METODIKA KLASIFIKACE METODIK

Pavel Drbal

VŠE nám. W. Churchilla 4, 130 00 Praha3, <drbal@vse.cz>, http://NB.vse.cz/~drbal

Abstract
This paper explains the systematic view on object methodologies, this view is suitable both for classifying and creating methodologies.

Anotace
Článek vysvětluje systematický pohled na objektové metodiky, který je vhodný k jejich klasifikaci i k jejich tvorbě.

1. Úvod

V článku je rozpracována idea Pavla Hrubého [1]. Pro klasifikaci metodik je vhodné použít to, čemu se obvykle říká systémový přístup. Chceme-li porovnat dva systémy, pak to provedeme v následujících krocích:

- Systém rozložíme na množinu prvků.
- Určíme vztahy mezi prvky.
- Mezi dvěma systémy porovnáme množiny prvků.
- Porovnáme vztahy mezi adekvátními prvky.

Velmi důležitá jsou kritéria, kterými identifikujeme prvky a vztahy ve srovnávaných systémech. Teprve nalezení společných kritérií umožňuje porovnání systémů. *(Můžeme porovnávat jablka s hruškami, jestliže si jako kritéria určíme hmotnost, barvu, chemické složení.)* Porovnání prvků a vztahů jsou dvě různé věci, můžeme mít dva různé systémy, které mají tytéž prvky a různé vztahy mezi těmito prvky (například oddělení firmy při práci a večírek oddělení).

V dalším se omezíme na systémy objektově orientované projekce, mezi jiným předpokládáme, že čtenář zná pojmy jako je třída, instance, asociace, typ jednání (Use Case).

2. Kritéria

Všichni se jistě shodneme na tom, že základními kritérii jsou (1) granulita a (2) pohled. Pojem „granulita“ (úroveň podrobností nebo úroveň rozkladu) je zřejmý. Pojem „pohled“ je více specializovaný – v podstatě se jedná o to, že při tvorbě modelu systému jedny aspekty zdůrazňujeme, jiné zanedbáváme (abstrahujeme od nich).

Za hlavní přínos [1] považuji rozdělení kritéria pohledů na dvě kritéria, která prozatím pojmenujeme (2a) pohled dle vnitřních aspektů a (2b) dle účelových aspektů.

Za základní prvky systému „**objektové projekce**“ považujeme „**modely**“. Můžeme se tedy dívat na objektovou projekci jako na posloupnost modelů, mezi kterými jsou určité vztahy (například

následnosti). Nadále slovem „model“ budeme rozumět model vytvářeného systému, buď celého nebo jeho části.

V tomto textu se nevyhneme používání slova „systém“ ve dvou významech – jako projekcí vytvářený systém (konstruovaný systém) a jako projekční systém – tj. jak vytváříme vytvářený systém.

Z výše uvedených úvah plyne, že pro identifikaci prvků projekčního systému máme k disposici tři kritéria (níže je vysvětlíme):
A) granulita,
B) členění modelů dle vnitřních aspektů (modely dle pohledu na vnitřní aspekty),
C) pohledy z hlediska užití (rozumí se užití modelů v projekci).

2.1 Granulita

Add A) Granulita je úroveň podrobností. Například můžeme modely dělit takto:
1. úroveň systému jako celku
2. úroveň podsystémů
3. úroveň tříd
4. úroveň procedur
5. úroveň programového kódu

2.2 Vnitřní aspekty

Add B) V [1] se zavádí tyto čtyři modely:
B1. Popis prvků modelu.
B2. Statický (vztahový) model
B3. Dynamický (interakční) model..
B4. Životní cyklus prvků modelu.

