Fakulta Informatiky a Informačných Technológií
Slovenská Technická Univerzita
Bratislava

Študijný odbor: Softvérové inžinierstvo

Bc. Ján Šnirc

Integrácia modelovania vlastností do UML

Diplomová práca

Vedúci diplomovej práce: Ing. Valentino Vranić PhD.
December, 2005

ANOTÁCIA

Slovenská technická univerzita v Bratislave
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

Študijný odbor: Softvérové inžinierstvo
Autor: Bc. Ján Šnirc
Diplomová práca: **Integrácia modelovania vlastností do UML**
Vedenie diplomovej práce: Ing. Valentino Vranić PhD.
2005, December

Modelovanie vlastností predstavuje dôležitý prístup zaoberajúci sa hierarchickým spôsobom variabilitou na abstraktnej úrovni. Tento prístup je široko využívaný pri vývoji rodín softvérových systémov. UML predstavuje štandardný modelovací jazyk a preto integrácia modelovania vlastností do neho je nanajvýš aktuálna. Cieľom tohto projektu je integrácia modelovania vlastností do UML s ohľadom na abstraktnosť jeho prvkov, na ktorej je modelovanie vlastností založené. To je dosiahnuté odvodením prvkov modelu vlastností zo základných prvkov metamodelu UML. Modelovacie nástroje takto rozsiahle rozšírenie metamodelu nepodporujú a preto bolo rozšírenie implementované prostredníctvom profilu jazyka UML, ktoré môže byť jednoducho použité väčšinou súčasných modelovacích nástrojov.

ANNOTATION

Slovak University of Technology Bratislava
FACULTY OF INFORMATICS AND INFORMATION TECHNOLOGIES

Degree Course:	Software engeneering
Author:	Bc. Ján Šnirc
Thesis:	**Integrating Feature Modeling into UML**
Supervisor:	Ing. Valentino Vranić PhD
2005, December	

Feature modeling is an important approach to dealing with variability at an abstract level in a hierarchical manner. It is extensively used in software product lines. With UML as a standard modeling language, it is important to integrate feature modeling into it. The goal of this project is an integration of feature modeling into UML with respect to abstractness of feature modeling elements around which is feature modeling based.This is achieved by deriving feature modeling elements from the deeper levels of the UML metamodel. UML modeling tools do not support such extensive modifications of the UML metamodel, we implemented our extension as a UML profile that can be used as a solution at hand for most contemporary UML modeling tools.

PREHLÁSENIE

Týmto čestne prehlasujem, že diplomovú prácu „Integrácia modelovania vlastností do UML“, vrátane príloh a zdrojových kódov som vypracoval sám, na základe mojich vedomostí a použitej literatúry, uvedenej v závere práce.

V Bratislave, 12. decembra 2005 Bc. Ján Šnirc

POĎAKOVANIE

Na tomto mieste by som chcel poďakovať vedúcemu diplomovej práce Ing. Valentinovi Vranićovi PhD. za jeho rady a pomoc pri riešení diplomovej práce

Obsah

1 Úvod 1
2 Prehľad notácií modelovania vlastností 3
2.1 Diagram vlastností v notácii FODA 3
2.1.1 Voliteľné vlastnosti 3
2.1.2 Alternatívne vlastnosti 3
2.1.3 Povinné vlastnosti 4
2.2 Diagram vlastností v notácii Czarneckého-Eiseneckera 4
2.2.1 Alebo-vlastnosti 4
2.2.2 Normalizácia diagramu vlastností 5
2.2.3 Bod variácie 6
2.3 Diagram vlastností v notácii podľa Riebisha 6
2.4 Diagram vlastností v rozšírenej notácii Czarneckého-Eiseneckera 8
2.5 Modelovanie vlastností pomocou návrhových priestorov 9
2.6 Diagram vlastností metódy FORM 11
3 Prehľad notácií rozširujúcich UML o podporu modelovania vlastností 13
3.1 Diagram vlastností metódy FeatuRSEB 13
3.2 Rozšírenie UML o modelovanie vlastností podľa Claussa 14
3.3 Rozšírenie UML o modelovanie vlastností podľa Robaka 16
3.4 Porovnanie analyzovaných notácií 16
4 Identifikovanie množiny relevantných prvkov diagramu vlastností 19
4.1 Prvky notácie FODA 19
4.2 Prvky notácie Czarneckého–Eiseneckera 19
4.3 Prvky notácie podľa Riebischa 20
4.4 Prvky notácie modelovania vlastností pomocou návrhových priestorov 20
4.5 Prvky notácie FORM 20
4.6 Prvky rozšírenej notácie Czarneckého-Eiseneckera 21
5 Integrácia modelovania vlastností na základe rozšírenia metamodelu UML 23
5.1 Metamodel UML 23
5.2 Opis metamodelu UML 25
5.2.1 Abstraktná syntax 25
5.2.2 Formálne pravidlá 26
5.2.3 Sémantika 26
5.2.4 Štandardné prvky 26
5.2.5 Poznámky 26
5.3 Rozšírenie metamodelu jazyka UML o prvky modelu vlastností 26
6 Integrácia modelovania vlastností na základe profilu UML 33
6.1 Profil jazyka UML 33
6.2 Profil jazyka UML podporujúci modelovanie vlastností 33
6.2.1 Identifikovaná podmnožina metamodelu UML 34
6.2.2 Prvky profilu modelu vlastností 36
6.3 Vizualizácia profilu 41
7 Zhodnotenie 43
8 Použitá literatúra 45
Príloha A Špecifikácia elementov profilu 47
Príloha B Elektronické médium 57
Install\ 57
Docs\ 57

Zoznam obrázkov

Obr. 2-2-1 Diagram vlastností metódy FODA 3
Obr. 2-2-2. Diagram vlastností FODA obsahujúci dve sady alternatívnych vlastností 4
Obr. 2-3 Porovnanie diagramu vlastností - notácie FODA a Czarneckého-Eiseneckera 4
Obr. 2-4 Diagram vlastností - alebo vlastnosti 5
Obr. 2-5 Diagram vlastností - normalizácia diagramu s voliteľnou alternatívnou vlastnosťou 5
Obr. 2-6 Diagram vlastností - normalizácia diagramu s voliteľnou alebo vlastnosťou 5
Obr. 2-7 Kardinality notácie FODA 6
Obr. 2-8 Notácia vlastností podľa Riebisha 7
Obr. 2-9 Notácia vlastností podľa Riebisha 7
Obr. 2-2-10 Kardinalita vlastností 8
Obr. 2-2-11 Skupina vlastností s kardinalitou 8
Obr. 2-12 Návrhové priestory - Model vlastností vykurovacieho systému [6] 9
Obr. 2-13 Návrhové priestory - Alebo-vlastnosť [6] 10
Obr. 2-14 Návrhové priestory - vzťah vyžadovaný [6] 10
Obr. 3-1 Diagram vlastností FeatuRSEB: systém telefónnych služieb [3] 14
Obr. 3-2 Rozšírený UML diagram vlastností: Systém na spracovanie objednávok [10] 15
Obr. 3-3 Rozšírený UML diagram vlastností: Systém na spracovanie objednávok [10] 15
Obr. 3-4 UML komponent diagram - poštový predplatný systém [12] 16
Obr. 3-5 Porovnanie analyzovaných notácií 17
Obr. 4-1 Prvky notácie FODA 19
Obr. 4-2 Prvky notácie Czarneckého-Eiseneckera 19
Obr. 4-3 Prvky notácie Riebisha 20
Obr. 4-4 Prvky notácie návrhových priestorov 20
Obr. 4-5 Prvky notácie FORM 21
Obr. 4-6 Prvky rozšírenej notácie Czarneckého Eiseneckera 21
Obr. 5-1 Vrstvy modelu UML 23
Obr. 5-2 UML balíčky podporujúce štrukturálne modelovanie 23
Obr. 5-3 UML balíčky podporujúce modelovanie správania 24
Obr. 5-4 UML balíčky podporujúce doplnkové konštruktory 24
Obr. 5-5 Balíček Kernel 27
Obr. 5-6 Základný diagram balíčka FM Constructs 28
Obr. 5-7 Koncept, vlastnosť, vrstva rozšíreného metamodelu UML 28
Obr. 5-8 Abstraktné vzťah rozšíreného metamodelu UML 28
Obr. 5-9 Vlastnosť Subfeature Dependency rozšíreného metamodelu UML 29
Obr. 5-10 Stereotypy vlastnosti Subfeature Dependency 29
Obr. 5-11 Vzťah Group Dependency rozšíreného metamodelu UML 30
Obr. 5-12 Príklad modelovania vzťahu Group Dependency 30
Obr. 5-13 Stereotyp alternative vzťahu group dependency 30
Obr. 5-14 Stereotyp or vzťahu group dependency 31
Obr. 5-15 Prvok corelation rozšíreného metamodelu UML 31
Obr. 5-16 Príklad využitia korelácie 32
Obr. 5-17 Vzťah implemented rozšíreného metamodelu UML 32
Obr. 5-18 Príklad modelu vlastností s implementovanými vlastnosťami 32
Obr. 6-1 Vlastnosť s viacerými skupinami alebo-vlastností vyjadrená pomocou variačných bodov 34
Obr. 6-2 Rozšírenie metatriedy dependency 35
Obr. 6-3 Rozšírenie metatried asociation a package 35

