Proudy v objektově orientovaných metodikách

Pavel Drbal

Vysoká škola ekonomická, Katedra informačních technologií, nám. W. Churchilla 4, 130 00 Praha 3, Česká republika

Abstrakt
Článek shrnuje jednotlivé objektově orientované metodiky v jednotnou metodiku a definuje jednotlivé proudy použitých prostředků jako důsledek typu úlohy.

Úvod

Rozvoj objektově orientovaných metodik dosáhl již určitého stupně, kdy je možné vidět určitou ustálenost. Lze označit určité proudy jednak v metodikách samotných, jednak v jejich dalším vývoji. Je možnost rozeznat následující trendy:

- ustálení v prostředcích,
- rozlišení typu metodiky podle typu úlohy,
- další rozvoj.

Tyto trendy jsou zakryty velkým množstvím publikací o tomto předmětu. Každý autor má ve své knize svůj osobní přínos k metodice, a ten zdůrazňuje, aby se čtenáři vryl do vědomí. Navíc je každý autor omezen typem úloh, ve kterých je angažován. Tento text se pokouší podat objektovou metodologii jako arsenál prostředků a technik vhodných pro tvorbu velkých (informačních) systémů.

Objektové metodiky nevznikly na zelené louce, jsou logickým dovršením vývoje předchozích (strukturálních) metodik. Přebírají vše, co se osvědčilo, přidávají něco svého, část osvědčených přístupů pouze zabalí do nového kabátku. Takovým typicky děděným přístupem je současné (paralelní) zpracování několika modelů, které používají určitý vlastní (zobecněný, tj. omezený) pohled na celý systém.

Nemyslím si, že by bylo možno nezaujatě rozhodnout, zda je lepší objektový nebo strukturální přístup. Samozřejmě velmi záleží na volbě kritéria, podle kterého rozhodujeme co je lepší. Pokud za toto kritérium zvolíme snadnost úpravy již hotového systému, pak většina lidí, kteří mají zkušenosti s používáním obou postupů, se kloní k objektovému přístupu. Lze však konstruovat situace, kdy je zřetelně objektový přístup nevýhodný.

Ustálení v prostředcích

Za vcelku ustálené rysy považuji objektovost (objektovou orientaci), etapisaci postupů, iteraktivnost postupů a jednotlivé modely. Stručně to charakterizujeme.

Objektovost

Za nezaměnitelnou výhodu objektového přístupu považuji přísné oddělení části programu (vnitřku objektu od ostatního programu) a nutnost jednoznačně definovat komunikační protokol mezi objektem a jeho okolím. Tato vlastnost se projevuje ve větší průhlednosti systému a v omezení dosahu změn, pokud je v již hotovém systému provádím.

Vlastnosti objektů (třída jako zobecnění objektů, zapouzdření, dědění jako generalizace a specializace, asociace jako zobecnění linků, polymorfismus) dále stručně charakterizujeme z hlediska návrhu systémů, považuji však za vhodné zdůraznit vlastnost **synergie**, tj. že objekt jako jednotka skládající se z dat a funkcí má další žádané vlastnosti, které volná (rozptýlená) skupina dat a funkcí nemá.

- **Třída** je identifikovantelný typ entity nebo konceptu v daném prostředí. Třída je skupina datových proměnných a chování (operace, funkce), což je sdíleno instancemi toho typu. Instance třídy je objekt v užším smyslu slova.
- **Objekt** má svou vlastní identitu již tím, že existuje. Je schopný reagovat bez ohledu na to, jakým způsobem k němu přistupujeme.
- **Zapouzdření** je výrazná vlastnost, která umožňuje oddělit definovaným způsobem objekt od jeho okolí.
- Objektových hierarchií nad objekty je více druhů, nejpopulárnější je **dědění**, které můžeme chápat buď jako generalizaci (abstrakci) nebo jako specializaci (redukci).
- **Zpráva** je požadavek, který jeden objekt posílá druhému. Zpráva vyžaduje, aby přijímací objekt provedl určitou operaci nebo aby poskytl určitá data.
- **Link** je spojení mezi objekty (cesta) po kterém může být předávána zpráva.
- **Asociace** je abstrakce linků, tj. je to spojení mezi třídami, které může (ale nemusí) být realizováno linkem.
- **Polymorfismus** je princip, že různé objekty reagují na tutéž zprávu různě, tj. svým vlastním způsobem.

