PROGRAMOVÁNÍ SE STÁVÁ MODELOVÁNÍM

Alena Lukasová

1. Od tradičního programování k umělé inteligenci

Informatika bezesporu patří v současné době k disciplínám rozvíjejícím se až příliš překotně, než aby zůstával čas na pozastavení se nad skutečností, že pojem programování od svého tradičního pojetí získal v poslední době poněkud širší záběr. Dříve jsme totiž od počítačů požadovali především to, aby za nás prováděly ty (v podstatě výpočtové) postupy, které jsme dobře znali, ale jejichž provedení by pro nás bylo neúměrně pracné. Dnes chceme, aby za nás počítače i myslely. V této souvislosti se sice používá pojmu umělá inteligence, ale ten je pro svůj velmi široký záběr pojmem spíše mlhavým.

Obecně řečeno, v informatice jde o zkoumání možností a zákonitostí přenášení lidské duševní práce "na bedra" počítačů. Jakého typu ale může být lidská duševní činnost ? Co tedy svěřujeme počítačům ? Sdělujeme jim své znalosti, jak provést určité duševní práce, ale též jim sdělujeme svá poznání o reálném světě a způsoby, jak z těchto poznatků odvodit jiné poznatky.

2. Počítačem modelujeme své znalosti o systému

Dříve jsme jsme si nutnost přistupovat k programování jako k modelování ani neuvědomovali. Modelovali jsme převážně některé znalosti procedurální, a to většinou odtržené od celku. Potom jsme často marně volali po znovupoužitelnosti programů. Modelovali jsme neúplně konceptuálně pojaté celky prostřednictvím algoritmů a ty pak programovali v programovacích jazycích procedurálního typu. Nabyli jsme dojmu, že smysl práce informatika je ve vyšetřování vlastností algoritmů odtržené od jejich sémantiky. Algoritmy stály v popředí našeho zájmu a pracovali jsme s daty, zdůrazňuji, pouze s daty.

Byl to rozvoj databázových systémů, který nás donutil dodávat datům jejich sémantický obsah a dělat tak z nich informace. Abstraktní datové typy, které se staly mocnými nástroji reprezentace znalostí deklarativního charakteru, dostaly zelenou. Tím se nám podařilo do svého modelování zatáhnout i ty znalosti, které nemají procedurální povahu, tj. znalosti deklarativní. Znalosti tohoto typu jsou totiž nezbytně nutné pro realizaci myšlenkového procesu zvaného usuzování.

Když už jsme si vědomi toho, že modelujeme, zabývejme se otázkou, co a jak modelujeme.

Úvahy o modelování je třeba začít u pojmů systém. Programování je třeba pojímat jako modelování určitého, patřičným způsobem zjednodušeného, systému. Entity neboli objekty našeho zájmu jsou v modelované části světa obvykle dobře rozpoznatelné, stejně tak, jak jsou rozpoznatelné jejich vzájemné vztahy a komunikace. To umožňuje poměrně přesnou korespondenci mezi modelovanou skutečností a jejím modelem.

Definovat systém znamená provést předtím jistou abstrakci, zanedbat věci nepodstatné a charakterizovat systém ve zjednodušené formě. Znamená to stanovit zájmové objekty, jejich vztahy a operace, které mezi uvažovanými objekty probíhají. Entity, které spolu se svými vlastnostmi a vzájemnými vztahy tvoří objekty našeho modelovacího zájmu, se deklarují. Deklarují se pomocí vyspělých určitým vhodným způsobem strukturovaných datových typů, jako jsou např. záznamy tvořící relaci.

3. Model a jazyk

Model odpovídá pojmové úrovni, jazyk je fixačním a komunikačním (nejen s počítačem) prostředkem modelu. Jazyk je uchopitelným a manipulovatelným nástrojem modelování.

Zatímco pojmem model se rozumí spíše konceptuální stránka přístupu k modelování určité skutečnosti, je jazyk jeho skutečným prakticky použitelným nástrojem.