Obr. 6-4 Rozšírenie metatried components a realization 35
Obr. 6-5 Rozšírenie metatriedy artifact 36
Obr. 6-6 Povinná vlastnosť v profile UML 37
Obr. 6-7 Voliteľná vlastnosť v profile UML 37
Obr. 6-8 Korelácia v profile UML 38
Obr. 6-9 Alebo-vlastnosti v profile UML 38
Obr. 6-10 Alternatívne vlastnosti v profile UML 39
Obr. 6-11 Vzťah implementácie v profile UML 39
Obr. 6-12 Vrstvy v profile UML 40
Obr. 6-13 Kardinalita vlastností v profile UML 40
Obr. 6-14 Kardinalita skupiny vlastností v profile UML 41
Obr. 6-15 Profil UML podporujúci modelovanie vlastností 42

1 Úvod

So vzrastajúcim záujmom o rodiny softvérových systémov sa stáva potreba vyjadrenia variability abstraktným spôsobom čoraz dôležitejšia. Technika modelovania vlastností umožňuje zachytiť variabilitu konceptu hierarchickým a implementačne nezávislým spôsobom a to prostredníctvom jeho vlastností a vzťahov medzi nimi.

Koncept predstavuje abstraktnú, univerzálnu ideu, ktorá slúži na označenie kategórií entít, udalostí a vzťahov medzi nimi. Je nositeľom významu, ktorý môže byť vyjadrený viacerými spôsobmi. V prípade modelovania vlastností je kladený dôraz predovšetkým na identifikovanie spoločných a rozdielnych vlastností.

Vlastnosť podľa Kanga [1] reprezentuje významný, používateľom viditeľný aspekt, alebo charakteristiku softvérového systému v danej doméne. Pod vlastnosťou (feature) rozumieme vlastnosť (property) systému, ktorá je dôležitá pre zákazníka. Pod podobnými vlastnosťami rozumieme tie, ktoré sa nachádzajú vo všetkých inštanciách konceptu, pod odlišnými tie, ktoré sa nachádzajú iba v niektorých. Vo všeobecnosti rozlišujeme nasledovné vlastnosti [2] :

- Konkrétne vlastnosti, ako napríklad algoritmy, ktoré sú implementované ako samostatné komponenty.
- Aspektuálne vlastnosti ako synchronizácia, ktoré môžu byť modularizované
- Abstraktné vlastnosti ako požiadavky na výkon, ktoré sú prenesené na konfiguráciu komponentov a/alebo aspektov
- Skupinové vlastnosti ako variačné body, ktoré sú prenášané na rozhrania kompatibilných komponentov, alebo majú čisto organizačný charakter bez implicitných požiadaviek.

Výstupom modelovania vlastností je:

Diagram vlastností - kľúčový prvok modelovania vlastností. Predstavuje jednoduchú a intuitívnu grafická notáciu zachytávajúcu variabilitu vlastností a vzťahov medzi nimi spôsobom nezávislým od implementačných techník.

Doplňujúce informácie:

- popis vlastnosti
- dôvod zahrnutia jednotlivých vlastností
- priorita
- zainteresované osoby (stakeholders)
- obmedzenia (composition rules)
- dostupnosť (availability sites)
- doba viazania (binding sites)

Cieľom tejto práce je integrácia modelovania vlastností do UML, ktoré predstavuje v súčasnosti štandardný modelovací jazyk. Pri integrácii je potrebné prihliadať predovšetkým na abstraktnosť konceptu a vlastností od ich implementácie.

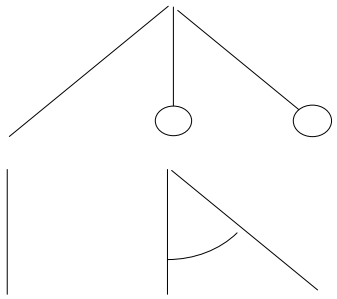
Práca je rozdelená na tri hlavné časti. Úvodná časť analyzuje existujúce notácie modelovania vlastností vrátane notácii rozširujúcich UML o modelovanie vlastností. Nasleduje identifikácia množiny relevantných prvkov analyzovaných notácií a v poslednej časti uvádzam integráciu tejto množiny na úrovni metamodelu a profilov jazyka UML.

2 Prehľad notácií modelovania vlastností

Obsahom tejto kapitoly je prehľad základných notácii modelu vlastností. Notácie sú zoradené chronologicky a ich rozšírenia na seba plynule nadväzujú.

2.1 Diagram vlastností v notácii FODA

Diagram vlastností, ktorý bol prvý krát predstavený práve prostredníctvom metódy FODA, predstavuje stromovú štruktúru, ktorej koreň reprezentuje koncept a uzly opisujú vlastnosti, resp. ich pod-vlastnosti. Vzťah medzi rodičovskou vlastnosťou a vlastnosťou potomka vo všeobecnosti indikuje, že vlastnosť potomka vyžaduje zahrnutie rodičovskej vlastnosti. V opačnom prípade je pod-vlastnosť nedosiahnuteľná, t.j. nie je ju možné zaradiť do inštancie konceptu. Konkrétne typy vlastností, vzťahy medzi nimi a ich význam je opísaný v nasledujúcom texte.



Obr. 2-2-1 Diagram vlastností metódy FODA

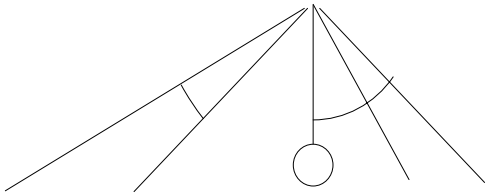
2.1.1 Voliteľné vlastnosti

Pod voliteľnou vlastnosťou (optional feature) rozumieme takú vlastnosť, ktorá môže, ale nemusí byť zahrnutá do inštancie konceptu. Jedná sa o vlastnosť, ktorá pridáva hodnotu hlavným vlastnostiam produktu. Graficky je znázornená priradením prázdneho kruhu k danému uzlu (Obr. 2-2-1, vlastnosti f2 a f3).

2.1.2 Alternatívne vlastnosti

Oblúk spájajúci dve, alebo viaceré vlastnosti označuje množinu vlastností, ktoré označujeme ako alternatívne (alternative features). Z danej množiny vlastností je do inštancie konceptu zahrnutá práve jedna z nich. Koncept môže obsahovať viaceré množiny alternatívnych vlastností, pričom tie môžu byť voliteľné, alebo povinné (Obr. 2-2-2)

f1



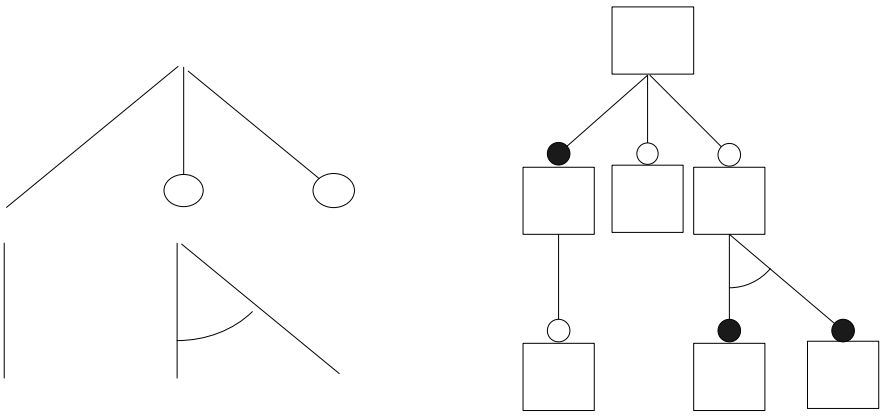
Obr. 2-2-2. Diagram vlastností FODA obsahujúci dve sady alternatívnych vlastností

2.1.3 Povinné vlastnosti

Vlastnosti bez špeciálnej notácie označujeme ako povinné (Obr. 2-2-1, vlastnosti f1 a f4). Povinné vlastnosti (mandatory features) musia byť zahrnuté v každej inštancii konceptu. Predstavujú hlavné vlastnosti, ktoré identifikujú produkt.

