Oberon

Václav Snášel, Vladimír Sklenář

1.Úvod

Profesor Niklaus Wirth se do podvědomí programátorů celého světa dostal díky jazyku Pascal, jehož je tvůrcem. Zájem tohoto významného odborníka v oblasti programování a informatiky o překladače, programovací jazyky a osobní stanice, jej vedl k tvorbě programovacích jazyků zároveň s vyvíjením hardwaru.

Nejdříve zkonstruoval osobní počítač Lilith. Během projektu se projevila omezení Pascalu jako systémového programovacího jazyku, což vedlo k vytvoření programovacího jazyku Modula a později i Modula-2 s cílem umožnit podporu softwaru napracovních stanicích Lilith. Později byly vytvořeny osobní počítače Ceres-1, Ceres-2, resp. Ceres-3, které používaly operační systém Medos implementovaný ptávě v Module-2.

V roce 1985 začal Wirth společně s Gutknechtem pracovat na zcela novém systému s cílem dosáhnout větší extenzibility a flexibility. Wirth, uchvácen přesností a spolehlivostí kosmické sondy Voyager, která v době vytváření projektu míjela Oberon, jeden z měsíců Uranu, nazval projekt na jeho počest.

Původní projekt byl vypracován pro pracovní stanice Ceres. V současné době existuje systém Oberon pro mnohé platformy a operační systémy. MacIntosh, IBM RS/6000, DEC stanice, IBM PC/386 kompatibilní zde, jej můžeme provozovat pod MS-Dos a Windows, SPARC stanice. Tyto implemantace Oberonu jsou volně dostupné v síti Internet na adrese neptune.inf.ethz.ch.

Projekt Oberon v sobě zahrnuje operační systém a programovací jazyk. Proto budeme dále mluvit o jazyku Oberon a systému Oberon.

2. Systém Oberon

V systému Oberonu chybí pojem hlavního programu. Základní proveditelná jednotka je příkaz. Termín příkaz naznačuje akci v abstraktní datové struktuře. Tedy, kdokoli píše program v systému Oberon - píše příkaz.

Systém Oberon je řízen pomocí událostí. Smyčka událostí (Event Loop) je centrální komponenta celého systému.

Hlavní rysy systému Oberon jsou následující:

- je založen na objektově orientovaném programování,
- užívá dynamické linkování (přidává moduly k již běžícímu programu),

- text se chápe jako abstraktní datový typ,
- systém obsahuje nástroje pro vývoj programů, editování textu a grafiky,
- automatický garbage collection,
- má rychlý jednoprůchodový kompilátor, který dává uživateli téměř iluzi interpreta,
- charakterizuje síťové uživatelské rozhraní, tzv. tool viewer, který je směsí menu a příkazových řádků,
- jsou zrušeny jednolité aplikace s jejich pomalu nahrávajícími se procesy, jejich místo je nahrazeno sadou příkazů,
- systém je rozšiřitelný, uživatel může kdykoli přidat nové vlastní příkazy,
- je zrušen rozdíl mezi příkazem a vstupem textu.

Příkazy se nevypisují, ale využívá se již vyobrazených textů v textové části obrazovky a odtud jsou příkazy vybírány přímo.

Systém Oberon se liší od obvyklých operačních systémů v několika rysech:

- chybí pojem programu; volání procedury je, na místo aktivace z programu, jednotka akce specifikována operátorem
- každé volání procedury (příkaz) je atomická akce v dialogu mezi operátorem a počítačem: přepíná z jedné úlohy na jiné události mezi příkazy uživatele spíše než mezi dvěmi libovolnými strojovými instrukcemi
- příkazy se zadávají z textů a z jiných druhů dokumentů spíše než z klávesnice. Místo psaní přímo na obrazovku příkazy generují ne nestálý výstup ve formě zobrazených struktur dat
- rozhraní mezi dvěmi po sobě následujícími akcemi se skládá z abstraktních datových struktur
- Oberon poskytuje rozšířenou interpretaci příkazů
- Oberon charakterizuje jednoduchý a neobyčejně výkonný systém. Adresář disku je organizován jako B-strom.
- moduly jsou nahrávány pod Oberonem jen tehdy, jsou-li aktuálně užívány. Průběžné nahrávání je významné, protože balíky se mohou skládat z textů modulů, z nichž jenom některé jsou využity pro nějakou specifickou aplikaci. Průběžné nahrávání je kontrolováno stránkovými chybami , které jsou vyvolány mechanismem virtuálního adresování
- systém a uživatelské balíky jsou implementovány v jazyce poskytujícím rozšíření datových typů a polymorfní operace se zárukou bezpečí typů
- implementace systému Oberon může být rozšířena deklarací nových datových typů. Objekty rozšířených typů jsou kompatibilní s objekty jejich základního typu a proto mohou být integrovány k existujícím datovým strukturám
- v systému Oberon není reálný rozdíl mezi uživatelem a programátorem. Uživatel, který má k dispozici silnou základnu modulů, může rozšířit systém nebo jej adaptovat ke svým potřebám programováním nových nástrojů.

