Výuka OOP na Ostravské universitě

František Huňka

Se základním blokem výuky OOP u studentů informatiky začínáme ve 3 a 4 semestru. Studenti v prvních dvou semestrech absolvují základní kurz programování v Pascalu. Tento postup považujeme za výhodný ze dvou důvodů:

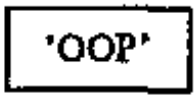
- všichni studenti již mají znalosti o práci s počítačem
- mají základní představu o postupech programování (strukturované, modulární).

Výuku OOP jsme se rozhodli začít s čistě objektově orientovaným systémem. Termín *čistě objektově orientovaný* systém reprezentuje jazyk a prostředí, které neumožňuje jiné než objektově orientované programování. Tyto požadavky splňuje Smalltalk/V, který je dostupný a funkční již dokonce od počítačů PC AT–286.

Nejdříve objasňujeme základní pojmy a to *objekt, třída, instance, zpráva a metoda.* V podstatě na těchto pojmech je pak vystavěna celá filosofie systému Smalltalk. Užíváme slovíčka systém, protože Smalltalk v sobě zahrnuje celé integrované prostředí. Asi třetinu času věnujeme objektově orientovaným principům a vlastnímu jazyku, další třetinu času integrovanému prostředí systému Smalltalk a zbytek času pak přehledu tříd a jejich metod. U klasických programovacích jazyků není výuka programování tak závislá na užití knihovny. U objektově orientovaných jazyků je tato závislost na počtu tříd významná a musíme počítat s jistou časovou investicí pro alespoň základní přehled. Tato investice se pak následně vyplatí při práci na dalších projektech.

Nejobtížnější bylo vysvětlit základní princip, to je že prostě existují jen objekty a zprávy. Celá činnost programu je realizovaná posíláním zpráv objektům. Např.

celé číslo – objekt



řetězec – objekt

pole – objektů

Např. výpočet faktoriálu 12 se ve Smalltalku realizuje:

12 Factorial

kde: *12* je objekt (příjemce zprávy)
Factorial je zpráva, která způsobí vyvolání metody.

Systém Smalltalk obsahuje tutoriál, kde jsou uvedeny základní příklady. Tím, že se propracujeme řadou dodaných příkladů, názorně vidíme, jak můžeme pracovat s plně objektově orientovaným jazykem a prostředím.

Rozdíl mezi Smalltalkem a Pascalem ilustruje následující elementární příklad se zadáním:

Uživatel zadá řádek textu, ten program načte a spočítá frekvenci výskytu jednotlivých písmen (velká písmena převádí na malá).

zápis v Pascalu

```
program a1;
const
      size=80;
var
      s: string[size];
      i,k: integer;
      c: char;
      f: array[1..26] of integer;
begin
      writeln('zadej text');
      readln(s);

      for i:=1 to 26 do
            f[i]:=0;
      for i:=1 to size do
            begin
            c:=asLowerCase(S[i]);
            if isLetter(c) then
                  begin
                  k:=ord(c)–ord('a')+1;
                  f[k]:=f[k]+1
                  end
            end;

      for i:=1 to 26 do
            write(f[i],' ')
end.
```

zápis ve Smalltalku

```
s c f k |


f:= Array new: 26.

s:=Prompter
      prompt:'zadej text'
      default: ''.
1 to: 26 do:[:i|
      f at: i put: 0].
1 to s size do: [:i|

c:=(s at: i) asLowerCase.
      c isLetter
ifTrue: [
      k:=c asciiValue
            – $a asciiValue + 1

f at: k put: (f at: k) + 1
                  ] ].
^f
```

Poznámka: V zápisu programu v Pascalu jsme použili dvě nestandardní funkce:

Funkce *asLowerCase(arg)* – je-li *arg* znak velkého písmene, funkce ho převede na malé písmeno.

Funkce *isLetter(arg)* – *je-li arg* písmeno, je hodnota funkce *true*, jinak *false*.

