ZÁSADY KONCEPTUÁLNÍHO TOTÁLNĚ OBJEKTOVĚ ORIENTOVANÉHO MODELOVÁNÍ

Martin Molhanec

České vysoké učení technické – FEL, K-313
Technická 2, 166 27 PRAHA 6, Dejvice, Česká republika
tel.: (++420) 2 2435 2118
mailto: molhanec@fel.cvut.cz
http://martin.feld.cvut.cz/~molhanec

Abstrakt

Obsahem příspěvku jsou úvahy o správných zásadách konceptuálního totálně objektově orientovaného modelování. Autor nejprve definuje co je vlastním obsahem konceptuálního objektově orientovaného modelování. Bude se dále zabývat pojmy jako jsou objekt, vlastnost, podobnost, třída, struktura a souvislost.

TE log.5.: Poznej, co máš před očima, a co je ti skryto, se ti odhalí.
Neboť není nic skrytého, co nebude odhaleno.

1 ÚVOD

Snad si mohu dovolit říci, že jako tradičně v posledních letech se jeden můj příspěvek na této konferenci zabývá problematikou objektového přístupu, jeho chápání a způsobu jeho výkladu. Aniž bych si dělal nárok na neomylnost, a fakta obsažená v tomto příspěvku je nutné chápat jako osobní vhled autora na tuto problematiku, jsem přesvědčen o skutečnosti, že mé příspěvky na toto téma mají mezi, sic nepočetnou, odbornou veřejností kladné a zaujaté přijetí.

Motivem pro vznik mých příspěvků [1], [2], [3] a [4] na téma problematika objektového přístupu byla především skutečnost, že jsem při četbě různých učebnic vykládajících objektový přístup, v poslední době výlučně založených na výkladu UML, zjistil, že s jejich autory a jejich pojetím a výkladem četných pojmů bytostně nesouhlasím. Po srovnání jejich výkladu s výkladem objektově orientovaného modelování v referenčních publikacích samotných autorů UML [5] a [6] jsem s úlevou konstatoval, že referenční autorský výklad je správný a v souladu se způsobem, jak bych si podobné pojmy dovolil vykládat já sám. V čem je tedy problém? Problém je bohužel ve skutečnosti, jak původní správný výklad předních autorů objektových metodik překrucují přečetní epigoni živící se psaním přemnohých učebnic UML [7] a [8]. Pochopitelně nelze říci, že jsou výše uvedené učebnice špatné jako celek. Každá z nich obsahuje, jak správná fakta, tak fakta poněkud překroucená. Neexistuje striktně binární hodnocení – pravda a lež. K těm lepším učebnicím, dle mého úsudku, patří například [9] a [10].

Díky výše uvedeným skutečnostem, které se staly zdrojem těchto úvah, jsem se dostal ke snaze lépe proniknout do základních principů objektového modelování. Některé své úvahy na toto téma jsem již uveřejnil na konferenci Objekty 2004 [1] aniž by, jak se zdá, zasáhly širší veřejnost. Nicméně jeden konkrétní výsledek si dovolím na svůj účet připsat, a to příspěvek

mého kolegy Vojty Merunky [11] na téma „Normalizace v objektových databázích“, jehož motivací byly naše četné dialogy a pře na různá témata z oblasti objektového modelování.

2 OBJEKTOVÉ POJETÍ MODELOVÁNÍ

V oblasti modelování informačních systémů si objektové pojetí našlo cestu na slunce až koncem 80 a počátkem 90 let. Mezi přední představitele tohoto inovativního pojetí je třeba uvést následující autorské veličiny softwarového inženýrství: Coad a Yourdon [12], Booch [13], Rumbaugh [14] a Jacobson [15]. Každá z výše uvedených celebrit je tvůrcem své vlastní objektově orientované metodiky (výjimku tvoří autorská dvojice Coad a Yourdon). Protože však nejzákladnější princip – objektové paradigma – je ve všech pracích výše zmíněných autorů totožný – vznikl společný objektový model UML (Unified Modeling Language), jako výsledek práce tří výše zmíněných autorů, pánů Booche, Rumbaugha a Jacobsona, který se stal dominantním, ve své podstatě standardním modelem současné doby.

