rušivé momenty opožděného spoolingu tisků, aktivace terminálového zařízení, sdílení dat atd., které mohou podnítit zájem o procesy, jimiž jsou programátorovy činnosti realisovány. V programy řízených dialozích koncových uživatelů takové nebezpečí bývá řidší, tím větší je bezradnost ve výjimečných situacích. Tyto momenty jsou reálnými zdroji přetrvávající nevědomosti a to tím spíše, že mnohé ostatní formy paralelismu a distribuovanosti povětšinou známe pouze z literatury (připomeňme namátkou duplexní zápis souborů na velkokapacitních pevných discích).

Důležitým krokem na cestě k distribuovaným systémům jsou počítače s více procesory řídícího kódu. Podnětným impulsem páté generaci jsou systémy s regulárním paralelismem (Single Instruction, Multiple Data), vykonávající identické operace na 'stejnolehlých' datových prvcích; a zdaleka nejde jen o numerické aplikace, princip se ukázal výhodný i pro prohledávání textů a rozpoznávání tvarů (pattern searching, pattern recognition). Na počátku osmdesátých let byl zdařilou aplikací tzv. ICL Distributed Array Processor, realisovaný jako pole 64 x 64 procesorů, pracujících každý nad čtyřmi kbity paměti (celkem 2Mb) plně kompatibilní s ostatní vnitřní pamětí hostitelských počítačů řady ICL 2900, který mohl být aplikacemi užíván buď jako obvyklá paměť nebo (výlučně) ve své procesorové funkci.

Těchto zařízení bylo vyrobeno několik desítek, většina instalací dosud pracuje; výkonnost systémů stoupala s řádem řešeného problému. Technickým úskalím se patrně stal postupující řád integrace paměťových prvků; alespoň autorovi není známa ani firemní literatura odkazující na implementaci DAPu pomocí prvků vyšší integrace.

V rámci současných evropských integračních snah je u téhož výrobce rozvíjen mj. projekt 'FLAGSHIP', předpokládající neregulární víceprocesorový paralelismus (Multiple Instruction, Multiple Data) s libovolným počtem procesorů (v rozmezí 8 - 256), z nichž každý obsluhuje 16 až 64 Mbyte paměti, které mezi sebou, pamětmi a hostitelským systémem komunikují vysokými rychlostmi (prototyp užívá 4Mb RAMs a procesory Motorola 68020).

'FLAGSHIP' má mít k dispozici deklarativní jazyky zpracovávané metodou redukce výpočtových grafů, které jsou representovány v kon-

venční paměti existujícími 'packety', obsahujícími zápis funkce/operace spolu s jejími argumenty. Forma grafu zřetelně lokalisuje přirozený paralelismus výpočtových algoritmů, redukce spočívá v paralelním zpracování packetů směrem ke kořeni grafu.

Zcela reálnou budoucností některých československých podnikových výpočetních středisek jsou několikaprocesorové miniaturisované nodální systémy ICL 3915 až 3935 s velkokapacitními fixními disky, lokálními sítěmi s přenosovou rychlostí 10 Mbitů za sekundu a s distribuovaným projekčně-programátorským vybavením, které mají s výše citovanými MIMD stroji principiálně obdobnou meziuzlovou komunikaci. K nim se v textu ještě vrátíme.

Spolu s pokrokem v oblasti databázových struktur byly navrhovány i konstruovány specialisované databázové procesory. Odtud byla vyvozena s užitím fuzzy vyhledávání koncepce 'datového síta' ve formě datového procesoru spjatého s kontrolerem diskových pamětí, osvobozující uživatelský program (a s ním procesory řídícího kodu) od selekce dat. Hardwareová realisace, chráněná řadou původních patentů, se ukázala komerčně nesmírně úspěšná a CAFS-ISP, Content Addressable File Store - Information Search Processor, se stal samozřejmým základním příslušenstvím počítačů ICL.

CAFS pracuje jako datový filtr na všech paralelně čtených diskových segmentech. Využívá jej jak speciální firemní software, tak extense programovacích jazyků, umožňující relační přístup k síťovým strukturám IDMS i jednoduchý aparát pro prohledávání diskových souborů klasických organisací. Je zřejmé, že existence tohoto zařízení na konkrétní instalaci může podstatněji ovlivnit nejen výkonnost úloh, ale především projektování datové základny. Relační pohled na data by měl stimulovat rozvoj znalostních bází.