3. Jazyk Oberon

Vývoj tohoto nového jazyka je opakem obecných praktik a trendů, ale má nevyčíslitelné výhody.

Programovací jazyk Oberon se vyvinul z projektu, jehož cílem bylo vytvořit moderní, pružný a výkonný operační systém pro pracoviště s jediným uživatelem. Hlavním úkolem bylo soustředit se na vlastnosti, které jsou opravdu nutné, a jako důsledek vypustit vlastnosti postradatelné.

Původně tvůrci plánovali systém vytvořit v Module, protože tento jazyk účinně podporuje teorii modulového vzoru a operační systém by měl být navržen ve stylu samostatně kompilovatelných části, měl by být řadou základních modulů. Aplikace pak musíme považovat za cílené rozšiřování základní řady. Moderní jazyky, jako Modula, podporují rozšiřitelnost především v procedurální oblasti, zatímco jsou omezenější ve sféře datových typů. Např. Modula nepřipouští definici nových datových typů jako rozšíření jiných již definovaných typů. Problém se pak řeší pomocnými programy.

Koncept plánovaného operačního systému také vyžadoval vysoce dynamickou centralizovanou správu paměti spoléhající na techniku úklidu volných míst v paměti (garbage collection). Brzy se ukázalo, že pravidlo soustředit se na podstatné a vyloučit nepodstatné, se musí aplikovat i na jazyk, ve kterém je operační systém vytvářen. Jazyk Oberon vznikl zjednodušením jazyka Modula a doplněním o některé nové rysy jako například objektově orientované konstrukce.

Vlastnosti zavedené do Oberonu

Rozšíření typů (dědičnost).

Nejvýznamnějším dodatkem je zavedení rozšířených záznamových typů, což umožňuje konstrukci nových typů na základě typů již existujících a zakládá určitý stupeň kompatibility mezi typy starými a novými.

Např. již definovaný záznam

```
T = RECORD
   x, y : INTEGER
END
```

může být rozšířen následujícím způsobem

```
T0 = RECORD(T) T1 = RECORD(T)
   z : REAL nebo w : LONGREAL
END END
```

na typy s položkami x, y, z, resp. x, y, w.

Obecně tedy definujme typ

```
T = RECORD(T)
   <definice rozšiřujících položek>
END
```

který je (přímým) rozšířením typu T a opačně, T je (přímým) základním typem T'. Samozřejmě rozšířené typy mohou být rozšiřovány dále.

Pravidlo kompatibility přiřazování stanovuje, že hodnoty rozšířených typů lze přiřadit k proměnným jejich základních typů. Např. tedy záznam typu T0 může být přiřazen proměnné základního typu T (a ne naopak). Přiřazení se provede jen pro společné položky, ostatní jsou ignorovány.

Tento koncept rozšiřitelných datových typů nabývá na významu při rozšiřování na ukazatele (pointery). Pointer P' vázaný na T' rozšiřuje pointer P, je-li P vázáno na základní typ T typu T'. Rozšíříme-li pravidlo přiřazování, aby pokrylo i tento případ, je možné tvořit datové struktury,

jejichž uzly jsou různých typů. Tato nehomogennost je automaticky vázána tím, že uzly jsou spojeny ukazateli společného základního typu.