V objektovém přístupu je velmi důležitý koncept „třída". Podstatou této důležitosti je to, že při návrhu nepracujeme s objekty, které posléze tvoří informační systém, ale s jejich generalizacemi, se třídami. Tento princip generalizace se při návrhu používá na mnoha místech.

Etapisace

Myslím, že rozdělení projekce do čtyř etap (místo původní analýzy a designu) odpovídá potřebám, viz obrázek na další straně. Každá z etap má své hlavní téma, všechna jsou důležitá. Nelze některému dávat přednost, kvalita celého řešení závisí na kvalitě jednotlivých kroků. Nelze v dalším kroku napravit chybný přístup předchozího kroku, každá etapa má svůj úkol.

Rozbor úlohy (model požadavků)

Seznam požadavků na vytvářený systém, seznam omezení a podmínek, kterým musí vytvářený systém vyhovovat. Cílem této etapy je porozumět úloze, vytvořit si jednotnou terminologii a vytvořit prostředek, jak se domluvit se zadavatelem a uživateli. Zde se nejvíce projevuje závislost na typu úlohy a záměrech zadavatele.

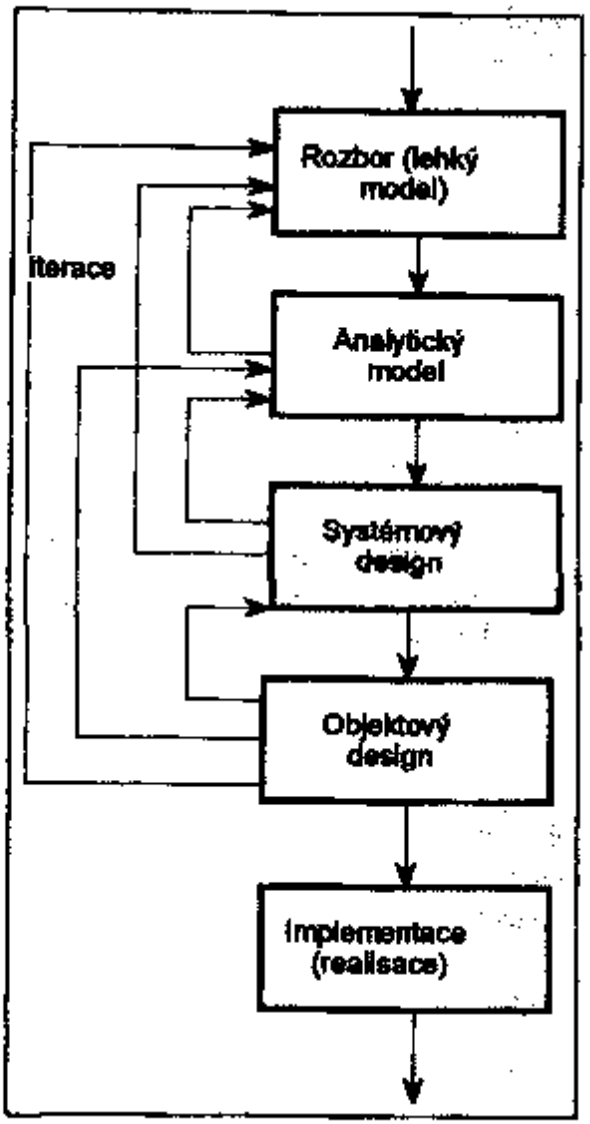
Vlastní analýza

Cílem je vytvořit analytický (logický) model, tj. popis řešení nezávislý na prostředí. Prostředkem k dosažení tohoto cíle je tvorba pohledových (dílčích) modelů (modelů spolupráce, objektového, dynamického, funkčního ap.). Analýza projektu není řešení projektu, zde „pouze" hledáme logické vazby, které musí být nutně splněny, zde definujeme hrubý algoritmus celého řešení. Cílem analýzy je ideální řešení. Výsledkem analýzy je jednak logický model, který tvoří základ dalších kroků, jednak naše porozumění složitosti úlohy a jejím požadavkům. Obojího využijeme v další etapě.

Systémový design

Souhrnně se dá říci, že se zde řešení strategická rozhodnutí pro implementaci.

V této etapě se dává do souladu logický model a možné implementační prostředí. (Pod pojmem implementační prostředí rozumím jak prostředí, ve kterém bude realizovaný systém existovat, tak vývojové prostředí, které je použito k realizaci modelu.) Na jedné straně se podle typu a vlastností modelu vybírá vývojové prostředí, na druhé straně se model upravuje tak, aby odpovídal implementačnímu prostředí. Sem patří volba architektury, způsobu řízení, implementačních prostředků ap.