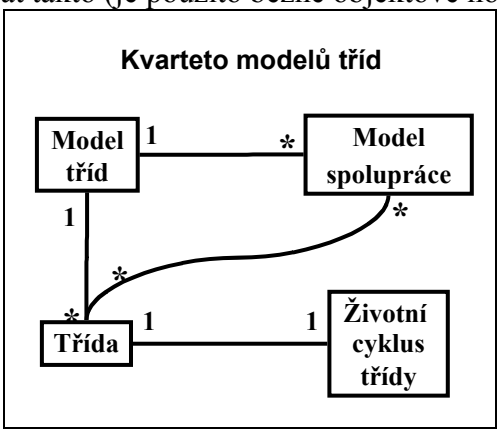
Jako typický příklad poslouží **objektový model**. Prvky modelu jsou třídy – výše zmíněné kvarteto je toto:

Ba1. **Popis tříd**, tj. pro každou třídu výčet a charakteristika metod, atributů podmínek a omezení.
Ba2. **Objektový model** (také se říká model tříd). Je to síť, jejíž uzly jsou třídy a jejíž hrany jsou vztahy asociace, agregace, generalizace (dědění).
Ba3. Dynamické modely mohou mít více representantů, nejčastější jsou **sekvenční model** (model objektových interakcí) a **model spolupráce**. Obě representace jsou síť, jejíž uzly jsou objekty, vztahy jsou předávání zprávy s určeným pořadím.

Uvědomte si podstatný rozdíl mezi modelem tříd (uzly jsou třídy) a dynamickými modely (uzly jsou instance tříd – objekty).

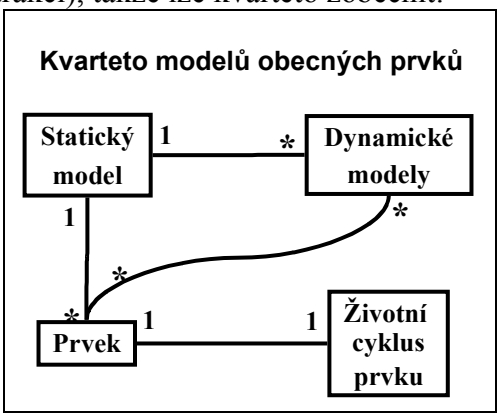
Ba4. Životní cykly jsou obvykle realizovány **stavovými diagramy** jednotlivých tříd. Životní cykly mohou mít rozličné representace, kromě stavových diagramů také přechodové tabulky, grafy aktivit; ostatně je lze zapsat i v Backus-Naurově formě.

Vztahy v kvartetu tříd lze zapsat takto (je použito běžné objektové notace):



Obrázek 1: Kvarteto modelů tříd

V objektových metodikách má většina používaných prvků charakter tříd (rozlišují se abstrakce a instance, je několik stupňů abstrakcí), takže lze kvarteto zobecnit:



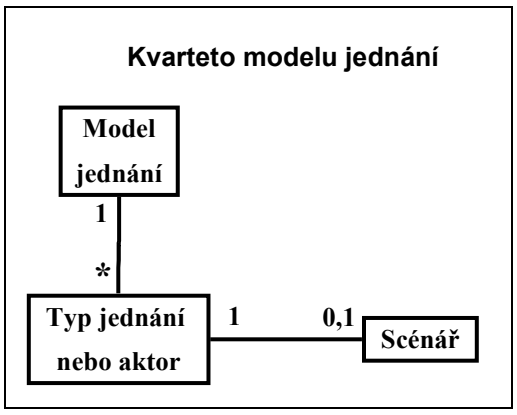
Obrázek 2: Kvarteto modelů obecných prvků

Pro jednotlivé části kvarteta je zaveden pojem artefaktu (jeho definice je níže).

Výše zavedené kvarteto je abstrakce používaných postupů a slouží ke klasifikaci existujících užitečných postupů, nikoliv jako svěrací kazajka, které se skutečnost musí přizpůsobit. Ukážeme si to na kvartetu typu jednání (Use Case).

Bb1. Slovní **popis aktorů a typů jednání**.
Bb2. **Model jednání**.
Bb3. Dynamický model se nepoužívá.
Bb4. Životní cyklus typu jednání se popisuje **scénářem**.