Zájem o vývoj centrálních systémů zhruba na přelomu sedmdesátých a osmdesátých let opadl v důsledku explose mikropočítačů. Toto období doslova rozmetalo dosavadní počítačovou praxi; z našeho pohledu je však nejpodstatnější rozvoj lokálních komunikačních sítí umožňujících spolupráci i různorodých zařízení formou výměny dat, sdílením diskové kapacity či speciálních zařízení (grafika, komunikace s centrálními systémy). Spolu s růstem přenosových rychlostí užitím optiky mohla být formulována i uskutečněna představa distribuované uži-

vatelské interaktivy, realisované souběžnými procesy na různých počítačových prostředcích lokální heterogenní sítě. Z tohoto pohledu se LAN (Local Area Network) blíží svou funkcí 'rychlé sběrnici' střediskového počítače třetí generace.

Zůstává ovšem zásadní rozdíl v úrovni součinnosti komponent centrálního systému a součinnosti různorodých zařízení otevřené lokální sítě. Vysoká samostatnost jednotlivých celků v lokální síti je obvykle zdrojem větší pracnosti při tvorbě distribuovaných aplikací; to plyne už z toho, že část kódování respektujícího komunikační protokol sítě musí být zahrnuta v uživatelském vybavení kooperujících prostředků.

Podstatně lepší vyhlídky má lokální síť centrálního počítačového systému, zvláště pocházejí-li všechny začleněné prostředky od téhož výrobce a existuje-li solidní základní programové vybavení pro provozování komunikačních služeb v síti. A což teprve, jsou-li k dispozici ucelené distribuované aplikace, speciálně třeba pro projektování, tvorbu, dokumentaci a provozování centrálních databázových služeb. Pak už k degradaci sítě nepřispěje asi ani výkonný grafický, barevný, 32-bitový lokální prostředek s několika megabytes operační paměti a dostatečnou diskovou kapacitou, komunikující vysokou rychlostí, který by mohl vyvolávat v uživateli pocity plné soběstačnosti. Samozřejmě tehdy, jsou-li lokální uživatelské potřeby dostatečně podporovány spolehlivými centrálními datovými službami a je-li lokální zařízení užíváno tvůrčím způsobem odborníky dané profese, dostatečně obeznámenými s principy distribuovaného systému.

Osou technického řešení počítačových systémů se tedy stává mezi-procesorová komunikace, typy a formáty zpráv a režim jejich výměny. Podstatných změn by měly doznat i některé stavební principy operačních systémů, což dosud naráží na nedostatečnost mezinárodních standardů a nejednotnost v jejich implementaci. Problémem lokálních komunikačních programových balíků bývá plné využití možností koncových zařízení (grafika, speciální funkce). Tutéž výhradu však můžeme kdykoli učinit vůči operačním systémům centrálních počítačů, které dostatečně rychle nereagují na vývoj koncových zařízení.

Obraťme nyní pozornost zpět k nodálním několikaprocesorovým systémům. Už i proto, že tzv. ICL Distributed Mainframe 1 (DM1, modely 3915 až 3935) je téměř jistým prvým zástupcem této třídy počítačů v ČSSR.

'Lokální' systém se skládá ze dvou (jednoho) až čtyř uzlů, tvořených procesorem řídícího kódu, pamětí, inženýrskou diagnostikou a kontrolery tří lokálních sítí - sítě meziuzlové komunikace, sítě ovládání magnetických paměťových zařízení a komunikační sítě terminálových prostředků a klasických velkokapacitních tiskáren.

Komunikační síť může být sdílena několika nodálními systémy; totéž platí i o síti ovládání magnetických paměťových zařízení. Takto lze periferie rekonfigurovat v průběhu pracovního cyklu (dne,směny) mezi několika systémy. Údržba databází tak například může být prováděna na jiné instalaci než interaktiva; variabilita a odolnost centrálních systémů vzrůstají. Poznamenejme, že použití fixních disků přináší jiný způsob zabezpečení aktivních dat, než byl provozován na instalacích se spolehlivými vyměnitelnými diskovými médii.