Druhá věc, kterou zde zajišťujeme je organizace vlastní práce. známe typ úlohy (z lehkého modelu), známe hlavní charakteristiky řešení (z analytického modelu), známe zvolenou architekturu. Jedná se jednak o různé konvence (pojmenování, sdílení prostředků), jednak o organizační záležitosti (přidělení práce skupinám), jednak o nasměrování implementačního úsilí (zda šetřit čas či paměť).

Objektový design

Tvorba předpisu pro implementaci. Je to nejdůležitější etapa pro vlastní programování. Předchozí etapy spíše zadávaly algoritmy, tj. zadávaly jejich vstupy a výstupy než že by přesně popisovaly algoritmy. Skutečný návrh algoritmů je vytvářen až teď.

Navrhují se realizace vazeb, tj. rozhoduje se, jakým způsobem se realizují jednotlivé agregace a asociace. Likvidují se asociace typu M:N, přesněji řečeno, takové asociace se nahradí speciálním typem objektů (to je společné se strukturovaným návrhem). Rozhoduje se, zdali se asociace realizuje přímo (tj. ukazateli) nebo pomocí nějakého seznamu (jeho prohledáváním).

Zvolím nástroje. Rozumí se tím aparát pro rozhraní s uživatelem (okna), třídy typu kolekce nebo kontejner, zajištění persistence objektů, reprezentace grafických výstupů, styk s uzly počítačové sítě. Základní dilema zde je: buď samostatná tvorba (je to pracnější) nebo nákup nástroje (nástroji musím přizpůsobit svůj model řešení úlohy).

Tato a další rozhodnutí zásadně ovlivňují efektivnost vytvářeného systému a pracnost jeho vytváření.

Iterativnost

Důležitou součástí všech metodik je iterativnost. Znamená to možnost vrátiti se z kterékoliv etapy do některé předchozí a opakovat postup. Při zjištění nějakých potíží nebo nesrovnanosti je zapotřebí najít jejich zdroj, opravit původ potíží a zopakovat postup. Při návrhu systémů nelze záplatovat, obejít současnou potíž a rychle pokračovat dále. Návrh nelze lepit z kousků jako vlaštovka hnízdo. Důležité jsou ekonomické argumenty. Provést znovu část analýzy je lacinější než vyhodit několikatýdenní práci programátorů.

Iterativnost je dobrý prostředek k pochopení problému (když nemáme lepší). Jestliže je nějaká oblast definována vágně, pak uděláme prvý návrh. Nějakou dobu tento návrh rozpracováváme, a když jsme pochopili jeho podstatu, vrátíme se s novými vědomostmi znovu a původní návrh upravíme. Nebo se na problém podíváme z jiného úhlu, a po rozpracování a pochopení nových aspektů původní návrh upravujeme.