2.2 Diagram vlastností v notácii Czarneckého-Eiseneckera

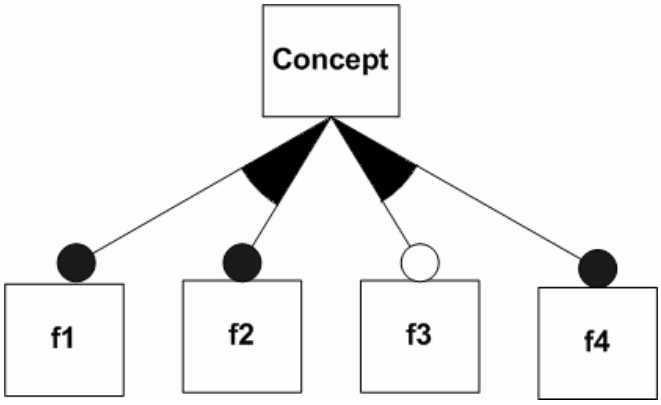
Notácia podľa Czarneckého-Eiseneckera [9] predstavuje mierne upravenú a rozšírenú základnú notáciu diagramu vlastností FODA. Pôvodná notácia bola obohatená o alebo-vlastnosti (or-features), normalizáciu diagramu vlastností ako aj ďalšie prvky uvedené v tejto kapitole. V súčasnosti predstavuje široko akceptovanú notáciu, z ktorej vychádzajú aj ostatné rozšírenia uvedené ďalej v texte.



Obr. 2-3 Porovnanie diagramu vlastností - notácie FODA a Czarneckého-Eiseneckera

2.2.1 Alebo-vlastnosti

Alebo-vlastnosti (or-features) predstavujú množinu vlastností, z ktorých je do inštancie konceptu zahrnutá jedna, alebo viaceré z vlastnosti. Množina alebo-vlastností je graficky znázornená plným oblúkom (Obr. 2-4) Môže obsahovať okrem povinných vlastností aj voliteľné vlastnosti, ale ako bude neskôr uvedené, počas normalizácie diagramu vlastností budú množiny alebo-vlastností obsahujúce jednu alebo viacero voliteľných vlastností, nahradené voliteľnými vlastnosťami.

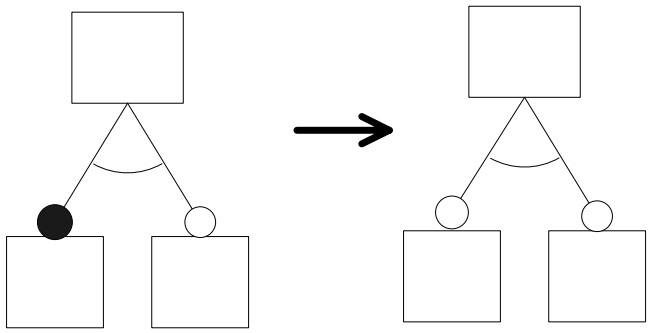


Obr. 2-4 Diagram vlastností - alebo vlastnosti

2.2.2 Normalizácia diagramu vlastností

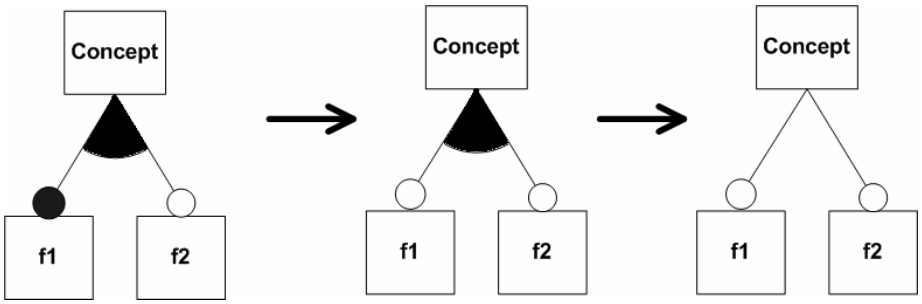
Uzol v diagrame vlastností môže obsahovať pod-uzly s nasledovnými vlastnosťami: povinné, voliteľné, alternatívne, alebo-vlastnosti, voliteľné alternatívne vlastnosti a voliteľné alebo-vlastnosti.

Pri bližšom pohľade na alternatívne vlastnosti vidíme, že ak jedna vlastnosť z množiny alternatívnych vlastností je voliteľná, výsledkom je zvyšné vlastnosti sa správajú akoby boli tiež voliteľné a preto ich môžeme jednoducho nahradiť (Obr. 2-5).



Obr. 2-5 Diagram vlastností - normalizácia diagramu s voliteľnou alternatívnou vlastnosťou

Podobne, ak je jedna z vlastností množiny alebo-vlastností voliteľná, výsledkom je, že zvyšné vlastnosti sa správajú akoby boli voliteľné a preto ich môžeme nahradiť. Množina alebo-vlastností, ktorá obsahuje iba voliteľné vlastnosti je ekvivalentná ďalej s voliteľnými vlastnosťami (Obr. 2-6).



Obr. 2-6 Diagram vlastností - normalizácia diagramu s voliteľnou alebo vlastnosťou

2.2.3 Bod variácie

Bod variácie (variation point) predstavuje označenie pre uzol, ku ktorému sú pripojené voliteľné, alternatívne, voliteľne alternatívne, alebo alebo-vlastnosti, ktoré sú spoločne označované ako premenlivé vlastnosti (variable features). Bod variácie predstavuje v diagrame vlastností miesto, v ktorom je zachytená variabilita vlastností konceptu a práve z tohto dôvodu je jeho explicitné označenie dôležité. Rozoznávame niekoľko typov bodov variácie

Dimenzia – vlastnosť, alebo koncept, ktorý má ako priamych potomkov alternatívne pod-vlastnosti.

Dimenzia s voliteľnými vlastnosťami - vlastnosť, alebo koncept, ktorý má ako priamych potomkov alternatívne voliteľné pod-vlastnosti.

Rozširujúci bod – vlastnosť, alebo koncept, ktorého aspoň jeden priamy potomok je voliteľná pod-vlastnosť (vlastnosť), alebo množina priamych alebo-pod-vlastností.

Rozširujúci bod s voliteľnými vlastnosťami – vlastnosť, alebo koncept, ktorý má ako priamych potomkov voliteľné pod-vlastnosti.

Rozširujúci bod s alebo-vlastnosťami – vlastnosť, alebo koncept, ktorý má ako priamych potomkov alebo-vlastnosti.

2.3 Diagram vlastností v notácii podľa Riebisha

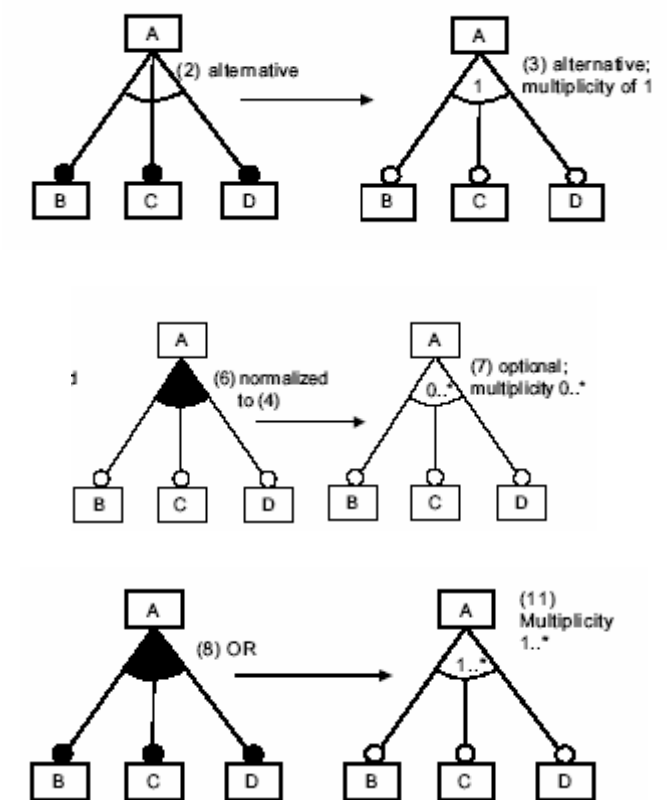
Kardinalita (multiplicity) je štandardnou modelovacou technikou. Avšak notácia diagramu vlastností metódy FODA[1], tak isto ako základná notácia Czarneckého-Eiseneckera [9] podporuje iba nasledovné kardinality

0..1	Pre voliteľnú vlastnosť
1	Pre povinnú vlastnosť
0..*	Pre voliteľné alebo-vlastnosti
1..*	Pre povinné alebo-vlastnosti

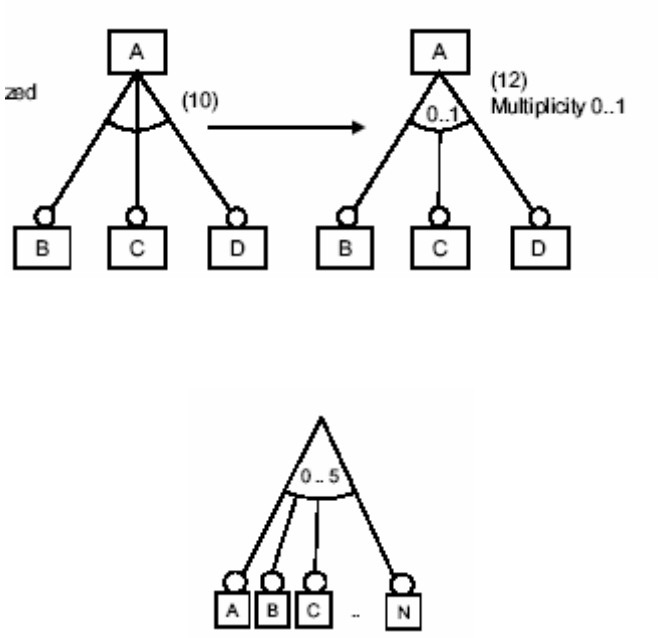
Obr. 2-7 Kardinality notácie FODA

V praxi však vznikajú požiadavky aj na ďalšie kardinality. Logickým riešením je modifikácia existujúcej notácie [5] tak, aby podporovala všeobecnú kardinalitu a bola unifikovaná vzhľadom na jazyk UML, v ktorom je kardinalita jednou zo základných modelovacích techník.

