P R O G R A M O V Á P O D P O R A S T R U K T U R O V A N É H O P R O G R A M O V Á N Í

RNDr. Pavel Drbal, CSc., VÚMS, koncernová účelová organizace

Článek pojednává o systému programů, který na sebe přebírá mechanickou část programátorské práce; popisovaný systém vychází ze zásad strukturovaného programování a usnadňuje hlavně ladění a úpravy programů. Autorská dokumentace vzniká zčásti přímo z textu programu.

Život programů lze rozdělit na tři hlavní etapy. Jsou to : (1) vznik programu, (2) ladění a (3) údržba programu. Náklady na jednotlivé etapy podle údajů z počítačově vyspělých zemí jsou přibližně v poměru 1:1:2, přičemž náklady na údržbu mají tendenci růst - v některých případech přesahují i tři čtvrtiny celkových nákladů. Zefektivnit programování má největší efekt tam, kde jsou největší náklady - tj. v údržbě programů. Tento fakt si občas neuvědomujeme, za náklady na program se někdy pokládají náklady na vznik a ladění programu do okamžiku jeho předání. Z tohoto povrchního pohledu často vzniká tlak zkrátit prvou etapu vzniku programu. Podle údajů z literatury i zkušeností má zkrácení prvé etapy za následek prodloužení dalších etap a tím i zvýšení celkových nákladů. Životní etapy programů lze dělit trochu podrobněji :

1a. analýza problému a návrh programu,
1b. zapsání programu v konkrétním programovacím jazyce, tzv. kódování,
2a. odstranění syntaktických chyb (omylů a překlepů),
2b. odstranění logických chyb,
3. opravy a úpravy programů, přizpůsobení programů měnícím se podmínkám.

Dosud bylo vytvořeno množství programovacích jazyků a metod práce, které usnadňují programátorskou práci v jednotlivých etapách v té či oné míře. Většina moderních metod vychází z představ strukturovaného programování.

Velkou bolestí programátorské praxe je dokumentace. Ani ne tak uživatelská dokumentace, tu si uživatelé většinou vynutí, ale hlavně autorská dokumentace. Autorskou dokumentaci si lze vynutit většinou pouze administrativními zásahy, její aktualizaci si nelze vynutit skoro vůbec. Příčina je jasná, v prvých dvou etapách není skoro potřeba, teprve v 3. etapě při fluktuaci programátorů či úpravě po velkém časovém odstupu je najednou potřeba, a to kvalitní.

Z těchto úvah jsme vycházeli v naší koncernové účelové organizaci VÚMS při tvorbě systému programů pro usnadnění programování, který nese prozatímní název MEPRO (MEchanizační prostředek PROgramování). Vycházeli jsme hlavně z představ M. A. Jacksona, zvláště z jeho notace. V následujících kapitolách jsou popsány hlavní myšlenky a obraty systému MEPRO, na závěr oceníme, jaký vliv má na jednotlivé etapy života programu. Na poslední stránce článku je uveden příklad, na nějž se v následujícím textu odvoláváme. Prosím čtenáře, aby pochopil, že vzhledem k rozsahu textu lze uvést jen minipříklad. Základem systému jsou tyto představy :

> Tvůrčí práce programátora končí určením struktury programu a specifikací operací v této struktuře.
> Další práce jsou převážně mechanické.
> Program i dokumentace musí vzniknout pokud možno z téhož textu.

Následující odstavce mají za úkol vytvořit představu o činnosti MEPRO. Předpokládají znalost technologie strukturovaného programování. Prosím čtenáře, aby si však uvědomil, že zde nelze uvést manuál systému MEPRO.

Zdrojový text

Program je reprezentován svým zdrojovým textem. Ze zdrojového textu programu jedna část MEPRA generuje program ve zvoleném programovacím jazyce, druhá část MEPRA z téhož zdrojového textu vytvoří dokumentaci ke generovanému programu (takovou dokumentací je poslední stránka tohoto článku). Podstatnou částí zdrojového textu je zápis struktury programu a seznam operací a podmínek (podrobněji viz následující kapitoly). Předpokládá se, že struk-

tura programu a seznam vzniknou při použití pravidel strukturovaného programování. Tomu je zápis přizpůsoben.