V uvedeném příkladu zatím nebyl ukázán žádný z efektivních zabudovaných stavebních bloků Smalltalku. Pokud bychom tyto zabudované objekty využili, program zapsaný ve Smalltalku by byl podstatně kratší:

```
s f |
s:= Prompter prompt: 'zadej text' default: ''.
f:= Bag new.
s do: [:c  c isLetter ifTrue: [f add: c asLowerCase]].
^f
```

Vysvětlení zápisu:

1ř. deklarace lokálních proměnných
2.ř načtení zadaného textu do proměnné
3.ř vytvoření instance *f* třídy *Bag*; třída *Bag* je funkčně podobná třídě množina (*set*), s tím rozdílem, že prvek obsahuje tolikrát, kolikrát se vyskytl
4.ř v cyklu se prochází celý zadaný text; je-li znak písmeno, převede se na malé písmeno a přidá se do instance
5.ř instance *f* vrací svůj obsah metodě, která ji volá.

Jak je vidět z uvedeného příkladu, záležitosti, které je třeba relativně komplikovaně zpracovávat v klasických jazycích, se dají ve Smalltalku provést velmi jednoduše.

Přínosem je také fakt, že se studenti seznámí s tzv. *browserem*. Česky bychom ho přirovnali k prostředku, který umožňuje prohlížení, (tříd, metod a zdrojových textů metod), editování (tvorbu nových metod) a jejich uložení. Obdobné prostředky totiž nalezneme i u vyšších verzí objektově orientovaných jazyků.

O výhodách výuky OOP prostřednictvím čistě objektově orientovaného systému jsme se již zmínili. Nyní k té druhé stránce, tedy nevýhody a problémy.

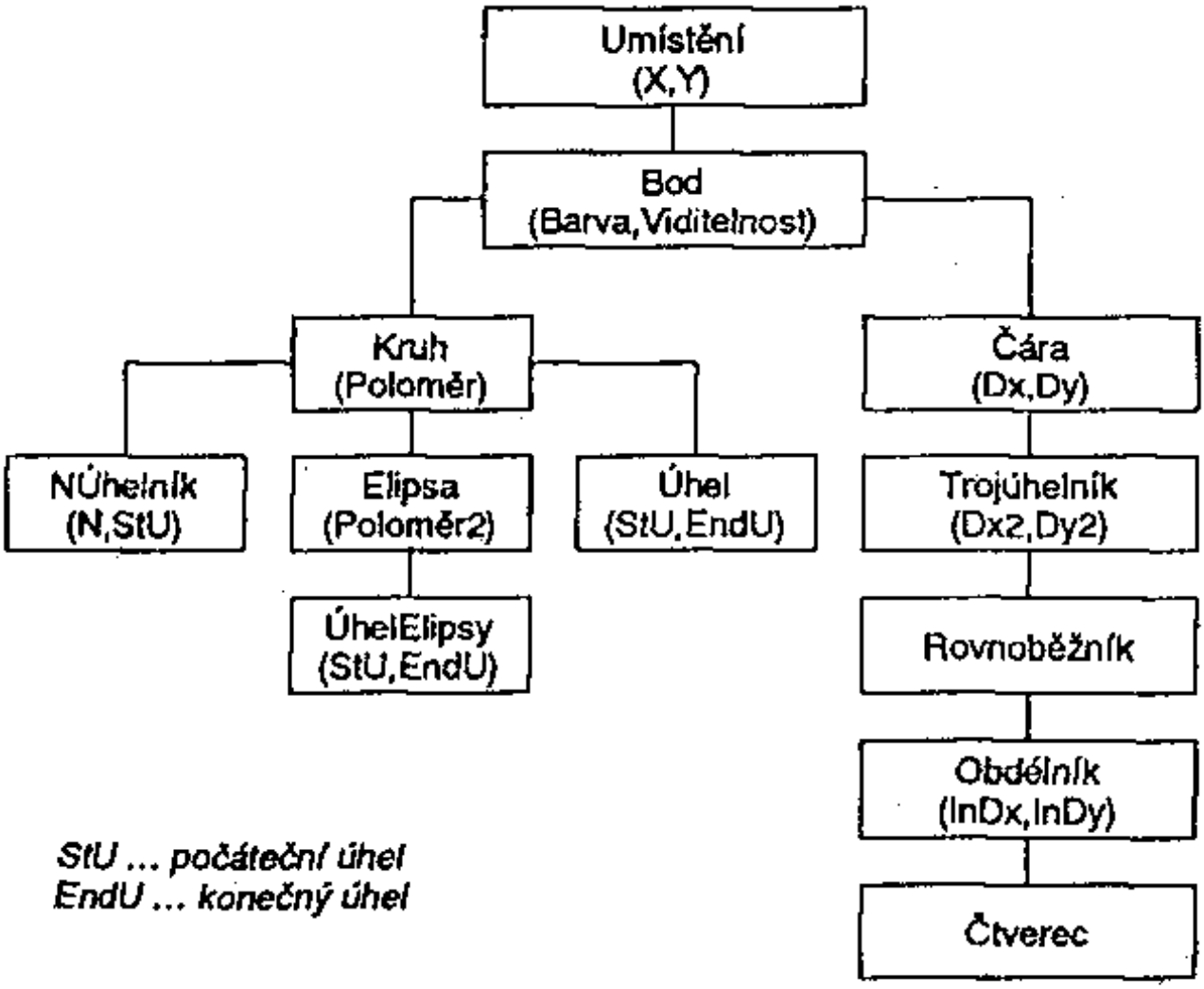
První největší problém spatřujeme ve změně tradičního přístupu myšlení. Především student sám musí chtít změnit svůj přístup. Zde jsme se často setkávali s tím, že změna způsobu myšlení byla nejobtížnější pro již „rutinované programátory", kteří již dostatečně ovládli nějaký klasický programovací jazyk. Kromě změny přístupu, to je že, vlastní programování probíhá formou posílání zpráv objektům, je zde ještě jeden nový prvek. Tím prvkem je objektově orientovaná knihovna. Činnost programování neznamená již