Uvedená notácia bola upravená na základe nasledujúcich príkladov, pričom kardinalita môže byť všeobecne vyjadrená ako (n..m) , $(n,m) \in N$



Obr. 2-8 Notácia vlastností podľa Riebisha



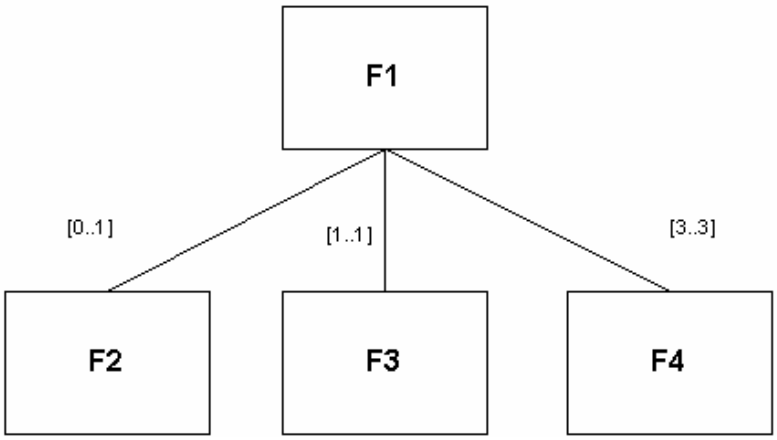
Obr. 2-9 Notácia vlastností podľa Riebisha

2.4 Diagram vlastností v rozšírenej notácii Czarneckého-Eiseneckera

Rozšírená notácia Czarneckého-Eiseneckera integruje existujúce notácie vychádzajúce z metódy FODA pomocou kardinality vlastností a skupín vlastností s kardinalitou.

Kardinalita vlastností

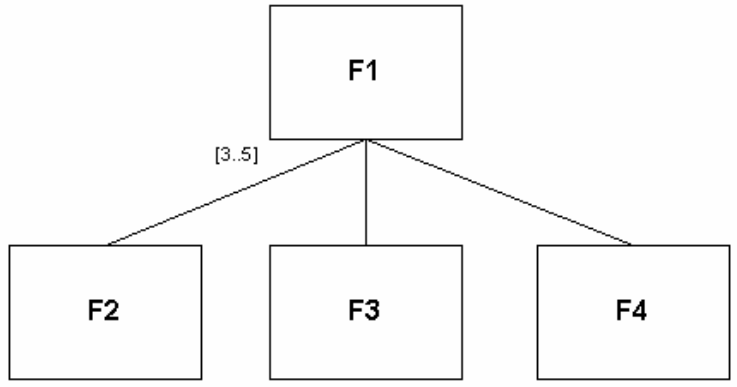
Kardinalita vlastností je vyjadrená ako komentár pridružený ku vlastnosti (napr. [1..*], alebo [5..5]). Povinné a voliteľné vlastnosti sú špeciálnymi prípadmi vlastností s kardinalitou [1..1], res. [0..1]. Zavedenie kardinality vlastností bolo motivované tiež praktickými aplikáciami, kedy kardinalita [3..3] vyjadruje zahrnutie troch inštancií vlastnosti do konceptu. Príklad modelu vlastností s kardinalitami je uvedený na Obr. 2-2-10 Kardinalita vlastností



Obr. 2-2-10 Kardinalita vlastností

Skupina vlastností s kardinalitou

Alternatívne vlastnosti metódy FODA a alebo-vlastnosti uvedené v základnej notácii Czarneckého-Eiseneckera môžu byť chápané ako mechanizmus zoskupujúci vlastnosti. Tento mechanizmus je zovšeobecnený ako množina vlastností označená kardinalitou. Tá špecifikuje interval vlastností, ktoré majú byť do konceptu zahrnuté. Alternatívne vlastnosti a alebo-vlastnosti sa tak stávajú špeciálnym prípadom skupiny s kardinalitou. Príklad modelu vlastností so skupinami s kardinalitami je uvedená na Obr. 2-2-11 Skupina vlastností s kardinalitou



Obr. 2-2-11 Skupina vlastností s kardinalitou

2.5 Modelovanie vlastností pomocou návrhových priestorov

Technika návrhových priestorov [6] umožňuje na rozdiel od pôvodnej techniky [7] návrhových priestorov zachytiť okrem požiadaviek na používateľské rozhranie aj všeobecné požiadavky.

Návrhový priestor predstavuje množinu požiadaviek a návrhových rozhodnutí. Tie sú podľa kritérií podstatných pre danú doménu rozdelené na dimenzie, ktoré môžu byť

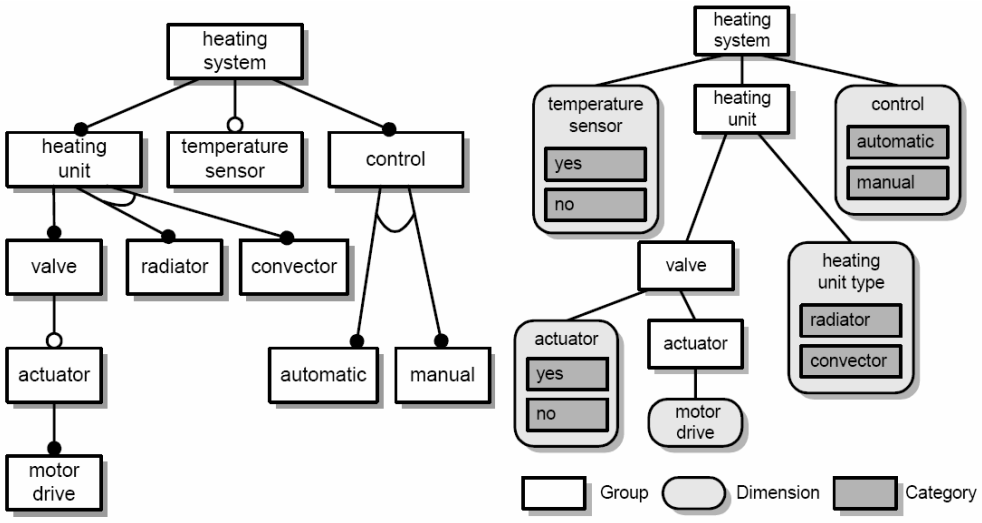
- **diskrétne**, tiež nazývané kategórie obsahujú jednotlivé návrhové alternatívy
- **spojité**, reprezentujúce celé a reálne čísla

Dimenzie, ktoré spolu súvisia sú združované do skupín, pričom skupina môže obsahovať ďalšiu skupinu, čo umožňuje vytvárať hierarchické štruktúry. Pomocou dimenzií, kategórií, skupín a vzťahov medzi nimi, popísanými v tejto kapitole je možné modelovať vlastnosti podobným spôsobom ako diagram vlastností metódy FODA [1]

Povinné vlastnosti - sú modelované ako dimenzia. Ak je povinná vlastnosť v stromovej štruktúre listom dimenzia je prázdna.

Alternatívne vlastnosti – predstavujú kategórie v jednotlivých dimenziách. Pri modelovaní alternatívnych vlastností je vhodné do modelu vložiť novú dimenziu (Obr. 2-12)

Voliteľné vlastnosti – sú modelované rovnakým spôsobom ako alternatívne vlastnosti, ale jednotlivé kategórie sú iba typu Áno/Nie.