Problém výkladu objektového paradigmatu spočívá především ve způsobu výkladu samotného. V mnohých učebnicích se pro výklad používá programátorský přístup, tedy výklad na základě skutečnosti, že čtenář – student, již umí programovat v nějakém objektově orientovaném jazyku, například Pascalu či jazyku C++, a proto základní pojmy z oblasti objektového paradigmatu již zná. Tento přístup je však velice zavádějící! Přestože se objektové programování stalo motivací pro vznik objektově orientovaného modelování, nelze objektově orientované modelování chápat pouze jako obrázky, které nám dokumentují strukturu našeho programu! Tento chybný přístup, tak rozšířený mezi nekvalifikovanými vykladači objektového modelování i mezi studenty samotnými, byl pochopitelně výše zmíněnými autory překonán. Velice dobře totiž chápali skutečnost, že objektové modelování, není určeno být pouhou dokumentací našich programů, modeluje reálný svět kolem nás za účelem jeho popisu a porozumění a pouze na naší vůli je využití tohoto modelu jako prvotního zdroje pro jeho implementaci ve formě konkrétního informačního systému v některém konkrétním programovacím jazyce. O této skutečnosti svědčí, jak obracení se k základním filosofickým pojmům, například u autorů Coada a Yourdona [12], tak skutečnost, že implementace objektového modelu může být uskutečněna, jak objektovými, tak strukturovanými nebo relačními technikami, například u Rumbaugha [14].

3 KONCEPTUÁLNÍ MODELOVÁNÍ

V dalším textu budu namísto objektového modelování mluvit o modelování konceptuálním. Důvod je totiž ten, že analytický objektově orientovaný model je ve skutečnosti formou modelu konceptuálního. Starší formou konceptuálního modelu byl model entitně-vztahový (Entity-Relationship Model) [16], tak jak ho známe z databázového modelování. Anglické slovo conceptual překládáme českým slovem pojmový. Jedná se tedy o pojmový model.

Z faktů uvedeným v předešlé kapitole, lze vyvodit následující předpoklady:

- Konceptuální modelování se zabývá modelováním skutečností reálného světa, a proto musíme tento svět pochopit a popsat prostřednictvím pojmů z tohoto světa.

- Konceptuální modelování není žádným způsobem závislé na případné implementaci jím vytvořeného modelu.

Ve svém příspěvku na konferenci Objekty 2004 [1] se zabývám vymezením pojmu konceptuální modelování. Pojem vymezuji pomocí následujících tvrzení:

1. Objektem jeho zájmu jsou objekty v reálném nebo abstraktním světě (*objekty*).
2. Zajímá se o vlastnosti těchto objektů (*atributy*).
3. Zajímá se o identifikaci těchto objektů (*klíčové atributy*).
4. Jeho zájmem je klasifikace těchto objektu dle různých společných vlastností (*třídy*).
5. Jeho zájmem jsou vztahy mezi těmito klasifikačními třídami (*dědičnost*).
6. Jeho zájmem je vzájemná strukturalizace těchto objektů (*skládání*).
7. Zajímá se o vzájemné vztahy mezi objekty a o klasifikaci těchto vztahů (*vztahy obecné*).
8. Všechny tyto aspekty považuje pro daný model za *trvalé po dobu života* jednotlivých objektů.

Podívejme se nyní na některé výše uvedené pojmy podrobněji.

3.1 Objekty a jejich vlastnosti.

Pokusme se pojem objekt chápat intuitivně, koneckonců jako jeden z nejzákladnější pojmů objektového paradigma bude stěží moci být definován pomocí ostatních pojmů. V monografii Object-Oriented Analysis autoři Coad a Yourdon [12] konstatují: „Od poloviny 70 let se v oboru informačního modelování začal užívat pojem objekt velice nestandardním způsobem v porovnání s jeho používáním v předešlých stoletích. V metodách entitně-vztahového modelování (Chen [16]), informačního modelování (Flavin [17]) a sémantického datového modelování (Shlaer a Mellor [18]) se termínem objekt rozumí pojem, který reprezentuje jeden nebo více výskytů jsoucnosti z reálného světa“. O něco dále autoři uvádějí svojí definici pojmu objekt takto: „Objekt je abstrakcí něčeho v oblasti našeho zájmu, reflektující schopnosti systému o tomto udržovat informaci a/nebo s tím interagovat. Je to také množina jemu (objektu) přináležejících hodnot atributů a výlučných služeb.“.