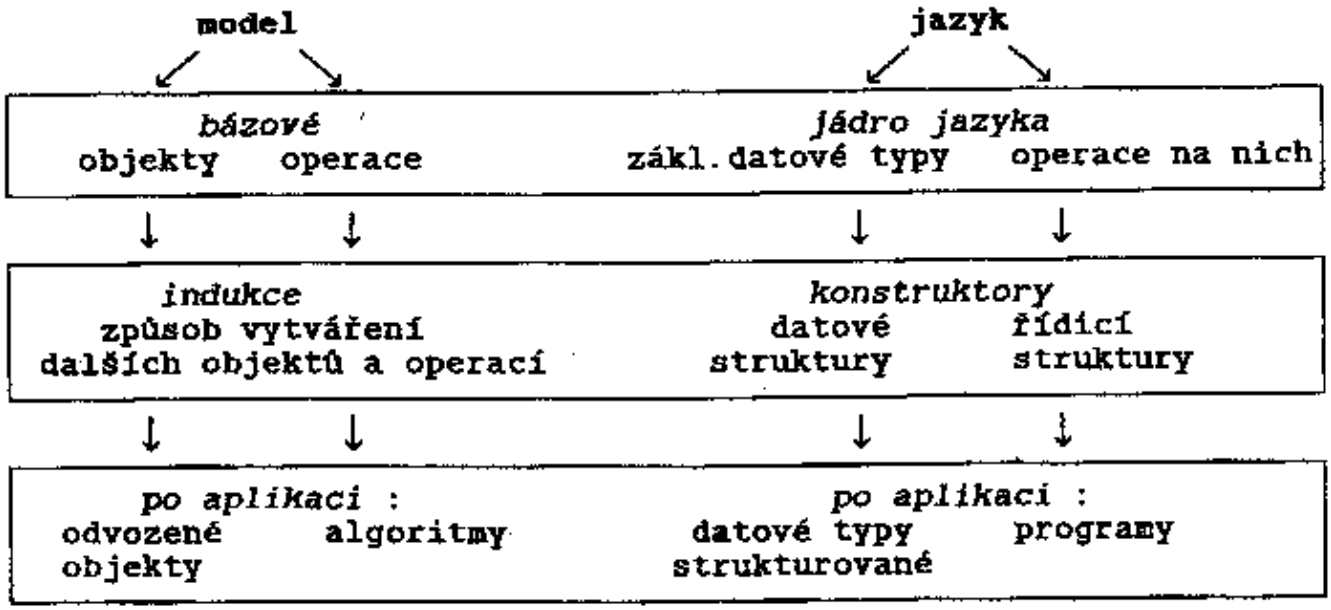
4. Co nám programovací jazyky poskytují pro naše modelování

Základem každého modelování je "vestavěná" část modelu, která obsahuje základní stavební prvky modelování. V případě modelování pomocí programovacích jazyků jsou jimi základní datové objekty a základní operace, které lze na datových objektech provádět.

Každý programovací jazyk má svůj základní vestavěný datový model pro reprezentaci odpovídajících modelovaných problémů. V případě jazyka PASCAL jej tvoří, jak je známo, základní datové typy. Model jazyka též obsahuje vestavěné základní operace na základních vestavěných datových typech.

Postrádá-li datový model jazyka vestavěné reprezentace datového modelu problému, je třeba reprezentovat požadovaný datový model pomocí abstrakcí podporovaných jazykem - datových struktur. Jimi se vytvářejí datové typy strukturované, které již mohou mít vysoký sémantický obsah. Podobně, chybí-li vestavěné prostředky pro reprezentaci operací na strukturovaných objektech, dopracujeme je tak, že vytvoříme za pomoci určitých konstruktorů modelu jejich algoritmy a ty pak programujeme.