Příkladem takového iterativního pochopení problému je volba tříd pro objektový model. Máme před sebou zadání a naší úlohou je vytvořit objektový model. Zhruba řečeno, naším prvým krokem je zvolit třídy, druhým krokem je určit, jak budou spolu komunikovat. Zvolit třídy je problém. Kulantně řečeno, definice tříd vyplývá z porozumění problému. Jestliže je mi daná oblast blízká, tak to nemusí být problém. Jestliže ale nevěřím genialitě své intuice, musím použít formálnější postup. Podstatou tohoto formálního postupu je iterace.

Příklad:

1. krok. Za kandidáty na třídy zvolím podstatná jména ze zadání.
2. krok. Roli a význam každého kandidáta popíši na několika řádcích.
3. krok. Vrátím se k seznamu kandidátů a prověřuji jej (oproti prvému kroku mám navíc jejich popis a určité pochopení jejich role). Slučuji kandidáty se stejným nebo podobným popisem, vyhazuji ty, kteří se ukazují být pouze atributy některého jiného, volím vhodnější jména.
4. krok. Redukovaný seznam kandidátů na třídy považuji za prvou verzi seznamu tříd a dělám prvý model. (Tj. určuji smysluplné vztahy mezi těmito třídami - určuji asociace.)
5. krok. Když model dosáhne určité logické uzavřenosti, přeruším tuto práci a vrátím se k seznamu tříd. Prověřuji, zdali dvě třídy, které mají stejné vazby, nejsou vlastně totožné, zdali třída s malými vazbami na ostatní není zbytečná. Trochu nadneseně by se dalo říci, se snažím dosáhnout jakési elegance řešení a že odstraňuji vše, co této eleganci vadí.

Návrh systémů je vlastně rytmické střídání tří činností: zpodrobnění, ověření a porovnání.

Zpodrobnění. Další rozvoj modelu systému. Přidávání nových vztahů, jejich popis a výstižné pojmenování. Tento rozvoj modelu znamená hlubší porozumění vytvářenému systému, v podstatě to je přechod z obecnější úrovně na konkrétnější.

Ověření. Po zpodrobnění následuje ověření abstraktnějších úrovní, jestliže nebyl pozměněn smysl pojmů a prvků. Pokud k takovému pozměnění došlo, je třeba nejít příčiny a odstranit je.

Porovnání. Po ukončení některé ucelenější etypy se tento aktuální pohled (model) porovnává s jinými. Každý model zdůrazňuje jiný pohled na vytvářený systém, přesto mají některé prvky stejné nebo podobné prvkům jiného modelu. Je zapotřebí porovnat, zdali změna v navrhovaném systému provedená v jednom modelu nezpůsobí změnu v jiném modelu. Pokud k tomu došlo, ověřujeme tuto změnu v tomto modelu.

Střídání pohledů na vytvářený systém a výše uvedený rytmus práce zaručuje určitý stupeň regulérnosti tvorby systému.

Modely

Objektově orientovaná metodologie se používá pro návrh složitých systémů, takových, které nelze pojmout jako jeden celek. Ke zvládnutí tohoto problému se používá technika modelů. Jeden model je pohled na systém, který zdůrazňuje jeden nebo dva aspekty a potlačuje aspekty jiné.

Minimálně jsou nutné dva modely, statický a dynamický. Statický model popisuje vytvářený systém jako síť objektů, které jsou propojeny cestami, po kterých může probíhat komunikace. Dynamický model je složitější, je nutno popsat dvě relativně samostatné složky: meziobjektovou interakci a chování jednotlivých objektů.

