Předmluva k I. části sborníku Lacko-B.

Využijme náš seminář PROGRAMOVÁNÍ 90 k zamyšlení nad cestami, kterými se bude ubírat vývoj programování v nejbližších letech.

Tak jako veletok vzniká spojením menších řek a říček, tak i současné a budoucí programování je tvořeno mnoha vývojovými proudy, které se vzájemně prolínají. Pro snadnější pochopení si jich všímneme v jejich isolované podobě, i když v praxi se často spojují, aby se dosáhlo většího účinku.

Proudy evoluční

Pro nejbližší dobu budou zřejmě nejvíce ovlivňovat programování myšlenky, které postupně rozvíjejí k dokonalosti klasické programovací techniky procedurálních programovacích jazyků. Bude tomu tak zřejmě proto, že tento způsob může přinést v krátké době kladné výsledky, které jsou tolik potřebné pro zmírnění povětšině "softwarové krize", která se prohloubila existencí milionů mikropočítačů.

Myšlenka nutnosti vícenásobné využitelnosti vypracovaného programu a myšlenka *parametrizace* funkcí programů, daly vzniknout takovým programovým produktům jako dBASE, FRAMEWORK, SUPERCALK, LOTUS 1-2-3, EXCEL a dalším.

Myšlenky standardizace ukázaly cestu, jak integrovat různé programy tak, aby vytvořily provázaný programový systém, dovolující vzájemnou výměnu dat, která tak lze použít pro různé účely. Zdáse, že v zužitkování myšlenek standardizace, zejména v oblasti ovládání programových produktů, zůstáváme stále ještě bodně dlužni. Týká se to nejen ovládání různých funkcí aplikačních programů, kde uživatel stejné funkce vyvolává program od programu různě, ale i ovládání operačních systémů a systémových standardních programů, kdy pro programátory jsou vytvářena neodůvodněně různá programátorská prostředí, což ztěžuje přechod programátorů od jednoho počítače k druhému. V oblasti výpočetní techniky bude muset standardizace podstatně změnit principy své práce, pokud nechce stále přicházet s myšlenkami typizace a unifikace v době, kdy zavedení standardů již nic nepřináší.

Převážná většina programů v současné době je, a v nejbližší době bude, psáno v procedurálních programovacích jazycích. Proto stále jsou aktuální myšlenky strukturovaného a modulárního programování.

Zmíněné evoluční proudy však neznamenají odklon z dosavadního směru vývoje programování. V tom je zřejmě obsažen i limitující faktor, který nedovoluje podstatně zvýšit produktivitu programování. Zároveň se objevují problémy s realizací složitých programových systémů pro aplikace z oblasti umělé inteligence.

Všechny výše uvedené směry byly tématem a středem pozornosti našich minulých seminářů. Proto nejsou účastníkům našeho semináře neznámé. Na tomto semináři proto bude pozornost účastníků obrácena na ty proudy ve vývoji programování, které mohou přinést revoluční změnu v programování a nebyly zatím zevrubně diskutovány.

Proudy revoluční

Naděje na revoluční změny v programování jsou spojovány s odklonem vývoje programování od procedurálního přístupu v programování a se změnou základního přístupu k programování.

V současné době jsou známy tři myšlenkové proudy, ze kterých může vzejít nový přístup k programování a přinést radikální, tolik potřebnou změnu.

Funkcionální přístup - využívá toho, že každý výpočet lze realizovat jako posloupnost aplikací funkcí, které z daných vstupů generují výstupy. Při aplikaci funkcionálního přístupu rozkládáme problém postupně na podproblémy tak dlouho, až dospějeme k primitivním známým funkcím. Z nich pak superpozicí sestavíme funkci, která řeší zadaný problém. Na principech funkcionálního programování byl navržen jazyk LISP. Tento jazyk se zdál původně značně akademický. Teprve potřeby, spojené s implementací systémů programů pro aplikace umělé inteligence, ukázaly jazyk LISP jako velmi vhodný nástroj pro tyto účely. V poslední době, v souvislosti s rozvo-

jem systémů CAD, ukázala využití systému AUTOCAD vhodnost použití jazyka LISP i pro oblast uživatelských rozšíření tohoto systému. Použití funkcionálního přístupu k programování navozuje potřebu upřednostnění funkční analýzy při návrhu informačních systémů.