Pojmy týkající se datového modelu a jazyka lze sobě vzájemně přiřadit. V případě tradičních programovacích jazyků by toto přiřazení vypadalo asi takto :



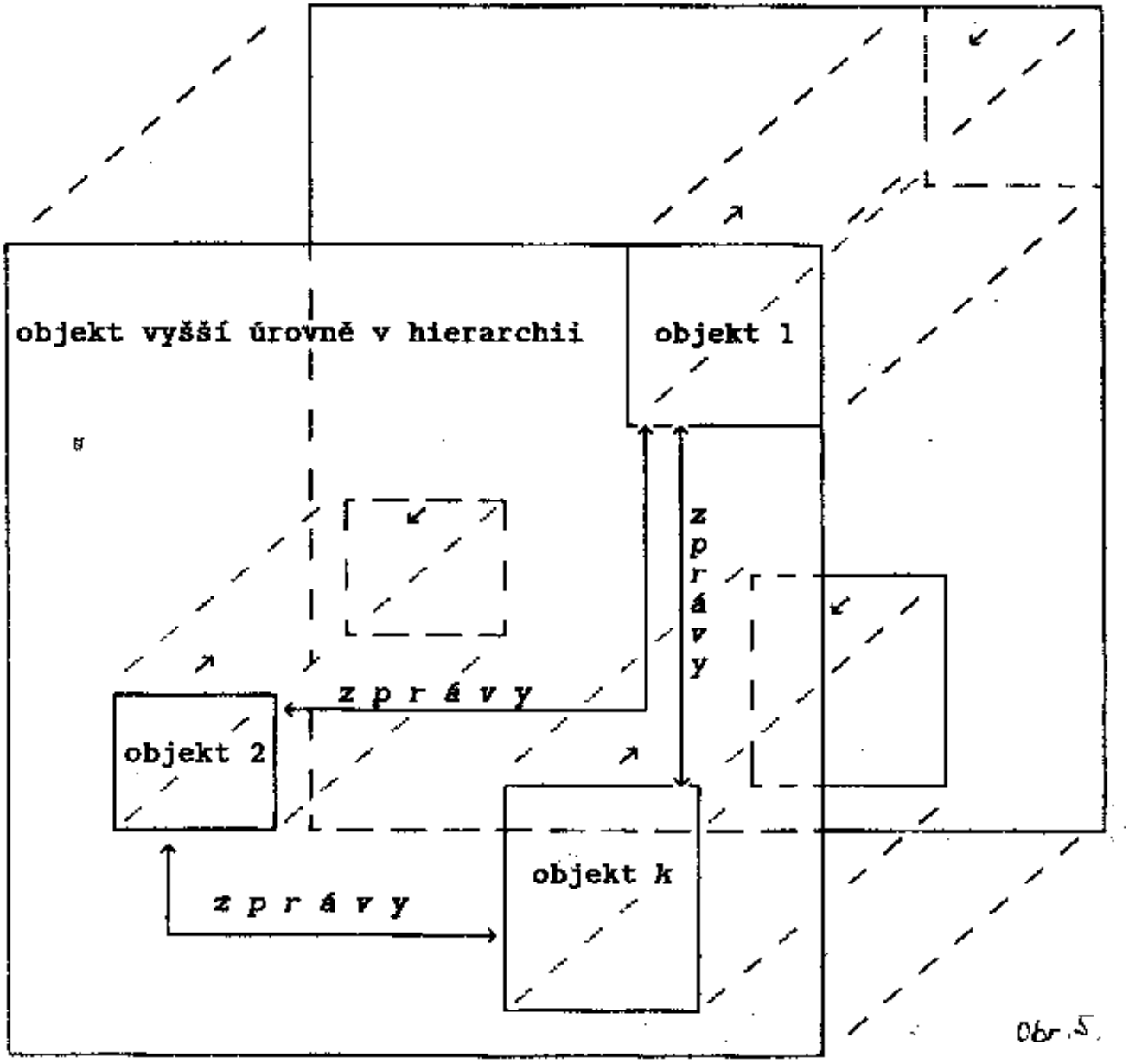
obr. 1

Modelování systému počítačem spočívá za prvé v reprezentaci statické části systému, tj. zobrazení deklarativních znalostí o objektech uvažovaného systému, pomocí jeho adekvátního datového obrazu. Za druhé pak spočívá ve své části dynamické představující popis určitých operací, které v systému probíhají. Návody k provádění operací jsou algoritmy, které se sdělují

počítači formou programu. V obecném smyslu jsou to opět data. Proto se často hovoří o datových modelech v obecném smyslu, zahrnujícím i operace na datech. Je-li stanoveno, co je datovým obrazem objektů lidského myšlení, tj. znalostí deklarativních i procedurálních, a jak lze datově zobrazovat myšlenkové pochody, je tím dán rámec pro datové modelování.