Z odkazem na výše uvedené, můžeme nyní pojem objekt jednoduše definovat, jako něco co existuje v reálném nebo abstraktním světě, jako pojem, který má nějaký smysl. Objektem je všechno o čem lze něco říci, každý objekt je pojem a každý pojem se stává objektem v určitém smyslu slova. Termíny jako osoba, faktura, automobil, slovo nebo barva jsou určité pojmy, které mají svůj význam, ale také to jsou objekty, které mají své vlastnosti (atributy) a které jsou v zájmové oblasti našeho informačního modelování.

Nedomnívám se, že zejména programátoři a studenti pochopí hluboký smysl výše uvedeného. Pro ně objektem zůstává proměnná programovacího jazyka, která zabírá kus paměti v počítači a pryč s nějakou zbytečnou filozofií. Doufám, že mé úvahy najdou zájem aspoň u analytiků informačních systémů.

Protože o objektu můžeme něco užitečného sdělit, například, že osoba je Martin a má dlouhé vlasy, že barva je červená a slovo je podstatné jméno, lze intuitivné pochopit, že objekty mají své vlastnosti (atributy) a protože objekty interagují s okolním světem, ten je ale také tvořen objekty, koneckonců jsme zatím nic jiného nežli objekty a jejich vlastnosti nedefinovali, mají i své metody (služby, chování). Všimněme si, že zatím jsme nepotřebovali vůbec definovat pojmy jako je dědičnost nebo vztah! Objektový systém potřebuje pouze objekty a nic jiného!

Další zajímavá skutečnost nyní odhalená! Objekt Martin má vlastnost barva očí . Ale termín barva očí je něco co má svůj smysl, je to tedy pojem, je to tedy i objekt! V totálně objektovém paradigmatu nic jiného nežli objekty neexistuje! Vlastnosti objektů jsou také objekty a každý objekt může být také vlastností jiného objektu!

3.2 Relevance, aplikační doména a úhel pohledu.

Častou chybou začínajících analytiků je snaha o vytvoření komplexního modelu reality. Vysvětlení pojmu relevance začínám dotazem: „Kdo proboha potřebuje u osoby znát barvu očí! Je vlastnost barva očí relevantní (pro nás v tomto okamžiku zajímavá) například u informačního systému vysoké školy, který udržuje informace o studentech vysoké školy, o tom v jakém ročníku studují a jaké jsou výsledky jejich studia? Nikoliv!“. Pokusme se o intuitivní definice pojmů zahrnutých v názvu této podkapitoly.

- *Relevantní* je to, co je pro nás zajímavé, o čem je pro nás potřebné udržovat informaci, kterou bude využívat uživatel našeho modelu (systému).

- *Úhel pohledu* je takový vhled na reálný svět, který záměrně zahrnuje pouze relevantní pojmy potřebné v dané aplikační doméně.

- *Aplikační doména* je podmnožina reálného světa, která je pro nás zajímavá, která je cílem našeho studia, našeho popisu.

3.3 Neexistující vztahy.

Zajímavou skutečností je fakt, že jsme zatím při našich úvahách nepotřebovali definici něčeho jako jsou vztahy. Klasický analytik nepřekročivší doposud své programátorské začátky bere existenci vztahů za něco základního a prvotního! A vztahy pak rozděluje na dědičnost a skládání a prostý vztah vlastně nezná, protože se přece v jeho programovacím jazyce nevyskytuje. Nemluvě o kreslení jakýchsi šestiúhelníků, protože si proboha musí ty své objektové proměnné někam ukládat a nechápe, že konceptuální (analytický) model je model reálného světa a nikoliv jeho nedokonalé programové implementace! O nic lépe na tom nejsou analytici zrození z tvůrců relačních databází. Sice chápou existenci prostých vztahů, ale dědičnost přeci nepotřebují, protože ji v relační databázi mohou implementovat pouze prostřednictvím prostých vztahů a mnozí z nich neuznávají ani vztah M : N, který se také v implementaci musí realizovat prostřednictvím vazebních entit a dalších vztahů!