Nodální systém hostí jedinou kopii řídícího systému, jenž umožňuje v každém uzlu hardwareovou a firmwareovou exekuci řídícího kódu jiného systému (konkrétně jde o starší ICL systémy a UNIX). Protože uzly sdílejí společné zdroje (magnetická media, paměť), musí k nim přistupovat pomocí mechanismů řízení kritických oblastí; jsou užity hardwareově implementované semafory, jejich reservace a uvolňování musí být oznámeny všem aktivním uzlům. Základním stavebním blokem alokace zdrojů se stává zobecněný koncept 'směrované' procedury vykonávané v rámci procesu. Stav procesu je plně definován virtuální pamětí, kterou proces v danou chvíli vlastní. Ta je členěna na segmenty jistých vlastností, zřizované jednotlivými vykonávanými procedurami. Distribuce procedur mezi uzly vynucuje tedy distribuci paměti; současně je distribucí procesů mezi uzly. Protože uzly nemusí být fysicky ekvivalentní, připouštíme rozdílnost kódu a různou velikost paměti, musí být mapování virtuální paměti do reálné paměti prováděno na úrovni jednoho uzlu. Kromě sdílených paměťových segmentů, jejichž změna musí být možná bez ohledu na to, zda reálná paměť přísluší zlu, který změnu vyžaduje, tak docházíme k pojmu 'nodální paměti' pro ty segmenty, které mají v každém uzlu týž význam, ale jiný obsah. Takové jsou především stránkovací tabulky. Jejich obsah není viditelný ostatním uzlům.

Meziuzlová komunikace obsahuje tedy kromě řízení času a procesů (clocking & low level scheduling), konektování a diskonektování uzlů, ošetřování chybových stavů a počáteční inicialisace celého nodálního systému tři synchronisační činnosti: změnu stavu systémových semaforů, změnu obsahu sdílených paměťových segmentů a distribuci procesů mezi uzly při vyvolání a opuštění procedury. Protože všechny tyto funkce jsou implementovány jádrem operačního systému, je přechod k nodální koncepci možný beze změny stávajícího uživatelského rozhraní (tím je v konkrétním případě operační systém VME).

Efektivnost implementace je podmíněna možností minimalisovat meziuzlovou komunikaci. K tomu lze využít ustavení virtuálních cest mezi každými dvěma uzly a explicitního nahrazení sdílených datových struktur strukturami nodálními (např. při komunikaci procesů trvale přítomných v jednom uzlu).

Zobecněný pojem procedury užitý v nodálních systémech už reflektuje potřeby deklarativních jazyků, které by měly řešit dosud nepřekonané rozdílnosti v chápání objektů konvenčních programovacích jazyků a databázových jazyků, včetně jejich presentace uživateli. Současný vývoj dospěl k neprocedurálním formalisovaným specifikacím datových objektů, jejich presentace a procesů nad nimi pomocí databázově implementovaných slovníků dat (viz ICL Quick Build software na VME instalacích, realisující již mnohé z dále popisovaných aspektů deklarativního programovacího stylu).

Bylo konstatováno, že konvenční programování vynakládá většinu kódování na konverse dat mezi fysickým záznamem, jazykem a jejich uživatelskou presentací (užívají se pojmy MMI: Man - Machine Interface nebo HCI: Human - Computer Interface). Dochází se ke koncepcím 'Persistent Language' a 'Persistent Information Space Architecture', jejichž objekty, ať jde o proměnné v klasickém smyslu, databázové entity (záznamy a položky v nich) či procedury, funkce a operace, nepodléhají kontextuálním reinterpretacím obsahu a transformacím uživatelské presentace na úrovni aplikačního programu.

Program v persistentním jazyce nepotřebuje pracovní data, soubory a HCI formáty konvenčních jazyků; prostě užívá své objekty v kontextu jednotného persistentního informačního prostředí, jež sdílí s mnoha dalšími uživatelskými procesy.