Struktura programu

Za strukturu programu považujeme stromový graf, jehož uzly jsou obdélníčky se jménem charakterizující příslušnou činnost - - příslušnou komponentu (viz příklad na konci textu). Uzly - čtverečky - jsou dvou druhů : složené komponenty (které jsou tvořeny z komponent nižší úrovně) a operace (přímo v programovacím jazyce realizovatelné). Složené komponenty jsou buď prováděny v sekvenci (čtvereček není označen), nebo jsou opakované (čtvereček je označen hvězdičkou a číslem podmínky ukončující opakování), nebo jsou vybírané (z několika komponent téže úrovně, čtvereček je označen kroužkem a číslem podmínky, jejíž platnost určí, že komponenta se provede). Název složené komponenty má formu identifikátoru, tj. začíná písmenem. Operace tvoří listy stromového grafu (tj. nejsou dále rozkládány), ve čtverečku je zapsáno číslo operace, resp. několik čísel operací oddělených čárkami. Struktura programu se zapisuje pomocí čísel úrovní podobně jako struktura záznamu v Cobolu či PL/1.

Tak zápis struktury minipříkladu je :

```
1    TABLE
2    1
2    TABLEBODY
3 *  LINE        C1:
4    6,3
4    LINEBODY
5 *  NUM         C2:
6    5,4
4    7,2
2    8
```

Při zápisu komponenty po čísle úrovně následuje typ (u sekvence se neuvádí), pak název komponenty, a u iterací a výběrů ještě číslo podmínky.

Zápis výběru je :

```
5   VÝBĚR
6 O VOLBA1   C4:
6 O VOLBA2   C5:
6 O OSTATNÍ  C6:
```

kde "O" označuje, že to je vybíraná komponenta. Jestliže podmínka C4 je splněna, provede se VOLBA1, je-li podmínka C4 nesplněna a podmínka C5 splněna, provede se VOLBA2 atd.

Seznam podmínek a operací

Podmínky se označují písmenem C (condition) a číslem zakončeným dvojtečkou (např. C2:), operace se označují číslem zakončeným tečkou (např. 5.), nebo jsou uvedeny písmenem S (statement -- např. S5.). Zápis operace (i podmínky) se uvádí ve speciálních závorkách, za těmito závorkami jsou uvedeny příkazy programovacího jazyka operaci (podmínku) realizující. Příklady :

```
*< C2: test na konec řádku >*
     IF COL-NO > LINE-NO
*<   5. vypočti tištěné číslo >*
     MULTIPLY LINE-NO BY COL-NO GIVING NUM (COL-NO)
```

Generace programu ze struktury

Ze struktury programu generuje MEPRO kostru programu, do které vkládá příkazy realizující operace (či podmínky). Struktura programu plně určuje předání řízení; všechny skoky a cílová návěští jsou generovány, takže programátor nemá příležitost udělat chybu. Jako příklad použijeme část minipříkladu, komponentu LINEBODY. Generací vznikne :

```
LINEBODY-ITR.
    IF COL-NO > LINE-NO
              GO TO LINEBODY-END.
    MULTIPLY LINE-NO BY COL-NO GIVING NUM (COL-NO)
    ADD 1 TO COL-NO
    GO TO LINEBODY-ITR.
LINEBODY-END.
```

V této ukázce je ze seznamu operací a podmínek převzata podmínka C2 a operace 5 a 4 - slovesa Multiply a Add. Jména paragrafů a skoky jsou generovány ze struktury. U generace v jazyce Assembler jsou poměry poněkud jiné, návěští jsou kratší a lze využít makrojazyk. Generovaný program a programová struktura korespondují v Cobolu pomocí jmen paragrafů, v Assembleru pomocí poznámkových štítků.

Pokud je úloha rozsáhlá, lze jednou strukturou zapsat její podstatnou (hlavní) část, k realizaci složitějších operací můžeme přistoupit jako k samostatným úlohám a realizovat je samostatnými strukturami.

Dokumentace programu

Ze struktury a seznamů se automaticky vyrábí dokumentace - - v současné době v operačních systémech DOS/3 a OS prostřednictvím edičního systému PES. Podle struktury programu vytiskne tiskárna stromový graf a vypíše seznam operací a podmínek v rozumné grafické úpravě. Tuto automaticky vyráběnou dokumentaci lze doplnit slovním popisem, který lze zařadit do zdrojového textu tak, že při generaci programu je ignorován, převezme se pouze do dokumentace.