5. Jak do tohoto pojetí zapadají konkrétní programovací jazyky

Datové modely tradičních imperativních programovacích jazyků se sobě navzájem značně podobají. Jde v podstatě o modely algoritmicky orientované. Pro ně je charakteristický důraz na operační neboli funkční část modelu, přičemž část týkající se datových typů zůstává v pozadí a má pomocnou funkci (obr. 2).



Obr. 5

Datové typy, pro jejichž strukturování programovací jazyky poskytují bohatý výběr prostředků, představují pro operační neboli funkční složku řešení problému, tedy pro algoritmy, nutné "pozadí". Názorně řečeno, datové typy zpravidla slouží jako odkládací prostory pro manipulaci se vstupními daty za účelem jejich zpracování na data výstupní. Datové typy jsou existujícím nebo postupně vytvářeným datům "šity na míru", resp. data jsou vytvářena v takové formě, aby se "vešla" do pro algoritmus nejvhodnějších datových typů.

Samotná data jsou v konvenčním programovacím pojetí pouhou "vstupní surovinou", která nemá zpravidla povahu hotového výrobku. Prostor, v němž algoritmus operuje, nebývá uzavřeným světem, nýbrž komunikuje s okolím pomocí vstupů a výstupů.

Rovina funkčního modelování, tj. "popředí" je tedy v pojetí tradičního imperativního programování tou rovinou, v níž spočívá těžiště modelu.

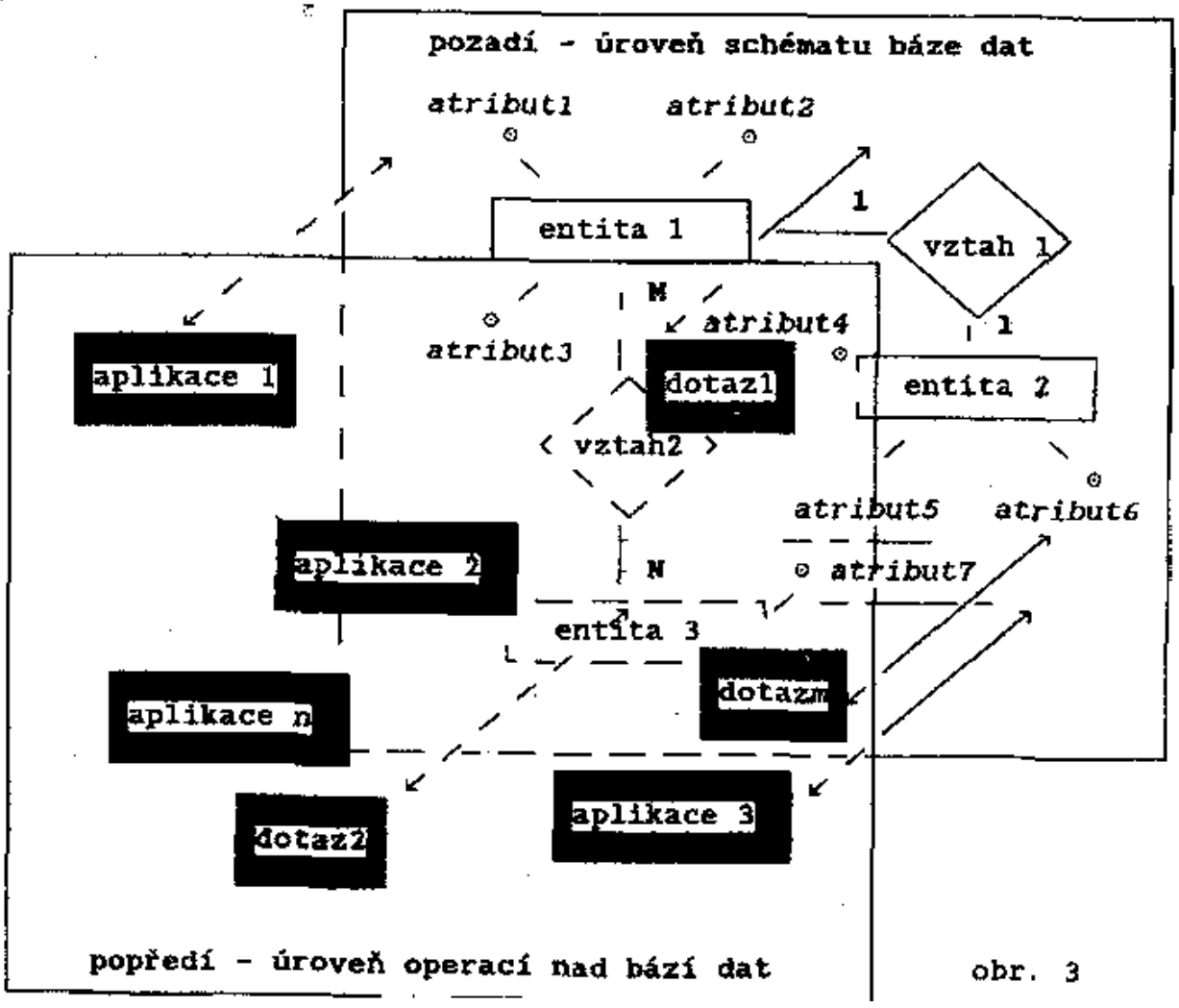
6. Existují modely a jazyky orientované výhradně na "pozadí" ?