To, co je považováno za vztahy, není prvotní. Vztahem totiž v konceptuálním modelování nazýváme několik od sebe rozdílných vlastností vyplývajících ze vzájemných souvislostí mezi objekty, které mají společné toliko to, že se snaží množinu všech objektů podle různých hledisek utřídit (klasifikovat). Je zajímavé, že z tohoto úhlu výkladu je jedním z naprosto primárních vztahů vztah mezi objektem samotným a jeho vlastností (atributem), která je přeci také objektem!

3.3.1 Podobnost.

Pokud se podíváme na naší relevantní množinu objektů uvidíme, že některé objekty jsou si v něčem podobné. Mají totiž shodné množiny vlastností. Například všechny objekty, které udržují informaci o nějakých osobách mají z našeho úhlu pohledu stejnou množinu relevantních vlastností! Tuto vlastnost – vztah, protože je to vlastnost daná určitou souvislostí mezi objekty, nazvěme podobností. Objekty, které si jsou podobné, jsou stejného typu, a můžeme tedy hovořit o vlastnosti typovosti. Množinu objektů stejného typu nazvěme třídou (class). Protože každý pojem je objektem i pojem třída je objektem, množina tříd je také pojem, je metatřídou (metaclass) a je to také objekt, ...

Třída, která má nadmnožinu vlastností jiné třídy je jejím potomkem a třída, která má podmnožinu vlastností jiné třídy je jejím předkem. Ejhle – zrodila se dědičnost (inheritance).

3.3.2 Struktura.

Jak jsem již dříve definovali, vlastnost objektu je také objektem. Pokud se v množině vlastností našeho objektu objevují vlastnosti přináležející stejné třídě, můžeme jejich jednotlivé výskyty nahradit jejich množinou (kolekcí). Toto není problém, protože termín kolekce je pojem a proto je to dozajista taktéž objekt a tedy může být i vlastností objektu. Co je to tedy kolekce? Vzhledem k definici třídy uvedené v předešlé podkapitole je kolekce podmnožinou třídy! Třída je množinou všech objektů daného typu, zatímco kolekce je množina pouze některých objektů daného typu. V našem pojetí těch, které jsou vlastností nějakého objektu!

Sic! Totálně objektové paradigma je nesmírně duševně namáhavá záležitost! Nicméně jsme si nyní odvodili typ vztahu, který se obvykle nazývá skládání, celek-část, agregace, kompozice nebo obecně kontejner.

3.3.3 Souvislost.

Posledním typem vztahu je prostý vztah. Jak k němu dojít pomocí objektového paradigmatu?! Prostým vztahem může být například vztah mezi autorem a knihou, je to vztah M : N a není to vztah typu dědičnosti ani kontejneru. Nicméně, bezpochyby lze tvrdit, že výrok je autorem je zcela určitě vlastností, podobně jako tvrzení má autora. Obě výše uvedená tvrzení (výroky) dávají skrze své vlastnosti do souvislosti různé objekty, které nejsou stejného typu ani se ze sebe navzájem neskládají! A o to tu jde! Prostý vztah je jednoduše vlastností objektu! Jak překvapivé! A vztahem je potom vztah mezi objektem a jeho vlastností (objektem). V jednodušším objektovém pojetí, kde některé atributy objektu jsou chápany neobjektově, nám podstata prostého vztahu uniká! V takovém zjednodušeném pojetí se prostý vztah objevuje pouze mezi objektem a jeho objektovou vlastností, která je však chápána jako samostatný objekt ve vztahu a nikoliv jako vlastnost!