Obr. 2-12 Návrhové priestory - Model vlastností vykurovacieho systému [6]

Vzťah medzi dimenziou a kategóriou sa nazýva korelácia. Rozlišujeme 2 druhy

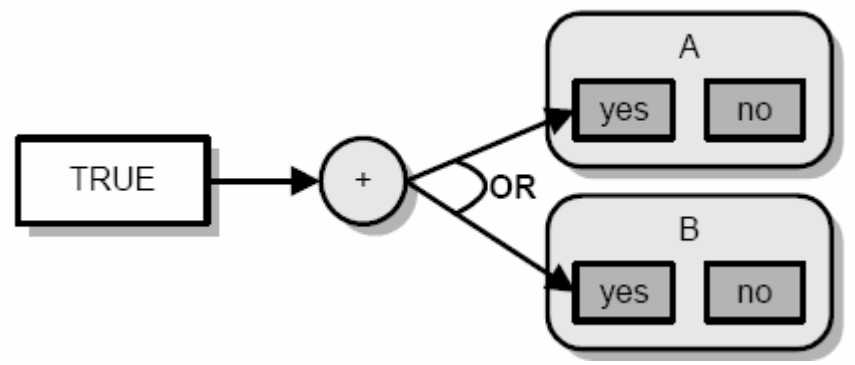
- **slabá** – číslo z intervalu <-1,1> určuje preferovaný spôsob kombinácie jednotlivých kategórií. (-1 kategórie by nemali byť spolu, +1 preferovaná kombinácia kategórií)

- **silná** – boolovská hodnota vyjadruje či sa môžu kategórie vyskytovať spolu. Predstavuje obdobu kompozičných pravidiel „required“ a „mutualy exclusive“ uvedených ako doplňujúce informácie v modeli vlastností metódy FODA.

Pomocou korelácie môžeme vyjadriť rôzne vzťahy medzi kategóriami ako napr. alebo-vlastnosti, vzťahy vyžadovaný, vylúčený, či odporúčanie ku kombinácii jednotlivých kategórií.

Alebo-vlastnosť

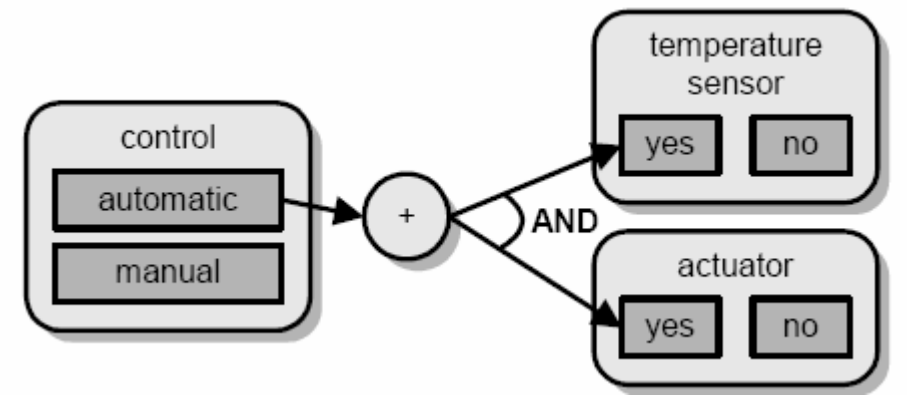
Je modelovaná pomocou silnej korelácie(Obr. 2-13) Ak je splnená podmienka musí byť zahrnutá aspoň jedna kategória „yes“ z dimenzií A a B



Obr. 2-13 Návrhové priestory - Alebo-vlastnosť [6]

Vzťah „vyžadovaný“ (required)

Je modelovaný tiež pomocou silnej korelácie. Ak je do modelu zahrnutá kategória „automatic“ musia byť tiež zahrnuté kategórie „yes“ dimenzií „temperature sensor“ a „actuator“.



Obr. 2-14 Návrhové priestory - vzťah vyžadovaný [6]

Vzťah „vylúčený“ (mutualy exclusive) - je obdobou vzťahu „vyžadovaný“, avšak silná korelácia nadobúda hodnotu -1

Odporúčanie kombinácie - tento vzťah vyjadruje odporúčanie, akým spôsobom majú byť jednotlivé kategórie navzájom kombinované. Odporúčanie je reprezentované pomocou slabej

korelácie. Vďaka tomuto odporúčaniu je možné vytvoriť preddefinovaný profil návrhového priestoru, ktorý sa ďalej modifikuje.

Sekvencie - vyjadrujú odporúčaný postup pri výbere jednotlivých možností.

2.6 Diagram vlastností metódy FORM

Metóda FODA [1], ktorá uviedla diagram vlastností ako prostriedok na správu požiadaviek bola rozšírená metódou FORM (feature-oriented reuse method) [4] o fázy návrhu a implementácie. Toto rozšírenie sa odrazilo aj v rozdelení diagramu vlastností na vrstvy podľa informácií, ktoré jednotlivé vlastnosti nesú . Model vlastností bol rozdelený na:

Vrstva schopností (capability layer) - vrstva zahŕňa funkcionálne požiadavky na systém, ako napr. poskytované služby či operácie, ako aj nefunkcionálne požiadavky ako napr. výkon systému.

Vrstva operačného prostredia (operation enviroment layer) - vrstva obsahuje atribúty konkrétneho prostredia, v ktorom bude systém nasadený. Jedná sa najmä o použitý hardvér, operačný systém, databázový systém a sieťové prostredie.

Vrstva doménových technológií (domain technology layer) - vrstva obsahuje implementačné detaily špecifické pre danú doménovú oblasť.

Vrstva implementačných techník (implementation technique layer) - vrstva obsahuje implementačné detaily, ktoré sú všeobecné a môžu byť použité pre viaceré doménové oblasti.

Diagram vlastností metódy FORM obsahuje tak ako FODA povinné, voliteľné a alternatívne vlastnosti. Ku základnému vzťahu skladá sa (composed of) pribudli ďalšie vzťahy generalizácia/špecializácia (generalization/specialization) označená bodkovanou čiarou a implementácia (implemented by) označovaná bodko-čiarkovanou čiarou.

3 Prehľad notácií rozširujúcich UML o podporu modelovania vlastností

Existujúce notácie, ktoré sa snažia viac-menej úspešne integrovať modelovanie vlastností do jazyka UML sú založené na jeho tzv. ľahkom rozšírení. Ľahké rozšírenie predstavuje prispôsobenie jazyka UML pomocou stereotypov (stereotypes), označených hodnôt (tagged values) a obmedzení (constraint).

Stereotyp [8] *„je nový modelovací element, ktorý rozširuje sémantiku metamodelu UML. Stereotyp musí byť založený na existujúcom type, alebo triede v metamodeli. Stereotyp môže rozširovať sémantiku, ale nie štruktúru typu, alebo triedy, na ktorej je založený. Niektoré stereotypy sú priamo preddefinované v UML, ďalšie môžu byť dodefinované. Stereotyp je spolu s označenými hodnotami a obmedzeniami jedným z troch rozširujúcich mechanizmov jazyka UML.“*

Označená hodnota [8] *“je explicitná definícia vlastnosti ako dvojica meno-hodnota. V označenej hodnote je meno referované ako označenie. Niektoré označenia sú v UML už preddefinované, ďalšie môžu byť dodefinované.“*

Obmedzenie [8] *„je sémantická podmienka, alebo zákaz. Niektoré obmedzenia sú už v UML preddefinované, ďalšie môžu byť dodefinované“*.

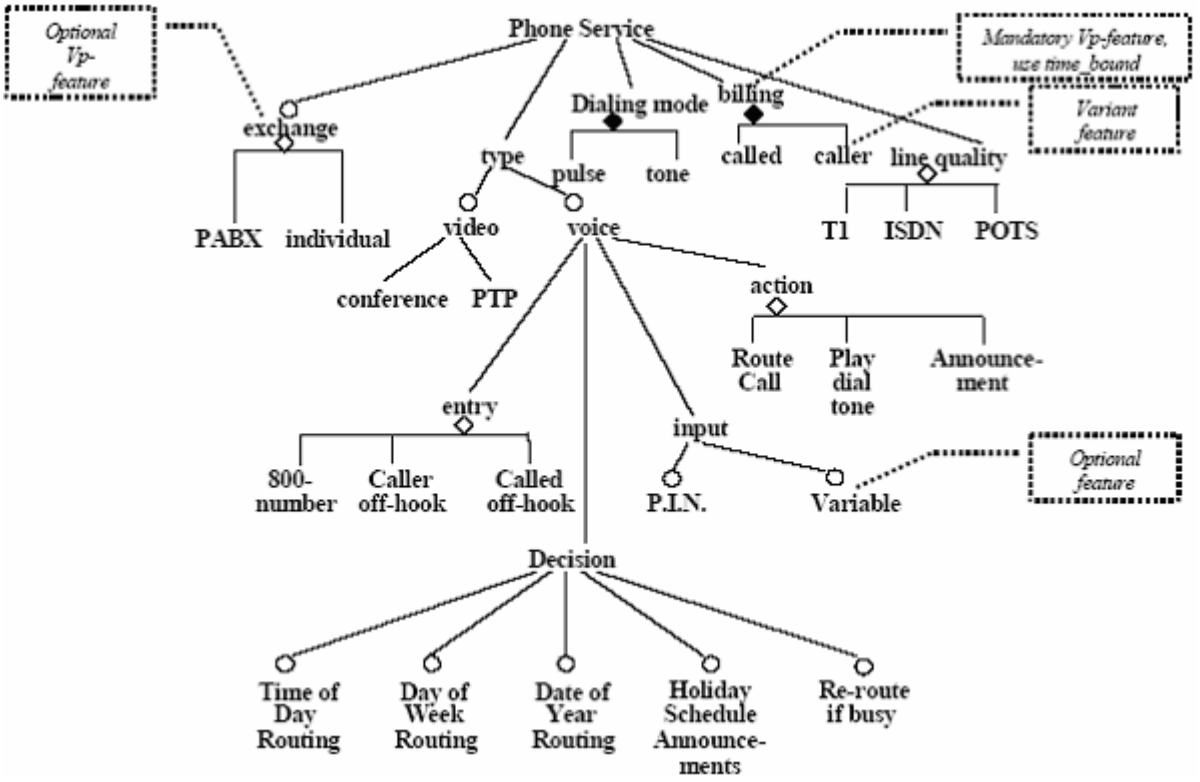
Prehľad existujúcich notácií využívajúcich ľahké rozšírenie jazyka UML na podporu modelovania vlastností je uvedený v nasledujúcom texte.