Jistým protipólem modelů a jazyků algoritmicky orientovaných jsou modely a jazyky formální. Např. formální logika prvního řádu disponuje pouze statickou částí modelu, tedy části popisu objektů, nikoliv operací na nich. Jemu odpovídající jazyk vychází ze své báze, kterou tvoří symboly jazyka (symboly pro konstanty, proměnné, logické spojky, funkční, predikátové a pomocné symboly). Indukce je pak dána soustavou pravidel syntaxe jazyka.

7. Jak je to s databázovým modelováním a programováním aplikací

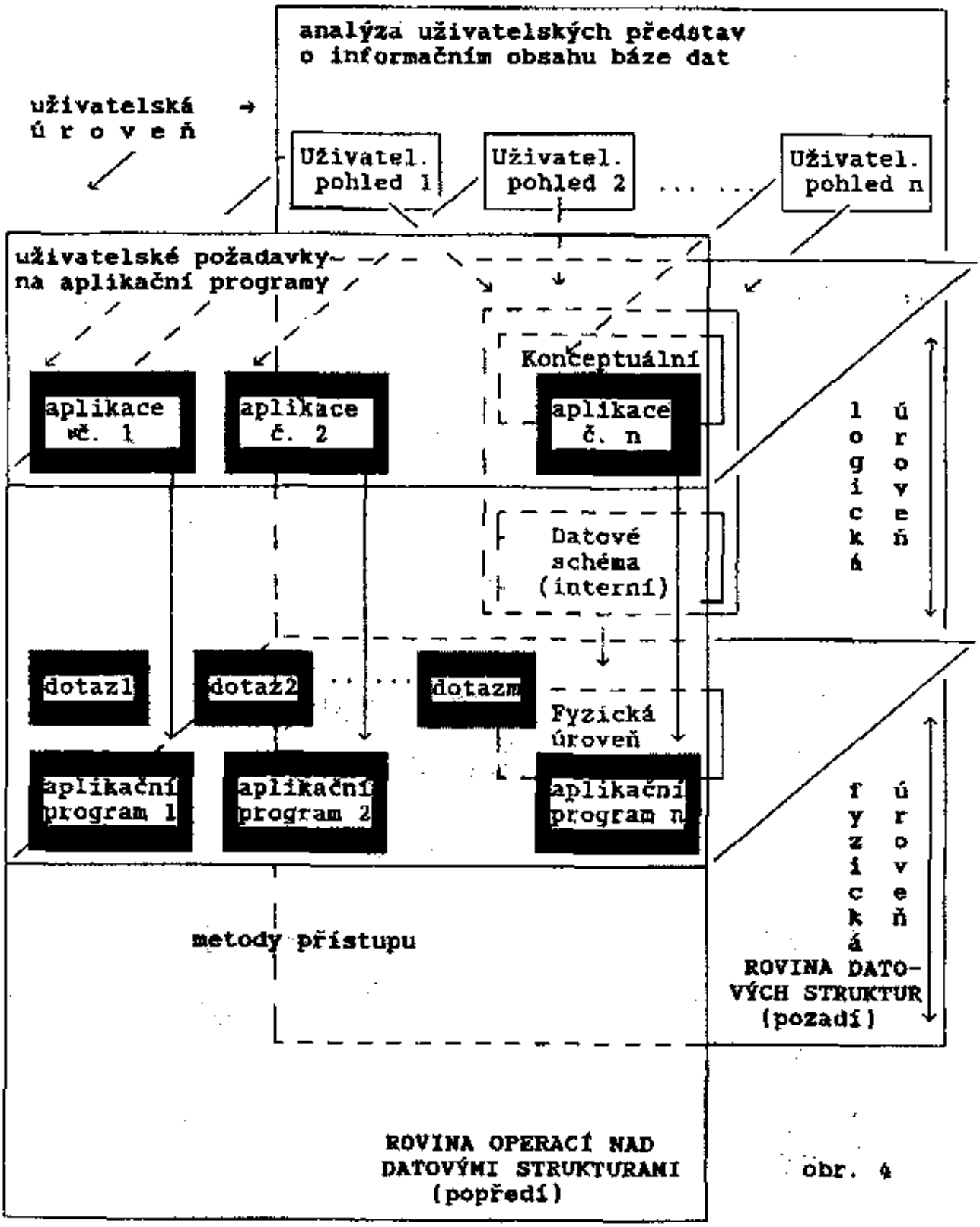
Databázová technologie od počátku svého rozvoje rozlišuje dvě úrovně, a to úroveň (pozadí), v níž se databáze navrhuje a později na základě navrženého schématu definuje, a úroveň programování databázových aplikací (popředí).

V databázovém světě se na pozadí spíše "modeluje", na popředí pak standardně "programuje". Báze dat, která představuje strukturovaný datový typ vysoké úrovně, nemá již jen pomocnou funkci pro algoritmické zpracování dat, jako je tomu zpravidla při konvenčním programování, ale sama o sobě je hotovým výrobkem - datovou základnou informačního systému. V případě databázové problematiky je těžiště systému nepochybně v rovině "pozadí" (obr. 3).



obr. 3

V databázovém světě je navíc zvykem rozlišovat i různé úrovně abstrakce, které se týkají jak pozadí, tak i popředí (obr. 4). Na pozadí představuje databázový model nejvyšší úroveň abstrakce. Ke konkrétnímu popisu struktur dat databáze a jejich vzájemných vazeb pak slouží, co do úrovně abstrakce situované poněkud níže databázové definiční jazyky. Ty představují určitou konkretizaci abstraktních databázových modelů. Vytvářet aplikační programy popředí lze v databázovém prostředí až poté, kdy byla databázovým modelováním vytvořena standardně strukturovaná báze dat jako jejich nutné pozadí.