„vymýšlení vymyšlenčho" jako spíše „nalezení vhodnčho" (v našem případě vhodné metody). Současné potíže jsou v nedostatku příslušných manuálů, které byly zcela v angličtině, což byl pro některé studenty nezvyk. Pro zlepšení výuky by prospělo zavedení většího počtu samostatných úloh.

Na úvodní kurz ve Smalltalku navazuje OOP v Pascalu. Nejprve jsou probírány základní principy. Pro názornost jsme se soustředili na jednoduché grafické výstupy. Tam je tento postup nejzřetelnější. Uvedenou hierarchii tříd rozšiřujeme o další třídy, viz obr 1.

Cílem je vytvoření souboru ve tvaru *.TPU* , který slouží jako objektově orientovaná knihovna a vlastního programu, který vytváří složitější obrazce a využívá soubor *.TPU.* (Obdobný přístup při počátcích výuky OOP je zaveden na katedře matematické informatiky MU Brno).

Po tomto seznámení se s aplikací OOP v Pascalu, kde právě jednoduché grafické výstupy jsou nejnázornější pomůckou, následuje použití a využití Objektově orientované knihovny Turbo–Vision. Při výuce se používá názorných příkladů, které jsou součástí firemní literatury.



Obr. 1 Hierarchie tříd

Závěr

„Klasické programování" je jak velmi intelektuálně náročné, tak je náročnější na čas. V čistě objektově orientovaném prostředí spíše vybíráme, hledáme kombinujeme a testujeme, zda náš produkt odpovídá našim představám. Dost často se stává, že po vytvoření vlastní metody nalezneme metodu, která splňuje požadované cíle jako součást metod objektově orientovaného systému.

Tyto dva úvodní kurzy mají seznámit studenty se základními principy OOP. V dalším studiu pak tyto poznatku využívají při objektově orientované analýze a objektově orientovaném návrhu.

Literatura:

[1] Smalltalk/V (OOPS) Digitalk Inc. 1987

[2] Goldgerg A.,Robson D.:
Smalltalk 80 Addison–Wesley 1989

[3] Polák J.: Objektově orientované programování ČVÚT 1991

Autor: František Huňka