3.1 Diagram vlastností metódy FeatuRSEB

Metóda FeatuRSEB [3] predstavuje integráciu metódy modelovania vlastností FODA a prístup metódy RSEB založenej na prípadoch použitia. Základný rozdiel medzi uvedenými metódami je nasledovný:

- **diagram vlastností** je orientovaný na znovu-používateľa
- **diagram prípadov použitia (use case diagram)** je orientovaný na používateľa.

Diagram vlastností metódy FeatuRSEB nahradil diagram prípadov použitia v metóde RSEB, t.j. predstavuje „+1“ model, ktorý spája spolu ostatné diagramy (UC diagram, analyze diagram, ...). Je zapísaný v jazyku UML a pozostáva z jednotlivých vlastností a vzťahov, ktorými sú zviazané. Vlastnosti sú reprezentované triedami so stereotypom <<feature>>, pričom atribút „optional“ na danej triede vyjadruje či sa jedná o voliteľnú vlastnosť alebo nie.



Obr. 3-1 Diagram vlastností FeatuRSEB: systém telefónnych služieb [3]

Medzi vlastnosťami existuje vzťah „pozostáva z“ a vzťah vyjadrujúci jednotlivé alternatívy, ktorý je reprezentovaný bodom variácie. Je to abstraktná vlastnosť, ktorá je ďalej zjemnená jednotlivými alternatívnymi, resp. alebo-vlastnosťami. Graficky je vyjadrená diamantom nad danou vlastnosťou. Podľa času viazania variačného bodu poznáme vlastnosti:

Viazané v čase znovu použitia označované bielym diamantom (agregácia) ako XOR vlastnosť. Iba jedna alternatíva môže byt zahrnutá do inštancie.

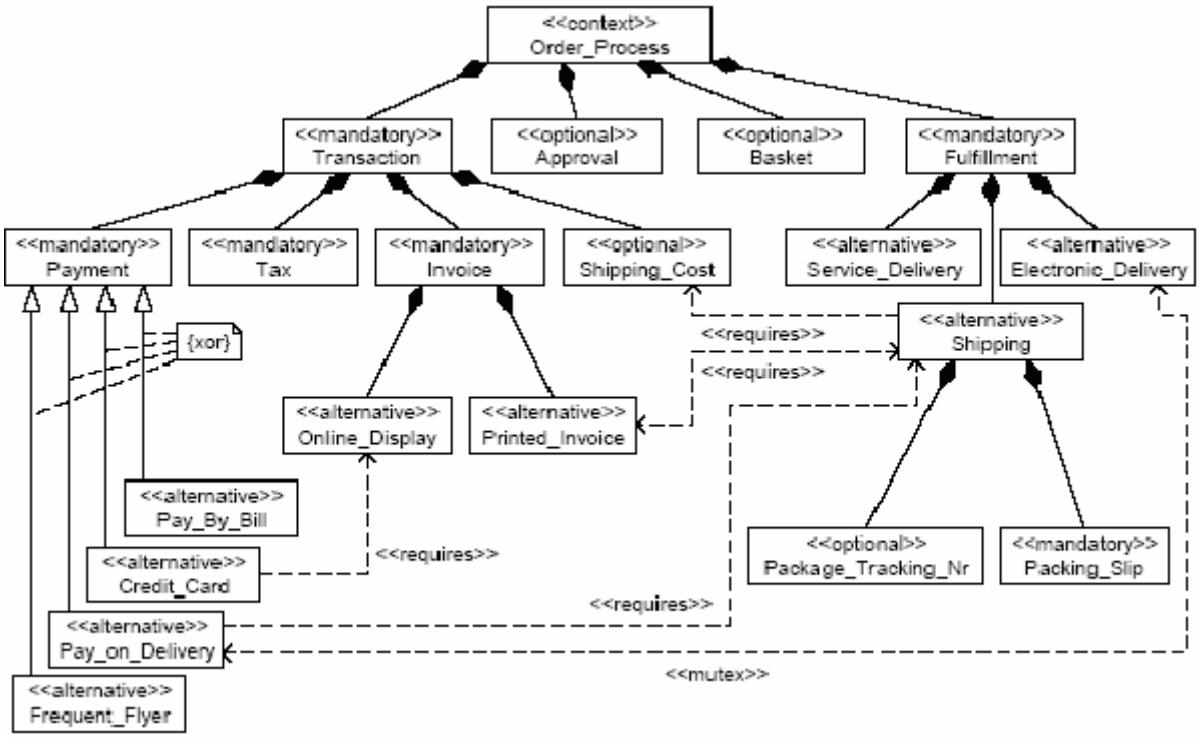
Viazané v čase použitia označované čiernym diamantom (kompozícia) ako OR vlastnosť. Zahrnuté môžu byt viaceré alternatívy.

Obmedzenia môžu byť vyjadrené pomocou doplňujúcich informácií ako aj priamo v diagrame pomocou asociácie s príslušným stereotypom. Tento spôsob je však vhodný iba pre diagramy s malým počtom elementov, kde vyjadrenie obmedzení priamo v diagrame vlastností nespôsobí jeho neprehľadnosť.

3.2 Rozšírenie UML o modelovanie vlastností podľa Claussa

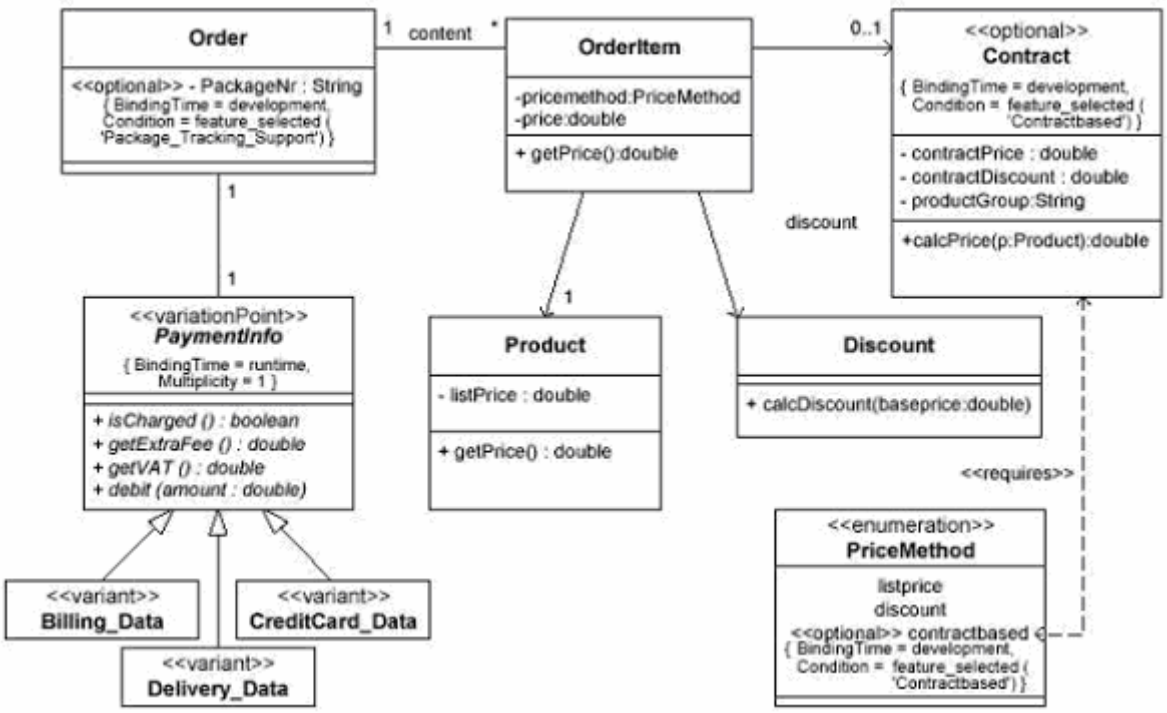
Rozšírenie UML o podporu modelovania vlastností podľa Claussa [10], predstavuje tzv. ľahké rozšírenie UML pomocou stereotypov a označených hodnôt.