obr. 4

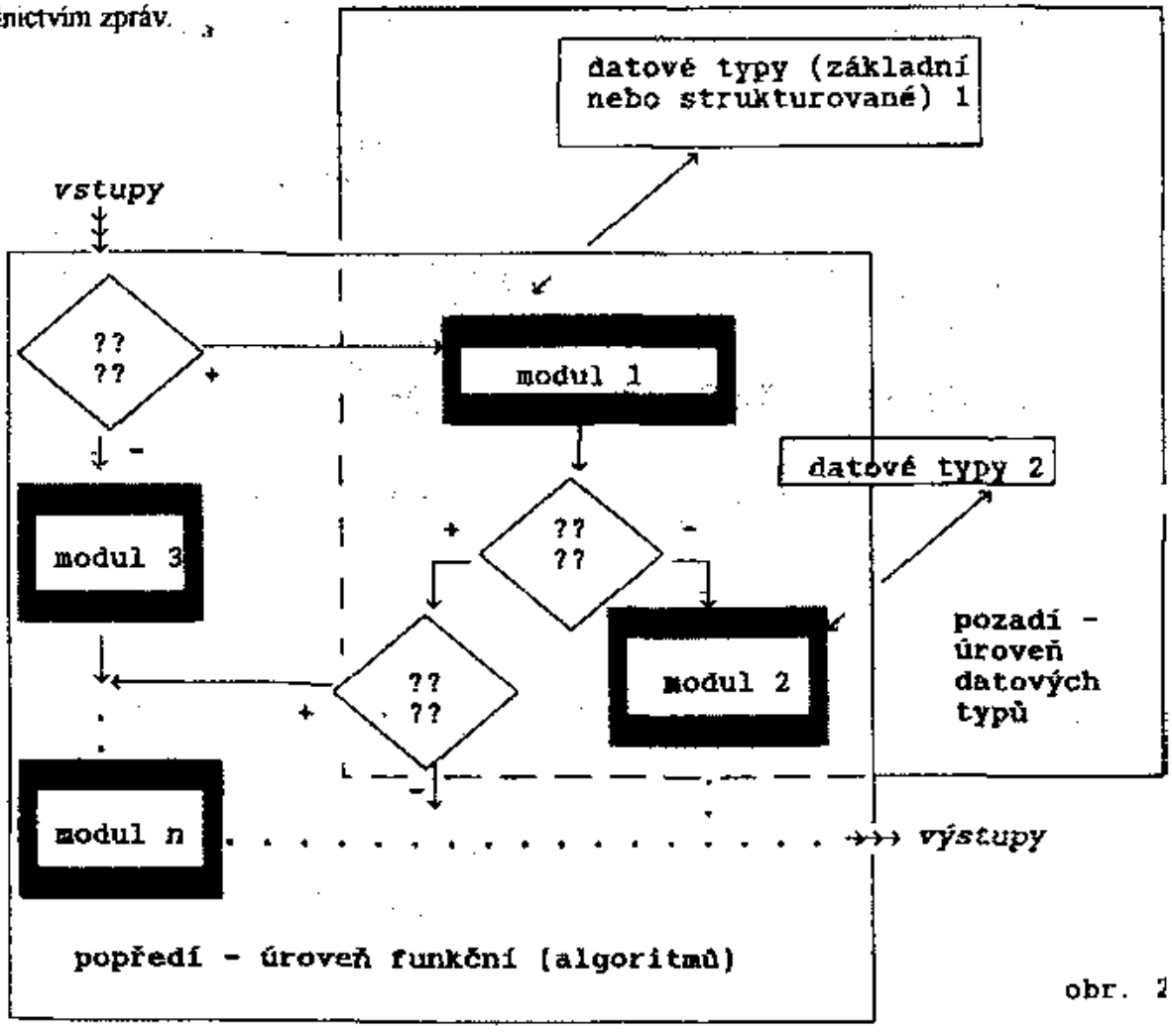
Pokud se u databází hovoří o programování, jde o programy pro manipulaci s informacemi, které jsou v ní obsaženy nebo z ní odvoditelné. A protože velká část těchto manipulačních úkonů má standardní podobu, byly pro ně v rámci databázových systémů vytvořeny odvozené operace, které mají povahu příkazů jazyka pro manipulaci s daty. Při programování databázových aplikací jsou tyto příkazy manipulačního jazyka bohatě využívány. Rovněž řada aplikačních programů - odvozených operací, jejichž účelem je získat určitou informaci z databáze, je standardního typu. To umožňuje tvůrcům databázových systémů vytváření neprocedurálních dotazovacích jazyků využívajících vlastností těch logických koncepcí, na nichž jsou příslušné modely, a tedy i jejich definiční jazyky, založeny.