Z podstaty aplikace těchto modelů vyplývá, že neexistuje dynamický model jednání a že životní cykly aktorů jsou mimo zájmovou oblast projektanta, i když jsou v zásadě možné. Příslušné kvarteto lze zapsat takto:



Obrázek 3: Kvarteto modelu jednání

2.3 Pohledy užití

Add C) Rozumí se pohled z hlediska užití projektanta. V okruhu metodik kolem UML to jsou tyto pohledy:

- analytický,
- jednání (Use Case),
- logický,
- komponent,
- rozložení (deployment)

Nepřipadá mi snadné charakterizovat tuto skupinu definicí jednoho rysu. Za nejsnáze uchopitelnou definici této skupiny považuji tuto konstruktivní:

Ze všech možných pohledů na moduly celého vytvářeného systému vydělíme granulitu (add A), statický a dynamický pohled (add B) a všechny ostatní pohledy zařadíme add C. *(Texty popisu prvku a jeho životní cyklus nejsou pohledy na celý vytvářený systém.)*

Název „pohled z hlediska užití“ jsem zvolil proto, že tyto pohledy dosti charakterizují jednotlivé etapy projekce.

3. Členění prvků

Máme tři kritéria, to znamená, že máme tři dimense „prostoru projekce“, každá dimense má určitou kvantifikaci, prvky prostoru projekce můžeme popsat třírozměrnou tabulkou. Obvyklým trikem zobrazíme třírozměrnou tabulku jako dvourozměrnou (třetí rozměr je popsán v políčku tabulky). Řádky označují jednotlivé úrovně podrobností, sloupce označují pohledy a v políčku tabulky je kvarteto modelů (třetí rozměr).

Tabulka 1: Schéma prvků procesu projekce

Pohled:	analytický	jednání	logický	komponent	rozložení
úroveň systém					
úroveň podsystémy					
úroveň tříd					
úroveň procedur					
úroveň kódu					

Podívejme se na tuto tabulku z několika hledisek, abychom si ujasnili, k čemu ji lze použít.

3.1 Co je prvkem tabulky?

Pavel Hruby zavádí užitečný pojem **artefakt**, který nám umožní zbavit se obtížných specifikací modelů a prvků.

Artefakt je (logicky ucelená) část postupu projekce, která má vlastní (textovou nebo grafickou) representaci. Tento pojem se obvykle kryje s pojmem model. Artefaktem je text procedury, model tříd nebo popis atributů a metod (a dalších složek) jedné třídy. Artefakt je základní prvek naší třírozměrné tabulky (tj. například výše zmíněné kvarteto se skládá ze čtyř artefaků).

Representace artefaktů. Artefakt sám je abstrakcí, která je realizována určitou representací. Ukážeme si to na kvartetu úrovně tříd. Artefaktem je model tříd (za třídně rozděleného světa se říkalo objektový model). Model tříd je jeden – jeho representace je všeobecně známá. Může být udělán tím nebo oním stylem, tak nebo onak důkladně, ale v zásadě je jeden. Jinak to je s dynamickým modelem. Máme dvojici modelů (spolupráce a sekvenční v terminologii UML), což jsou dvě representace téhož (lze je navzájem dost mechanicky převést). Máme však i další dynamický model – model akcí, který je vytvořen se zdůrazněním jiných aspektů než model sekvenční a spolupráce. Artefakt tedy není vždy pouze jeden model, může být representován i skupinou odlišných modelů, které však mají společnou charakteristiku – společný hlavní aspekt, podle kterého jsou vytvořeny (v diskutovaném případě tímto aspektem je dynamika).

Výběrovost artefaktů. Artefakt je možnost existence, nikoliv nutnost. Artefakty (místa v tabulce) říkají, že takový prostředek je možný a že má smysl, ne že musí existovat. Metodika, která by vyžadovala vyplnění všech míst v tabulce (vytvoření všech možných artefaktů) by byla velmi školometská. Typické pro metodiky je, že tuto tabulku vyplňují dosti řídce (méně než z 50%).

3.2 Výčet pohledů (dimense C)

Uvedený výčet pohledů (analytický, jednání, logický, komponent, rozložení) je pragmatický, je současný v tom smyslu, že odpovídá UML. Lze si snadno představit i jiný výčet – ovšem

myslím, že uvedené pohledy tvoří solidní základ. Pokud některý pohled nepoužíváme, znamená to, že žádný artefakt tohoto pohledu není obsazen (málokterá metodika používá všechny tyto pohledy). Je-li zapotřebí nějaký pohled navíc, tak se přidá.