Logický přístup - využívá toho, že každá úloha pro počítač je exaktně dána, takže ji lze formulovat na bázi matematické logiky. Řešení spočívá v sestavení soustavy logických relací. Relace zachycují jednotlivé dílčí mezikroky, vedoucí k dosažení cíle - jímž je řešení problému. S využitím principů logického přístupu byl navržen jazyk PROLOG, který zejména po prohlášení japonských odborníků, že ho zvolili za základ programování počítačů 5. generace, se dostal do centra pozornosti. Jazyk PROLOG se osvědčil jako velmi dobrý prostředek pro vytváření expertních systémů. Poznamenejme, že jazyk PROLOG vytváří příznivé prostředí k programování aplikací relační databáze.

Objektový přístup - využívá skutečnosti, že každý algoritmus pracuje s datovými strukturami. Objektový přístup doporučuje navrhnout potřebné třídy objektů a k nim příslušné operace, které jsou s výskytem těchto objektů svázány. Mechanismus dědičnosti dovoluje vytvářet hierarchie typů objektů. Objektový přístup preferuje použití datové analýzy při návrhu informačních systémů. Je zajímavé, že řada myšlenek, které navodil princip objektového programování, byla zahrnuta do jazyka SIMULA 67, kdy ještě principy objektového programování nebyly samostatně deklarovány. Za jazyk, vzniklý na základě principů objektového programování, je uváděn jazyk SMALLTALK. Značná popularizace myšlenek objektového přístupu v návaznosti na rozšíření datové analýzy způsobila, že řada jazyků jako C nebo Pascal byla rozšířena o možnosti definovat a pracovat s objekty podle zásad objektového programování.

Výhodou všech tří přístupů je, že se opírají o dobře propracovaný teoretický základ. Na druhé straně se zatím jejich aplikace uplatnily vždy v určitých specializovaných oblastech. Zatím nikoliv pro široké spektrum aplikací součas-

ných počítačů. Je to způsobeno řadou potíží, které se dosud nepodařilo překonat. Uvedme zejména vzdálenost od dosavadních programovacích technik, obtížnou využitelnost již existujících programů, nevyhovující čitelnost programů aj. Uvedené přístupy kladou nemalé nároky na výpočetní techniku. Proto pro zpracování programů např. v jazyku LISP jsou konstruovány speciální technické prostředky LISP-procesory.

Jak bylo již uvedeno, k pochopení jednotlivých přístupů k programování je potřeba se dobře seznámit jednak s teoretickými východisky jejich návrhů a dále s příklady programovacích jazyků, které tyto principy realizují. Odlišnost od dosavadních programovacích praktik způsobuje, že ne vždycky jsou cesty k pochopení těchto nových přístupů snadné. V praxi se projevuje i absence vhodné literatury, kterou by mohli naši programátoři pro seznámení s těmito novinkami použít. Proto předkládaný sborník ze semináře PROGRAMOVÁNÍ 90 si klade za cíl seznámit účastníky s principy funkcionálního, logického a objektového programování včetně ukázek několika vybraných programovacích jazyků.

Přednáška našich předních odborníků a navazující diskuse by měla přinést účastníkům lepší pochopení nových přístupů k programování. Je totiž konstatována skutečnost, že programátoři, jejichž praxe vychází ze zkušeností procedurálního programování, velmi těžko chápou zásadně odlišné principy funkcionálního, logického nebo objektového programování. Tento problém nastoluje otázku, zda výchova naší mládeže v oblasti programování se děje správným směrem, když setrvává u procedurálního přístupu k programování často konkretizovaného v jazyku BASIC. Na druhé straně je nutno přiznat, že tyto nové směry v programování nemají zatím vypracovánu metodiku výuky tak, aby ji bylo možno doporučit za základ výuky na školách.

Z dosavadní programátorské praxe víme, že kromě vlastních technických prostředků a teoretických poznatků ovlivňovaly vývoj programování aplikační oblasti počítačů. V nejbližší době se očekává podstatný nárůst zpracování grafické barevné informace prostřednictvím počítačů. V tomto směru má nejpříznivější

postavení objektový přístup.

Letošní seminář nemůže dát jasnou a vyčerpávající odpověď na otázku, jak se bude vyvíjet programování v příštím desetiletí, chce však dát účastníkům informace, které jim umožní lépe chápat ty proudy vývoje programování, jenž ho budou v příštím období ovlivňovat.

Branislav Lacko