Takovou dokumentací jednoduchého příkladu je poslední strana tohoto článku. Je zde přímo reprodukován výstup z tiskárny počítače (listing).

Generace mimo strukturu

Do zdrojového textu lze zařadit úseky, které jsou beze změny přejímány do generovaného programu, dokumentací však jsou ignorovány. Tak lze zařadit například popisy dat ap.

Programování

Hodnocení vlivu systému MEPRO na programování vychází jednak ze zkušeností se strukturovaným programováním vůbec, jednak z přímého používání tohoto systému. Tak třeba MEPRO samo je naprogramováno pomocí sebe sama. Obecně technologie strukturovaného pro -

gramování poněkud stírá rozdíl mezi analytikem a programátorem. MEPRO v podstatě nahrazuje profesi kódovače. Nyní zhodnotíme jednotlivé etapy života programu (uvedené v úvodu článku).

1a. Analýza problému a návrh programu

Technologie strukturovaného programování nám dává jasný prostředek pro zadávání úloh - datové struktury. Z nich již víceméně rutinně vzniká struktura programu. Zápis programové struktury stromovým grafem poskytuje prostředek přehledného popisu programu. Ze struktury programu získáme přehled jak o programu samotném (jak je zapsán), tak i jeho dynamickém průběhu. Tisk grafu struktury na tiskárně omezuje jeho rozměry na rozumnou velikost (velké struktury můžeme rozdělit). Automaticky vzniká dokumentace, která se uplatní v dalších etapách.

1b. Zapsání programu v programovacím jazyce

Zápis programu se redukuje na zápis operací, jejich sestavení v programu je automatické. Všechny skoky a návěští jsou generovány, čímž je splněn jeden z požadavků strukturovaného programování.

2a. Odstranění syntaktických chyb

Pořizování programu je podstatně redukováno. Každá operace či podmínka se zapisuje jen jednou, zápis struktury pomocí čísel úrovní je velmi stručný, je tedy méně příležitostí k omylům a překlepům.

2b. Odstranění logických chyb

Samozřejmě žádný systém nechrání úplně proti omylu a nedopatření. MEPRO poskytuje aktuální zápis struktury programu, který je názorný a který dává přehled o průběhu programu, takže logické chyby jsou většinou při pohledu na strukturu zřejmé. Pokud nesouhlasí výstupy programu s požadovanými, tak protože struktura programu je adekvátní struktuře dat, můžeme ihned určit, v které části programu musíme provést změnu. Úpravu provedeme buď změnou vyjádření operace v programovacím jazyce, nebo změnou struktury, nebo obojím. V každém případě dostaneme zbrusu nový program, není nutnost (a vlastně skoro ani možnost) dělat záplaty. Díky zkrácení této etapy (a etapy 1b) je vznik programu zřetelně urychlen.

3. Opravy a úpravy programů

Ze zdrojového textu může kdykoliv vzniknout hodnotná autorská dokumentace, které při znalosti technologie strukturovaného programování může rozumět kdokoliv, nejen autor programu. Je jen třeba dbát na mnemonika názvů, resp. na jejich vysvětlení. Nutná vysvětlení jsou součástí zdrojového textu, a tedy dokumentace. O realizaci úprav se mluví v předchozím odstavci. Úprava programu je snadná, takže to až svádí k vytváření většího počtu verzí.

Závěr

Technologie strukturovaného programování, ze které vychází systém MEPRO, naznačuje další směr ve vývoji programování - přenechání mechanických prací počítači a možnost věnovat se výhradně tvůrčí činnosti. V současné době je systém MEPRO provozovatelný v operačních systémech DOS/3 a OS a jazyky Assembler a Cobol. Jeho použití prokázalo, že je vhodný jak pro uživatelské úlohy (zpracování hromadných dat), tak i pro tvorbu komponent systému.

Příklad generace v Cobolu. Text příkladu je převzat z knihy M. A. Jacksona PRINCIPLES OF PROGRAM DESIGN. Vpravo je datová struktura, níže seznam podmínek a operací a programová struktura.

C1: IF LINE-NO > 10	4. ADD 1 TO COL-NO
C2: test na konec řádku	5. vypočti tištěné číslo
1. MOVE 1 TO LINE-NO	6. mezeruj
2. ADD 1 TO LINE-NO	7. vystup řádek
3. MOVE 1 TO COL-NO	8. konec programu