Jednotlivé typy vlastnosti sú reprezentované v jazyku UML ako triedy som stereotypmi <<context>>, <<mandatory>>, <<optional>> a <<alternative>>. Vzťahy medzi nimi sú modelované ako kompozícia (agregácia) a generalizácia. Na odlíšenie OR a XOR vlastností je použitá poznámka asociovaná ku danému vzťahu. Samotný diagram vlastností je pomocou týchto rozšírení mapovaný do diagramu tried jazyka UML (Obr. 3-2).



Obr. 3-2 Rozšírený UML diagram vlastností: Systém na spracovanie objednávok [10]

Obmedzenia na vzájomnú kombináciu jednotlivých vlastností (mutex, required) sú vyjadrené pomocou asociácií so stereotypom <<mutex>> pre vzájomne sa vylučujúce vlastnosti a so stereotypom <<required>> pre vlastnosti, ktoré musia byť zahrnuté spolu. Obmedzenia môžu jednostranné, t.j. platí až keď je zahrnutá vlastnosť, ktorej vychádza obmedzenie, alebo obojstranné, ktoré platí ak je zahrnutá ľubovoľná vlastnosť. UML diagram tried môže byť pomocou stereotypov ďalej rozšírený o variačné body a jednotlivé varianty (Obr. 3-3).

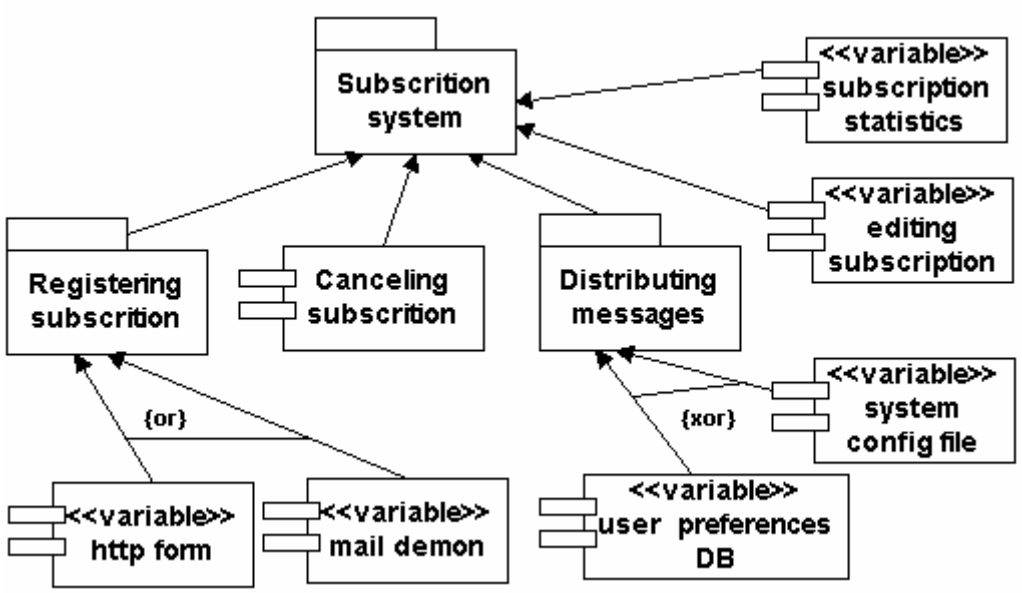


Obr. 3-3 Rozšírený UML diagram vlastností: Systém na spracovanie objednávok [10]

Ďalšie vlastnosti ako popis, čas viazania, priorita a iné, ktoré boli v zaznamenané ako textové informácie v doplňujúcich vlastnostiach k modelu vlastností môžu byť zapísane ako označené hodnoty konkrétnej vlastnosti.

3.3 Rozšírenie UML o modelovanie vlastností podľa Robaka

Rozšírenie UML o podporu modelovania vlastností podľa Robaka [12], rovnako ako u Claussa predstavuje tzv. ľahké rozšírenie UML, t.j. pomocou stereotypov a označených hodnôt. Povinné vlastnosti sú reprezentované ako UML element komponent, variabilné vlastnosti ako komponent so stereotypom <<variable>> a označenou hodnotou, ktorá pomenúva názov vlastnosti. Samotný diagram vlastností je vyjadrený ako diagram komponentov v UML (Obr. 3-4)

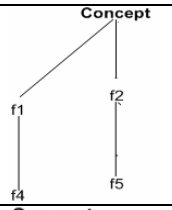
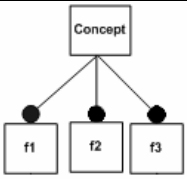
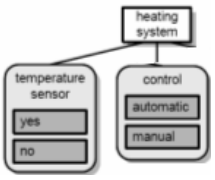
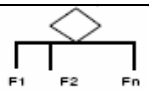
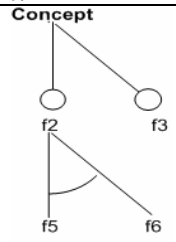
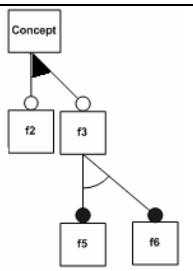
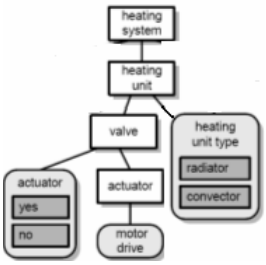
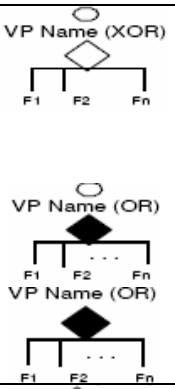
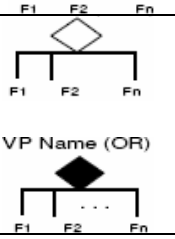


Obr. 3-4 UML komponent diagram - poštový predplatný systém [12]

Do modelu vlastností sú zahrnuté tiež skupiny vlastností, ktoré sú reprezentované ako UML element „package“. Táto skupina vlastností môže byť chápaná ako variačný bod uvedený v notácii Czarneckého-Eiseneckera [9]. Základný typ vzťahu medzi vlastnosťami „pozostáva z“ je vyjadrený pomocou asociácie, pričom vyjadrenie ostatných vzťahov je riešená pomocou poznámky pridruženej k danej asociácii. Podrobnejšia špecifikácia jednotlivých vlastností môže byť realizovaná uvedením ďalších stereotypov, resp. pridaním ďalších označených hodnôt, ktoré by vyjadrovali vlastnosti ako zdôvodnenie, čas viazania, zainteresované osoby a ďalšie.

3.4 Porovnanie analyzovaných notácií

Nasledujúca tabuľka poskytuje stručné porovnanie notácií opísaných v tejto práci. Notácie sú porovnávané vzhľadom na grafickú reprezentáciu povinných a variabilných vlastností, podporu modelovania pomocou variačných bodov a možnosť vyjadriť všeobecnú kardinalitu.

	FODA/ FORM	**Czarnecki/ Riebish**	**Návrhové priestory**	**FeatuRSEB**	**Clauss /Robak**
Povinné vlastnosti					Stereotyp mandatory
Variabil-né vlastnosti					Stereotyp alternative optional
Variačné body	-	Sú definované ale explicitná podpora chýba	Explicitná uvedené nie sú, ale skupina môže byť chápaná ako variačný bod		Stereotyp variation point
Kardinalita	-	Czarnecki – Riebish +	-	-	Kardinalita UML

Obr. 3-5 Porovnanie analyzovaných notácií

Okrem spoločných vlastností jednotlivých notácií vzhľadom, na ktoré ich môžeme porovnávať obsahujú aj prvky špecifické pre danú notáciu. Metóda návrhových priestorov obsahuje mechanizmus tzv. slabého obmedzenia, ktorý má odporúčajúci charakter a umožňuje vytvárať predvolený profil (inštanciu konceptu). Metóda FORM rozdeľuje vlastnosti do viacerých vrstiev, pričom definuje medzi vlastnosťami ďalšie väzby ako generalizácia/špecializácia a implementácia.

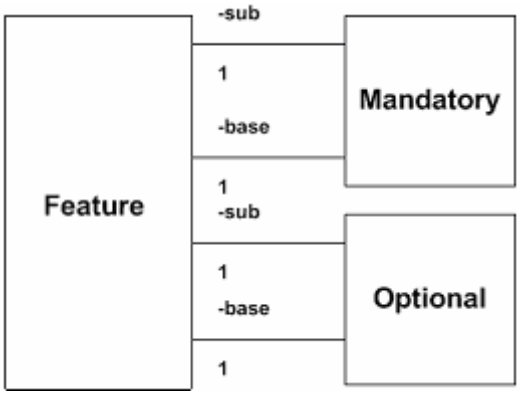
4 Identifikovanie množiny relevantných prvkov diagramu vlastností

Cieľom tejto kapitoly je na základe analýzy notácií modelovania vlastností identifikovať významné prvky diagramu vlastností, ktoré budú neskôr „zobrazené“ v profile jazyka UML a v rozšírení metamodelu jazyka. Množina je vytváraná inkrementálnym pridávaním vybraných prvkov analyzovaných notácií počnúc notáciou metódy FODA a končiac metódou FORM.