Všimněte si, že pořadí pohledů zhruba odpovídá pořadí etap projekce. Nelze to navzájem jednoznačně přiřadit (jeden pohled se může vyskytovat ve více etapách) ani není závazné pořadí, to vše závisí na konkrétní metodice, zhruba to však odpovídá.

3.3 Výčet úrovní podrobností (dimense A)

První a poslední úroveň je daná, tedy v úvahu připadají pouze modifikace vypuštění a vložení úrovně podrobností. Vypustit úroveň lze nepoužitím příslušných artefaktů, vložit úroveň je pravděpodobné jen v tom smyslu, že úroveň podsystémů se rozčlení na dvě – ani jedna z těchto operací nemění smysl a použití tabulky.

3.4 Charakter úrovní podrobností (a dimense B)

Základní kvarteto artefaktů je vytvořeno na základě pojmu modelu, což znamená, že je plně využitelné na úrovni tříd a výše. Jeho zavedení se nehodí pro úroveň procedur a strojového kódu.

Úroveň programového kódu je jasná, programový kód je text, žádné modely ani životní cykly tu nejsou zapotřebí – kód sám považuji za statický model. To, co by se dalo považovat za dynamický model kódu mi splývá s životním cyklem procedur. Jiná situace je s druhou (nemodelující) dvojicí artefaktů kvarteta. Popis prvků a životní cyklus prvků je vlastně popisem programových příkazů a jejich životních cyklů – a to je danost, to je realizační platforma.

Úroveň procedur. Artefakty v úrovni procedur jsou závislé na vstupní podmínce – omezit se na objektovou orientaci. (Jinou závislostí na objektovém paradigmatu je úroveň tříd.) Z objektového stylu řešení vyplývá, že procedury jsou relativně malé a jejich vazby nejsou složité. Rozumí se vazby uvnitř jednoho objektu, vazby mezi procedurami různých objektů se chápou jako meziobjektové vazby a patří o úroveň výše.

V [1] jsou uvedeny dva artefakty této úrovně, a to „procedura“ a „algoritmus procedury“, což odpovídá popisu prvku a životnímu cyklu prvku. Neberou se v úvahu modely (sítě), jejichž prvky by procedury byly. Z neobjektových přístupů takové modely jsou známy (například graf volání procedur jako statický model a strukturogram jako dynamický model). Ilustrujme si poslední dvě úrovně výřezem z tabulky 1):

Tabulka 2: Ilustrace úrovně procedur a programového kódu

Pohled:	analytický	jednání	logický
úroveň procedur	**Analytická procedura** **Analytický algoritmus**		**Procedura** **Algoritmus procedury**
úroveň kódu	**Prototyp**		**Zdrojový kód**

4. Vztahy

Chceme-li rozumně definovat vztahy mezi artefakty na abstraktní úrovni tak, abychom mohli porovnávat metodiky, dostáváme se do potíží. Relativně nejlépe lze zvládnout vztahy mezi artefakty kvarteta (uvnitř dimense B), dokonce můžeme určit kvantifikaci vztahů – viz Obrázek 1: Kvarteto modelů. Ani zde není vše jednoznačné, v [1] se uvádí, že jeden prvek může mít několik životních cyklů, s čímž samozřejmě nesouhlasím. Klasifikace ostatních vztahů (v rámci dimensí A a C) je obtížné. V zásadě můžeme zvolit následující základní přístupy.

Formální přístup. Mezi artefakty zcela určitě platí vztahy „zpodrobnění“ (refine) a „korespondence“ (trace, genidentita). Mezi konkrétními artefakty to jsou velmi důležité vztahy. Je podstatné, která skupina tříd je zpodrobněním určitého podsystému a jaké skupině tříd logického pohledu koresponduje třída analytického pohledu.

Ale na abstraktní úrovni vhodné pro klasifikaci metodik jsou tyto vztahy k ničemu, protože vyplývají z umístění v tabulce – artefakty na vyšší úrovni (méně podrobné) jsou vždy zpodrobněny artefakty na nižší (podrobnější) úrovni. Artefaktu z jednoho pohledu vždy koresponduje nějaký artefakt jiného pohledu. Tedy tento přístup klasifikaci metodik nijak nepomůže.