Pokud to laskavého a vytrvalého čtenáře přivádí k představě dalšího všeobsáhlého monstrózního jazykového giganta typu PL1 či ADA, pak řešitelé zmíněných projektů zdůrazňují potřebu řady problémově orientovaných deklarativních persistentních 'dialektů', zásadně však odmítají rozdílné jazykové prostředky pro komunikaci s hostitelským prostředím, pro přístup k datům, pro jejich výpočtové transformace a pro presentaci uživateli. Deklarativní persistentní jazyky tedy hodlají reflektovat uživatelské požadavky, nikoli obskurní pseudonezbytnosti neobratně implementovaných počítačových prostředků. Tím jsou blízké existujícím 'objektově orientovaným' jazykovým systémům jako SIMULA a LISP. Ještě jinak řečeno, jde o jednotný homogenní pohled na všechny objekty, pomocí nichž a s nimiž aplikace pracují.

Základem jazykového systému jsou vždy pojmenovávací pravidla a s nimi spjaté algoritmy selekce pojmenovaných objektů. Rozeznávají-li klasické jazyky v podstatě tři typy selekce objektů - selekci lokální proměnné, interpretaci významu parametru procedury a výběr procedury realisující operaci nad větou souboru, pak persistentní jazyk vystačí vždy s prvými dvěma, jež doplňuje mechanismem budování nových úrovní abstrakce nad persistentním informačním prostředím pomocí abstraktních datových typů, jimiž lze nově či alespoň lépe řešit uživatelskou problematiku. Pojmenovávání nových objektů vychází vždy z kontextu už selektovaných objektů. Implementační úroveň persistentního jazyka pak musí řešit v rámci adresování objektů dva zásadní problémy - převod objektu z jednoho kontextu selekce do kontextu jiného a rozpoznání identity dvou různě selektovaných objektů v rámci neomezeného distribuovaného persistentního informačního prostředí. Vyhodnocování pointrů musí respektovat potřeby sdílení objektů více procesy, což u procedur vede k sdílené reentrantnosti grafových výpočtových struktur, u databázových entit k transakcím s užitím operací typu START, ABANDON a COMMIT.

V operačních systémech VME a UNIX byl implementován PS-algol Persistent Language System a realisuje se jeho extense Napier Language. Práce jsou prováděny v rámci integračního projektu Alvey.

Pro úplnost dodejme, že zobecněný pojem směrované procedury intuitivně odpovídá významu operace v uzlu výpočtového grafu deklarativního persistentního jazyka.

Slovo závěrem:

tento text je doplňkem dvou článků z ročníků 82 a 85, popisujících a současně zobecňujících VME prostředí. Na rozdíl od všech dosavadních autorových příspěvků pro tento seminář, nevychází text z každodenní reality autorova pracoviště, nýbrž se převážně věnuje blízké budoucnosti některých československých VME instalací. Termín 'blízká budoucnost' byl užit s vědomím, že

> "vždy to trvá déle než jste očekávali; i tehdy, vzali-li jste tuto skutečnost v úvahu" (Douglas R. Hofstadter).

Užitá literatura:

/1/ B. C. Warboys, VME nodal architecture: a model for the realisation of a distributed system concept,
ICL Technical Journal, vol 4, iss 3, May 85

/2/ P. Townsend, Flagship hardware and implementation,
ICL Technical Journal, vol 5, iss 3, May 87

/3/ I. Watson, J. Sargeant, P. Watson, V. Woods, Flagship computational models and machine architecture, ICL Tech. J., vol 5/3

/4/ M. P. Atkinson, R. Morrison, G. Pratten, PISA - a Persistent Information Space Architecture, ICL Tech. J., vol 5/3

/5/ P. Broughton, C. M. Thomson, S. R. Leunig, S. Prior, Designing system software for parallel declarative systems,
ICL Tech. J., vol 5/3

/6/ J. R. W. Glauert, J. R. Kennaway, M. R. Sleep, DACTL: a computational model and compiler target language based on graph reduction, ICL Tech. J., vol 5/3

/7/ G. McC. Haworth, The CAFS system today and tomorrow,
ICL Technical Journal, vol 4, iss 4, November 85

/8/ R. W. Gostick, Software and algorithms for the Distributed Array Processor, ICL Technical Journal, vol 1, iss 2, May 79

/9/ D. R. Hofstadter, Godel, Escher, Bach: an eternal braid,
Penguin Books

/10/ Z. Rusín, Výpočetní systémy příštích let a jejich dopad na profesní sféru, Programování '82

/11/ Z. Rusín, Systémová podpora komunikačních úloh,
Programování '85