Vyhrazená slova jazyka Oberon se skládají výlučně z velkých písmen a nesmí být užita v roli identifikátorů.

```
+ := ARRAY IN THEN
- ^ BEGIN IS TO
* = CASE LOOP TYPE
/ # CONST MOD UNTIL
_ < DEFINITION MODULE WHILE
& > DIV NIL WITH
.    <= DO OF
, >= ELSE OR
; .. ELSIF POINTER
| : END PROCEDURE
( ) EXIT RECORD
[ ] IF REPEAT
{ } IMPORT RETURN
```

Vzhledem k rozsahu práce nebudeme uvádět další podrobnosti o jazyce Oberon.

Na závěr uvedeme ukázku jednoduchého modulu v jazyce Oberon, který rozšiřuje sadu příkazů o příkazy definované v tomto modulu.

```
(* BConfiguration.Mod.Bak*)

IMPORT Oberon, MenuViewers, TextFrames;
CONST LogMenu= "System.Grow.Edit.Locate Edit.Store";
            StandardMenu= "Edit.Store";
VAR
  X, Y: INTEGER;
  V: MenuViewers.Viewer;

PROCEDURE Basics;
BEGIN
  Oberon.AllocateSystemViewer (0, X, Y);
  V := MenuViewers.New(TextFrames.NewMenu ("System.Log", LogMenu),
    TextFrames.NewText(Oberon.Log, 0), TextFrames.menuH, X, Y);
  Oberon.AllocateSystemViewer (0, X, Y);
  V := MenuViewers.New(
    TextFrames.NewMenu ("System.Tool", StandardMenu),
    TextFrames.NewText(TextFrames.Text ("System.Tool"), 0),
    TextFrames.menuH, X, Y);
END Basics;
```

```
PROCEDURE NextViewers;
BEGIN
  Oberon.AllocateSystemViewer (0, X, Y);
  V := MenuViewers.New(
    TextFrames.NewMenu("Programmers.Tool", StandardMenu),
    TextFrames.NewText(TextFrames.Text ("Programmers.Tool"),
    0),TextFrames.menuH, X, Y);
END NextViewers;

PROCEDURE DOSColors;
BEGIN
  Display.ReplConst(Display.white,0,0,1000,1000,Display.replace);
  Display.SetMode (0, 2);
END DOSColors;

BEGIN
  Basics;
  NextViewers;
  DOSColors;
END Configuration.
```

4. Závěr

V roce 1991 byl zahájen projekt Oberon 3 jeden z jeho hlavních cílů je práce se supertextem to je s textem integrovaným s grafickými operacemi.

Vzhledem k tomu, že Oberon je volně dostupný přes FTP v sítíl Internet je poměrně snadné se s tímto systémem seznámit. O celémÍ projektu Oberon existuje rozsáhlá literatura. Dále chystáme dokumentaci k Oberonu v českém jazyce.

Systém Oberon budeme využívat pro výuku překladačů, OOP a moderních operačních systémů.

Literatura :

J.Gutknect The Oberon Guide. System Realease 1.2 , Dep. fur informatik, ETH Zurich, 1990

J.Gutknect: Oberon System 3: Vision of a future Software Technology., Software-Concept and Tools (1994) 15, 126-33

M.Jelínek: Systém Oberon. Diplomová práce UP Olomouc, 1994.

M. Reiser: The Oberon System. User Guide and Programmer's Manual., Addison Wesley, 1991

M.Jelínek, V.Snášel : Systém Oberon. Bakalář, v tisku.

N.Wirth: Ceres-Net: A Low-cost Computer Network Software-Practice and Experience 20,1 (1990) 13-24

N.Wirth: From Modula to Oberon Software-Practice and Experience 18,7 (1988) 661-670

N.Wirth: The Programming Language Oberon Software-Practice and Experience 18,7 (1988) 671-690

N.Wirth and J.Gutknect: Project Oberon. The Design of an Operating System and Compiler. Addison Wesley, 1992

N.Wirth and J.Gutknect: The Oberon System Software-Practice and Experience 19,9 (1989) 857-893

R.Sethi: Programming Languages, Concept and Construct, Addison Wesley, 1989

Autor :

RNDr. Václav Snášel CSc,
UP Olomouc, KI
Tomkova 40
77000 Olomouc

RNDr. Vladimír Sklenář
UP Olomouc, KI
Tomkova 40
77000 Olomouc