Vágní přístup. V [1] jsou zavedeny tyto vztahy: „instance“ (rozumí se jeden artefakt je instancí jiného), „realizace“ (rozumí se, že poznatky jednoho artefaktu ovlivní realizaci jiného), „vazba“ (bind); také se používá „korespondence“ a „zpodrobnění“.

Vztahy „realizace“ a „vazba“ jsou zřetelně vágní, v podstatě je můžeme zavést mezi libovolnými artefakty.

„Instance“ je použita jen v jednom typu vazeb – jsou to vazby mezi „typem jednání“ (z pohledu jednání) a „dynamickým modelem“ (interakčním modelem logického pohledu) na téže úrovní podrobností. To je zřetelně špatně, stěží může „typ jednání podsystému“ mít jako instanci dynamický model, který zahrnuje několik podsystémů. Správně je úhlopříčná vazba – typ jednání má jako instanci dynamický model pohledu na nejblíže nižší úrovni podrobností. Konkrétně „typ jednání podsystému“ má za instanci „dynamický model tříd“. Takto definovaný vztah „instance“ je správně, ve skutečnosti se však používá vztah mezi „životním cyklem typu jednání (tj. scénářem) a dílčím dynamickým modelem; tento vztah lze spíše označit jako „realizaci“
Tyto potíže vedou k otázce: **Je zapotřebí definovat abstraktní vztahy mezi artefakty?**

5. Klasifikace metodik

Podívejme se tedy, jak lze popsat metodiky pouze pomocí výčtu používaných artefaktů. Níže uvedené tabulky jednotlivých metod jsou převzaty z [1]. Použité artefakty jsou označeny zvýrazněním příslušných obdélníčků.

Tabulka 3: Metodika Shlaer-Mellor

Pohled:	analytický	jednání	logický	komponent	rozložení
úroveň domén					
úroveň podsystémy					
úroveň tříd					
úroveň procedur					
úroveň kódu					

Nejjednodušší metodika je Shlaer-Mellor. Omezuje se na logický (objektový) pohled, na vyšších úrovních (méně podrobných) se omezuje pouze na statické modely. Již z tohoto povrchního pohledu vyplývá omezení metodiky na nekomplikované systémy.

Tabulka 4: Metoda Fusion

Pohled:	analytický	jednání	logický	komponent	rozložení
úroveň systém					
úroveň podsystémy					
úroveň tříd					
úroveň procedur					
úroveň kódu					

Úplnější metodikou je Fusion. Liší se dvěma věcmi:

- modelováním uživatelských požadavků na systém (pohled jednání),
- důrazem na systém jako celek (po vytvoření modelu systému jako celku se rychle přejde na úroveň tříd).

Lze z toho usoudit, že metodika se soustředí na tvorbu nových systémů s malou dynamickou složitostí (absence životních cyklů tříd a malý důraz na podsystémy).

Tabulka 5: Metodika Rational Unified Process

Pohled:	analytický	jednání	logický	komponent	rozložení
úroveň systém					
úroveň podsystémy					
úroveň tříd					
úroveň procedur					
úroveň kódu					

Metodika RUP (Rational Unified Process) má tyto hlavní charakteristiky:

- vychází z uživatelských požadavků,
- nezabývá se systémem jako celkem ale soustředí se na podsystémy,
- klade velký důraz na design, což implikuje
  - velké systémy (komponentový pohled),
  - systémy rozložené na více počítačů (pohled rozložení).

Tyto charakteristiky odpovídají hlavnímu směru metodiky RUP – je určena pro rozvoj existujících systémů a ne pro vznik nových.

6. Závěr

Třírozměrná tabulka artefaktů se ukazuje jako účinný prostředek pro charakterizování a porovnání metodik. Dá se využít i jinými způsoby, například pro přehledný popis konkrétní metodiky a pro návrh nových metodik, diskuse těchto témat se však vymyká rozsahu tohoto článku.

Literatura

1. Hruby P.: Software Development Artifacts with UML, JOOP vol.12, no. 9, 2000, ISSN 1097-1408