Pokud se omezíme na tyto dva základní modely, tak snižujeme možnost sebekontroly porovnáváním různých modelů mezi sebou. Navíc toto rigidní omezení na „čistě objektový přístup" omezuje rozsah úloh, které je možno pohodlně řešit.

Slovo „model" se v této metodologii běžně používá v několika významech, které jsou zřejmé z přívlastků nebo kontextu. „Model" bez přívlastku znamená model vytvářeného systémů. Tento se skládá až ze tří základních modelů s přívlastky, a to z „objektového modelu", „dynamického modelu" a z „funkčního modelu". Některé rozsáhlejší části se také označují slovem model (s přívlastkem).

Není zde prostor k podrobnějšímu popisu jednotlivých modelů, to je předmětem připravované monografie. Podám zde pouze povšechné charakteristiky.

Objektový model

Objektový model je nejdůležitější, protože se do něj promítá většina poznatků získaných v jiných modelech. Skládá se z tříd a jejich vztahů, musíme si uvědomit důležitou věc: ve vytvořeném systému budou pracovat objekty, my ale navrhujeme jejich zobecnění -

třídy. Objektový model je statickým pohledem na systém, popisuje obsažené prvky a cesty pro předávání zpráv.

Funkční model

Funkční model je totožný s modelem datových toků strukturálního přístupu. Není nezbytný, objektový a dynamický model úplně popisují celý systém. Myslím si, že ze dvou důvodů je funkční model vhodný pro objektový přístup:

- Poskytuje sebekontrolu. Nezávislou tvorbou funkčního modelu můžeme ověřit správnost návrhu objektového modelu. Datové sklady odpovídají objektům nebo jejich částem, proces je několik operací.
- Jsou úlohy, při jejichž řešení je funkční model důležitější než objektový.

Funkční model poskytuje procesní pohled na systém, je to hierarchický rozklad úlohy vzhledem k procesům.

Model jednání

Model jednání popisuje interakce vytvářeného systému s okolím. Podstatou modelu jednání jsou slovní scénáře styku mezi uživatelem a systémem.

Model jednání má v procesu projekce dvě hlavní úlohy:

- Poskytuje přesné zadání, které se používá po celou dobu tvorby navrhovaného systému.
- Slovní scénáře jsou východiskem pro tvorbu dynamického modelu.

Dynamický model

Tento model je velmi složitý a je nutno jej rozložit na čtyři části, přesněji řečeno na čtyři kroky. Při popisu dynamického modelu se nevyhneme popisu postupu. Jde o tyto části:

1. slovní scénáře,
2. grafické scénáře, popisující interakci mezi objekty,
3. celosystémový diagram událostí,
4. přechodové diagramy jednotlivých objektů.

Dynamický model popisuje chování navrhovaného systému. Z hlediska tvorby to znamená, že popisuje předávání řízení mezi prvky systému. V objektovém prostředí to popisujeme tak, že objekty si navzájem předávají události.

Slovní scénáře se přebírají z modelu jednání. Slovní scénář je posloupnost vět (frází), které popisují požadavek aktora, odpověď systému, nový požadavek aktora, reakci systému a tak dále. Návrh obrazovky je dobrá pomůcka pro tvorbu slovních scénářů.

Ze slovních scénářů se odvodí **grafický scénář**, který popisuje interakce mezi objekty jednotlivých tříd.

Grafický scénář je prostředek pro zachycení chování systému a jeho vnitřní dynamiky. Je to prostředek, pomocí kterého můžeme odvodit vnitřní dynamiku systému z požadavků na jeho vnější chování.

Diagram událostí. Poté, co uznáme meziobjektovou dynamiku za dostatečně popsánu, provedeme inventuru všech událostí. Cílem je evidovat všechny události, které se týkají jedné třídy.

Chování objektu (třídy) popíšeme pomocí přechodového diagramu. Přechodový diagram je síť stavů a přechodů mezi stavy.