4 Závěr

V našem výkladu jsme se dostali k dočasnému konci, dočasnému z toho důvodu, že ve výkladu je možné dále pokračovat směrem k totálně objektovému výkladu další pojmů z oblasti objektově orientovaného paradigmatu. Je možné náš výklad dále upřesňovat a doplnit celou řadou příkladů dokonale osvětlujících náš myšlenkový vzor. Doufám, že budu mít příležitost tak učinit například v rámci dalších ročníků tohoto semináře.

Domnívám se, že kvalifikované uvažování o základních principech pojmů se kterými zacházíme, je důležité pro další rozvoj oboru, ve kterém tyto pojmy užíváme a je také důležité s ohledem na jejich výuku a praxi. Není pochopitelně jediná definitivní odpověď, které paradigma, který výklad je ten správný. Jsou však výklady správnější nežli jiné a já doufám, že ten mnou zde prezentovaný se k nim připočítává.

Literatura

[1] Molhanec, Martin. „Několik poznámek k porozumění objektového paradigmatu“. Sborník konference *Objekty 2004*, Praha. ČZU PEF 2004, s. 189-197. ISBN 80-248-0672-X.

[2] Molhanec, Martin. „Kritika některých výkladů objektově orientovaného paradigmatu“. Sborník konference *Tvorba software 2004*. Ostrava. Tanger, 2004, s. 163-172. ISBN 80-85988-96-8.

[3] Molhanec, Martin. „Objektové metodologie – jejich užití a výklad“. Sborník konference *Tvorba software 2003*. Ostrava. Tanger, 2003, s. 111-115. ISBN 80-85988-83-6.
On line: http://martin.feld.cvut.cz/~molhanec/VaV/files/publik/2003/OOkrit-co.pdf

[4] Molhanec, Martin. „UML – několik kritických poznámek“. Sborník konference *Tvorba software 2002*. Ostrava. Tanger, 2002, s. 150-159. ISBN 80-85988-74-7.
On line: http://martin.feld.cvut.cz/~molhanec/VaV/files/publik/2002/UML.pdf

[5] Booch, G., Jacobson, I., Rumbaugh, J.: *The Unified Modeling Language Reference Manual*. ADDISON-WESLEY, 1999. ISBN 0-201-30998-X

[6] Booch, G., Jacobson, I., Rumbaugh, J.: *The Unified Modeling Language User Guide*. ADDISON-WESLEY, 1999. ISBN 0-201-57168-4

[7] Joseph Schmuller: *Myslíme v jazyku UML*. GRADA Publishing, 2001. ISBN 80-247-0029-8.

[8] Meilir Page-Jones: *Základy objektově orientovaného návrhu v UML*. GRADA Publishing, 2001. ISBN 80-247-0210-X.

[9] Jim Arlow, Ila Neustadt: *UML a unifikovaný proces vývoje aplikací*. Computer Press, Brno 2003. ISBN 80-7226-947-X.

[10] Hana Kanisová, Miroslav Müller: *UML srozumitelně*. Computer Press, Brno 2004. ISBN 80-251-0231-9.

[11] Merunka, Vojtěch. „Normalizace v objektových databázích“. Sborník konference *Objekty 2004*, Praha. ČZU PEF 2004, s. 160-173. ISBN 80-248-0672-X.

[12] Coad, P., Yourdon, E.: *Object-Oriented Analysis*. YOURDON PRESS, 1991. ISBN 0-13-629981-4

[13] Grady Booch: *Object oriented design with applications*. The Benjamin/Cummings Publishing Company, 1991.

[14] James Rumbaugh, et al.: *Object-oriented modeling and design*. Prentice-Hall, 1991. ISBN 0-13-629842-9.

[15] I. Jacobson, M. Christerson, et. al.: *Object-Oriented Software Engineering: A Use Case Driven Approach*. Addison-Wesley, 1992. ISBN 0-201-54435-0.

[16] Chen, P. "The Entity Relationship Model Toward a Unified View of Data". *ACM Transaction on Database Systems*. March 1976.

[17] Matt Flavin: *Fundamental Concepts of Information Modeling*. Prentice-Hall, 1979.

[18] Sally Shlaer, Steve Mellor: *Object-Oriented Systems Analysis*. Prentice-Hall, 1988.

V Praze, 28. března 2005
Martin Molhanec