8. Souvisí orientace jazyka na "pozadí" nebo "popředí" s jeho procedurálností ?

Značně. Co se podaří rozumným způsobem "vtělit" do dat, to se pak "ušetří" na algoritmech. Databáze (relační) je strukturovaný datový typ (lze ji tak nazvat ?) s vysokým sémantickým obsahem. Navíc umožňuje pracovat s operacemi vyššího typu, než jsou základní operace s daty (relačními operacemi). Pak už stačilo zkonstruovat ekvivalentní neprocedurální dotazovací jazyk.

9. Jak je to s objektovou orientací

Jednou z cest, jak minimalizovat procedurálnost jazyků, je objektová orientace příslušného modelu, z něhož jazyk vychází. Objektově orientovaní specialisté jsou si velmi dobře vědomi toho, že modelují. Proto zpravidla nedělají chyby typické pro dřívější tradiční programování, tj. nevytrhávají jednotlivosti které nelze pojímat jako objekty, z celku. Objekt, představující zpravidla menší a co do funkčnosti autonomní výsek světa objektů, má sám pro sebe své pozadí i své popředí. Objekt má samostatný "život" a možnost komunikace s ostatními objekty prostřednictvím zpráv.



obr. 2

Zdálo by se, že objektová orientace je v programování jedinou rozumnou orientací. Obecně to ale není pravda. Pro informační systémy modelující deklarativní znalosti relační povahy je stále ještě relační reprezentace tou nejvhodnější cestou. Proto databázoví specialisté předpovídají jako nejperspektivnější vývoj spočívající v kombinaci relačního základu (relačního databázového systému) s objektově orientovaným rozhraním, a to vše v architektuře klient - server.

10. Závěr

Programování v tradičním smyslu je dnes zjevně na ústupu, a to nejen vlivem vyspělých jazyků pro navrhování softwarových systémů. Svůj podíl na tom má i vývoj informatických koncepcí směrem k většímu důrazu na sémantiku programů a zpracovávaných dat.

Je-li sémantickou doménou programu určitý výřez reálného světa pojímaný jako systém se svými dobře identifikovatelnými entitami, jejich vlastnostmi, vzájemnými vztahy a prováděnými operacemi, je program vlastně počítačovým modelem tohoto systému. V tomto pojetí je algoritmus a z něho vycházející program (zpravidla v tradičním imperativním programovacím jazyce) modelem nějaké naší procedurální znalosti. Znalosti deklarativního charakteru, které jsou nezbytné pro tzv. umělou inteligenci, se naproti tomu modelují sémanticky bohatými datovými strukturami (např. záznamy tvořícími relaci). Objektová orientace již si je vědoma toho, že modeluje systémy, a proto neodděluje modelování znalostí deklarativního a procedurálního typu, nýbrž modeluje tyto systémy a jejich podsystémy z obou hledisek současně.

Rozvoj abstraktních datových typů s bohatým sémantickým obsahem umožňuje zjednodušit operace na nich prováděné. Jazyk používající těchto vyspělých datových typů k deklarování znalostí se pak může stát i jazykem dotazovacím (téměř) zbaveným procedurálnosti. To je vidět u jazyků databázových systémů a zvláště u logického programování. Mezi sémantickou bohatostí datových struktur a potřebou jazyka být procedurální zde existuje jakási nepřímá úměrá.

Programové systémy pro umělou inteligenci nepředstavují nějaký koncepční zlom v trendech programování. Jsou jen jedním z krajních případů - deklarativní znalosti uspořádané do vhodných vyspělých struktur zde redukují nutné operace na minimum (např. prohledávání seznamů, stromů).

Literatura

1. Aho, A.V.; Ullman, J.D.: Foundation of Computer Science Computer Science Press, 1992
2. Delobel, C.; Lécluse, Ch.; Richard, P.: Databases: From Relational to Object-Oriented Systems. International Thomson Publishing, 1995
3. Russell, S; Norvig, P: Artificial Intelligence. A Modern Approach. Prentice - Hall International, Inc., 1995
4. Scheurer, T. : Foundations of Computing Addison - Wesley, 1994

Doc. RNDr. Alena Lukasová, CSc.
Katedra informatiky Ostravské univerzity
Bráfova 7, 701 03 Ostrava 1
tel. 6222808/221