Model spolupráce

Model spolupráce slouží k odvození objektového modelu z modelu jednání. Vytváří se ve třech fázích:

- tvorba seznamu tříd a seznamu úkolů (odpovědností),
- přiřazení odpovědností třídám,
- tvorba vlastního modelu spolupráce.

Model spolupráce výhradně slouží ke tvorbě objektového modelu. Model spolupráce reprezentuje dílčí pohled na systém, objektový model reprezentuje celkový pohled. I střídání těchto pohledů můžeme použít ke kontrole své práce.

Rozlišení typu metodiky podle typu úlohy

Volba metodiky samozřejmě závisí na mentalitě, zkušenostech a znalostech řešícího týmu. Když slyším odsouzení některé metodiky, tak jsou pro to dva hlavní důvody:

1. Mluvčí ovládá metodiku pro omezený okruh úloh a nevhodně ji použil.
2. Mluvčí příslušnou metodiku nezná (nebo nepochopil) a odsuzuje ji proto, aby odůvodnil svou neznalost.

Nevidím důvod, aby dobře pracující tým měnil svůj způsob práce ze strukturálního na objektový z viceméně módních důvodů. Na druhé straně, po svých zkušenostech bych se již nikdy nevracel zpět k neobjektovému přístupu. Osobně si na objektovém přístupu (v projekci i programování) nejvíce cením toho, že použití objektů nutí k pořádku ve věcech a že mám možnost si pomocí objektů vytvořit speciální jazyk pro řešení problémů.

Hlavní pozornosti v objektových metodikách se ani tak netěší samotný vznik nových systémů, jako návrh nových systémů tak, aby byly snadno upravovatelné. **Upravovatelnost systému je dána jeho projekcí**. Pokud se z tohoto hlediska udělá chyba, pak ji již nejde odstranit. Upravovatelnost systémů je hlavní motiv všech objektových metodik, upravitelnost je hlavní důvod volby objektové metodiky.

Objektových metodik je na duševním trhu (v literatuře) dostatek, ze všech jsem ale cítil určitou jednostrannost. Ať již v přílišném soustředění na pouhé objekty, nebo v celkovém zaměření metodiky. Teprve Rumbaughova metodika OMT má zřetelný rys univerzálnosti. myslím si, že je zapotřebí dívat se na objektově orientovanou tvorbu nových systémů jako na **metodologii**, tj. že to je souhrn mnoha metodik se společnými rysy.

Je zřejmé, že různé metodiky vyhoví různě při řešení jednotlivých úloh. Při zkoumání závislostí mezi typem úlohy a typem objektově orientované metodiky je vidět, že pro členění úloh jsou důležitá tato kritéria:

- míra účasti člověka na práci systému,

- vztah mezi objektovým modelem systému a mezi realitou.

Míra účasti člověka na práci systému

Podle míry účasti člověka na práci systému lze rozeznat systémy s **otevřeným cyklem** řízení, kde převážně rozhoduje člověk a systém připravuje podklady pro rozhodování. Opačným extrémem jsou systémy s **uzavřeným cyklem řízení**, kdy člověk pouze inicializuje práci systému, systém pracuje dále sám a rozhoduje se podle vestavěné inteligence. Samozřejmě existuje celá škála mezistupňů.

Dá se říci, že u systémů s otevřeným cyklem je důležitý objektový a dynamický model systému, protože musíme klást důraz na interakci mezi člověkem a systémem. Při tvorbě systémů s uzavřeným cyklem nelze příliš popisovat chování, základním modelem je buď objektový model nebo funkční model (záleží na druhém kritérii, totiž na tom, zdali existuje vzor v reálném světě). Jestli je zadání algoritmické, tj. zadání určuje posloupnost vnitřních dějů, pak se zadání nejlépe popíše funkčním modelem.

Model systému a realita

Informační systém vždy nějakým způsobem odpovídá realitě. Uvědomme si však, že tím, že nějaký informační systém vznikl, tím již byla realita určitým způsobem změněna. Ta změna může být různého stupně, počínaje minimální změnou (do reality byl přidán informační systém) až po zásadní změny (organizace podniku byla zásadně přizpůsobena informačnímu systému). Vezmeme-li v úvahu realitu v době zadání a realitu v době dohotovení modelu, můžeme vytvořit celou škálu úloh:

i. modelování existující reality,
ii. informační obsluha existující reality,
iii. modifikace existující reality,
iv. přizpůsobení reality požadavkům,
v. nová realita jako důsledek informačního systému
vi. investigativní tvorba.

V zásadě můžeme vymezit tři základní typy:

- Východiskem je **existující realita**. Úlohy typu (i) a (ii) jsou spíše technického charakteru, například monitorování počasí nebo trenažér pro piloty. V podnikové sféře se informační systém vždy promítne do změny reality. (Zkušenost říká, že pokud se nepromítne, tak je špatně navržen). Tento typ úloh se od následujícího typu liší mírou změn, dá se také říci, že se liší úmyslem provést nebo neprovést zásadní změny v organizaci.
- Východiskem jsou funkční požadavky a cílem je **nová realita** (většinou se novou realitou rozumí nová organizace podniku). V úlohách tohoto typu se úmyslně ignoruje existující realita, ale je jasně zadána funkce nové reality.
- Pod pojmem „investigativní tvorba" rozumím to, že souběžně vzniká jak informační systém, tak i nová realita, aniž je chování a vlastnosti nové reality zřejmé. Je znám **pouze** cíl činnosti. I v tomto případě se většinou jedná o řešení technických problémů, protože si stěží dovedu představit, že někdo dělá informační systém pro podnik aniž ví, čím se tento podnik bude zabývat.

Popis základních postupů

Lze vymezit několik základních postupů, jejichž výběr závisí na kritériích popsaných v předchozí kapitole. Jednotlivé postupy se odlišují v prvých fázích analýzy, tj. ve tvorbě logického modlu. Další postup práce pak již závisí na charakteru tohoto logického modelu.

Nosným modelem objektově orientovaného přístupu je téměř vždy objektový model. Jednotlivé postupy se liší tím, jakou cestou se k tomuto modelu dostaneme. Když si probereme jednotlivé typy modelů, vidíme, že jsou možné čtyři počáteční přístupy k řešení úlohy. Jsou to:

- Funkční model ve spolupráci s objektovým modelem.
- Objektový model ve spolupráci s funkčním modelem.
- Objektový model ve spolupráci s dynamickým modelem.
- Model jednání (tj. funkční model) ve spolupráci s objektovým modelem.

Výše uvedené typy přístupů jen vyjmenovávají zdůrazněné modely, samozřejmě nevylučují použití i dalších modelů. Nutno přiznat, že právě uvedené čtyřčlenné dělení je poněkud akademické. Je pravda, že nakonec se vždy ukáže, že jsem použil jednoho z výše uvedených postupů, ale nemusím se na začátku práce rozhodovat, který z nich použiji. Jediné, co musím na začátku rozhodnout, je zdali použiji výlučně pouze model jednání (a z něj budu odvozovat objektový model), nebo jestli budu vytvářet objektový model nezávisle na modelu jednání.

Shrnutí

Celkově lze počáteční rozhodování popsat schematickým algoritmem s následujícími body:

1. Nejprve vytvořím model jednání. Základní úlohou modelu jednání je dobře porozumět úloze, získat prostředek pro komunikaci se zadavateli a pro kontrolu dalších fází práce, nakonec také rozlišit akce uživatele a systému. Mohou nastat tři případy:
   - Model jednání nelze rozumně vytvořit. Týká se to většinou systémů s uzavřeným řídicím cyklem. Je nutno upřesnit úlohu jinými prostředky (například kontextovým diagramem) a přejít na jiný postup, viz bod 3.
   - Model jednání je jediným zdrojem pro tvorbu systému, tj. nechci (nebo nelze) vycházet z reality. To je postup popsaný Jacobsonem, viz bod 2.
   - Úloha modelu jednání se v této fázi omezí na upřesnění zadání a ve tvorbě modelu vytvářeného systému se bude pokračovat modelováním Reality", viz bod 3.
2. Z modelu jednání se odvodí model spolupráce, z modelu spolupráce se odvodí objektový model. Dále viz bod 4.
3. Vytvoří se objektový model a k němu se paralelně rozvíjí funkční model. Charakter úlohy se projeví v tom, zdali se klade důraz na manipulaci s daty nebo na dynamické chování systému.
   - Při složitém algoritmu datové manipulace je hlavním modelem funkční model. Projeví se to tak, že se operace objektů odvozují hlavně z procesů funkčního modelu.
   - Při požadavku složitého dynamického chování se z paralelního porovnávání funkčního a objektového modelu přejde na paralelní rozvoj objektového a dynamického modelu. Viz bod 4.
4. Východiskem tohoto kroku je to, že objektový model popisuje statickou stránku vytvářeného systému a že model jednání formuluje požadavky na chování vytvářeného systému. (Doporučuji se v běžných úlohách věnovat nejdříve statické stránce a teprve

pak dynamické.) Podstatou činnosti je tvorba dynamického modelu a z toho vyplývající úpravy objektového modelu. Tvorba dynamického modelu je popsána v následujících bodech 5 až 9.

5. Volba rozhraní mezi uživatelem a systémem.
6. Upřesnění scénářů styku uživatele se systémem.
7. Definice událostí a mezi objektového styku, tj. určení chování systému na úrovni objektů.
8. Evidence souvislostí mezi událostmi a objekty.
9. Určení chování jednotlivých objektů, což je již vlastně předpis řídicích operací, tj. těch, které určují chování objektů.

Vidíme, že základní postup končí vytvořením logického modelu buď bodem 3 (v případě algoritmické složitosti) nebo bodem 9 (v případě dynamické složitosti).

V případě algoritmické složitosti je obtížnost řešení dána složitostí vytvářeného modelu systému. Tato složitost se řeší klasickými prostředky, tj. členěním modelu na podsystémy, víceúrovňovým návrhem apod.

V případě dynamické složitosti se obtížnost projeví složitým postupem. Zde se projeví výhody objektového přístup, body 5 až 9 rozčlení zpracování dynamické složitosti do posloupnosti relativně jednoduchých kroků.

Další rozvoj

Uvnitř metodik se dá očekávat rozvoj prostředků pro určité speciální případy, který koliduje a bude kolidovat s obecností metodik. Vně samotných metodik lze očekávat rozvoj samostatné metodiky opakovatelně používaných prostředků (nástrojů) a rozvoj „nadobjektové" metodiky vzorců (patterns).

Literatura

1. Booch,G.: Object-Oriented Analysis and Desing, The Benjamin/Cummings, Redwood City 1994
2. Coad,P.-Yourdon,E.: Object-Oriented Analysis, Prentice-Hall 1991
3. Coad,P.-Yourdon,E.: Object-Oriented Design, Prentice-Hall 1991
4. Frost,S.: The Select Perspective, Select Software Tools, Cheltenham 1994
5. Jacobson,I.: Object-Oriented Software Engineering, Addison-Wesley 1994
6. Mellor,S.J.-Shlaer,S.: Object-Oriented Systems Analysis, Prentice-Hall 1988
7. Rumbaugh,J.-Blaha,M. etc.: Object-Oriented Modeling and Design, Prentice Hall, ISBN 0-13-630054-5
8. Šešera,Ľ.-Mičovský,A.: Objektovo-orientovaná tvorba systémov a jazyk C++, PERFEKT a.s. Bratislava 1994, ISBN 80-85261-66-9