4.1 Prvky notácie FODA

Diagram vlastností v notácii FODA predstavuje historicky prvý model, ktorý zachytáva variabilitu modelovania vlastností a definuje tak nasledujúce prvky identifikovanej množiny:

- **feature** - vlastnosť, základný prvok modelu vlastností
- **mandatory** - povinná vlastnosť, vzťah nezávislej vlastnosti v roli base so závislou vlastnosťou v roli sub
- **optional** - voliteľná vlastnosť, vzťah nezávislej vlastnosti v roli base so závislou vlastnosťou v roli sub

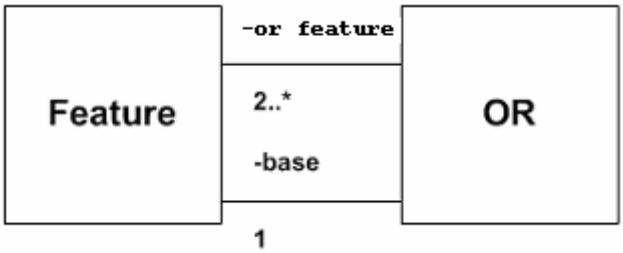


Obr. 4-1 Prvky notácie FODA

4.2 Prvky notácie Czarneckého–Eiseneckera

Notácia modelovania vlastností podľa Czarneckého–Eiseneckera uviedla nový typ vlastnosti alebo-vlastnosť, ktorá tvorí ďalší prvok identifikovanej množiny prvkov vlastností.

- **or** - alebo-vlastnosť predstavuje vzťah medzi viacerými vlastnosťami. Na jednej strane sa nachádza nezávislá vlastnosť v roli base, na druhej množina závislých vlastností v roli or feature s kardinalitou 2..*.

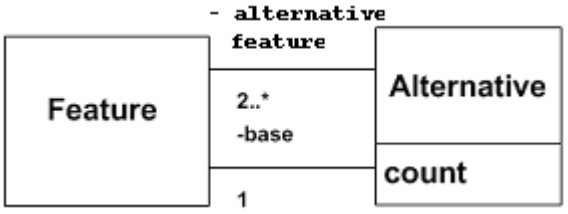


Obr. 4-2 Prvky notácie Czarneckého-Eiseneckera

4.3 Prvky notácie podľa Riebischa

Významným príspevkom tejto notácie bolo zavedenie ľubovoľnej kardinality pre alternatívne vlastnosti definované v metóde FODA, čím došlo k jej priblíženiu potrebám praxe.

- **alternative** – alternatívna vlastnosť predstavuje vzťah medzi nezávislou vlastnosťou v roli base a množinou vlastností v roli alternative feature s kardinalitou. Atribút value určuje počet vlastností v roli alternative feature, ktoré majú byť zahrnuté do konceptu.



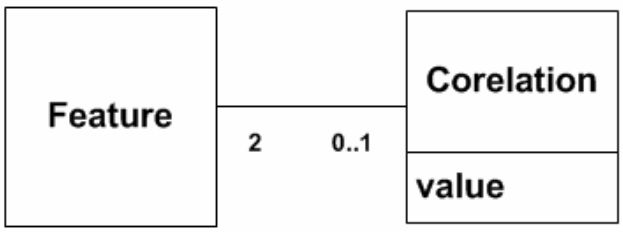
Obr. 4-3 Prvky notácie podľa Riebisha

4.4 Prvky notácie modelovania vlastností pomocou návrhových priestorov

Modelovanie vlastností pomocou návrhových priestorov predstavuje odlišný spôsob modelovania vlastností v porovnaní s predchádzajúcimi notáciami a to jednak v použitej terminológii, ako aj v grafickej reprezentácii modelu. Návrhové priestory definujú pojmy ako dimenzia, kategória, skupina a vzťahy medzi nimi. Väčšina z nich môže byť modelovaná pomocou vzťahov notácie FODA. Návrhové priestory však obsahujú aj prvky, ktoré sa vo FODE nevyskytujú a pritom predstavujú významné príspevky k modelovaniu vlastností. Ide najmä o koreláciu medzi kategóriami a na ňu nadväzujúce vzťahy ako required a mutualy exclusive.

Keďže vzťah required už má svoju obdobu vo vzťahu mandatory a vzťah mutualy exclusive sa dá vyjadriť ako slabá korelácia s hodnotou -1 rozhodol som sa do množiny prvkov modelu vlastností zahrnúť iba vzťah korelácia.

- **corelation** – korelácia, atribút value určuje váhu odporúčania v intervale <-1,1)



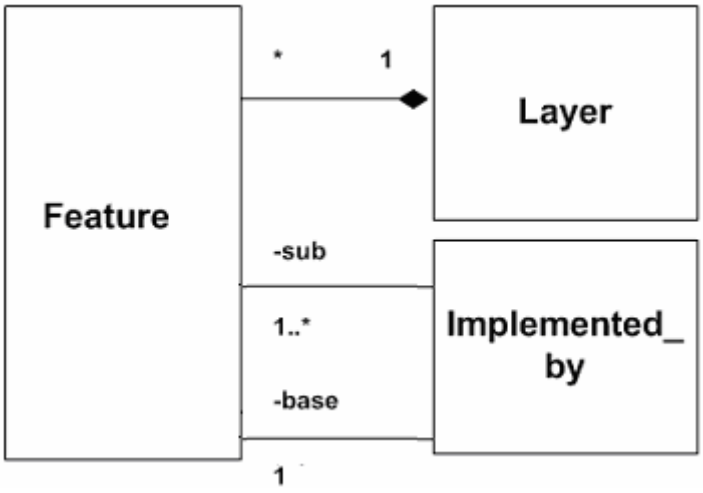
Obr. 4-4 Prvky notácie modelovania vlastností pomocou návrhových priestorov

4.5 Prvky notácie FORM

Metóda FORM rozšírila metódu FODA o podporu návrhu a implementácie. Z tohto dôvodu došlo k rozdeleniu vlastností do príslušných vrstiev, ktoré sú zachytené priamo v diagrame

vlastností. Medzi vlastnosťami bol definovaný vzťah implementácie, ktorý definuje implementáciu vlastnosti v inej vrstve.

- **layer** - vrstva obsahujúca vlastnosti modelu.
- **implemented_by** - vzťah medzi implementovanou vlastnosťou v roli base a implementujúcou vlastnosťou v roli sub

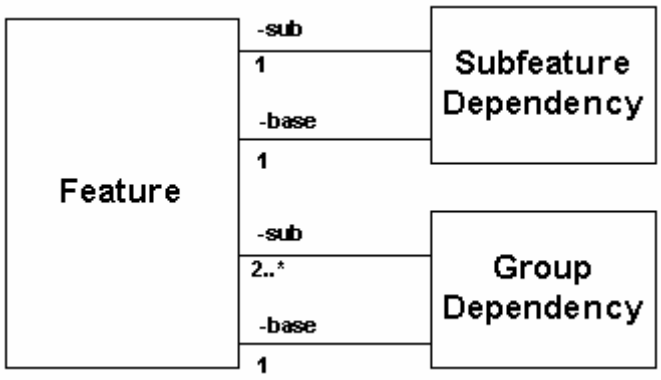


Obr. 4-5 Prvky notácie FORM

4.6 Prvky rozšírenej notácie Czarneckého-Eiseneckera

Rozšírená notácia Czarneckého-Eiseneckera zovšeobecňuje vzťahy medzi vlastnosťami pomocou ich kardinalít. Povinné a voliteľné vlastnosti sú identifikované na základe násobnosti vzťahu medzi dvoma vlastnosťami. Obdobným spôsobom sú zovšeobecnené alternatívne a alebo-vlastnosti. V rozšírenej notácii sú reprezentované ako skupina vlastností, ktorej kardinalita určuje typ vzťahu.

- **subfeature dependency** – vzťah závislosti medzi dvoma vlastnosťami. Jeho špecializáciou je sú povinné a voliteľné vlastnosti.
- **group dependency** – vzťah závislosti medzi vlastnosťou v roli base a skupinou vlastností. Jeho špecializáciou sú alternatívne a alebo-vlastnosti.



Obr. 4-6 Prvky rozšírenej notácie Czarneckého Eiseneckera

5 Integrácia modelovania vlastností na základe rozšírenia metamodelu UML

Jazyk UML je používaný v širokom spektre aplikačných domén. Preto rôzny používatelia majú špecifické potreby, ktoré môžu byť integrované do UML dvoma spôsobmi:

- pomocou profilov jazyka UML (tzv. ľahké rozšírenie definujúce dialekt jazyka UML)
- pomocou rozšírenia metamodelu UML (tzv. významné rozšírenie definujúce člena rodiny jazykov UML)

Obsahom tejto kapitoly je predstavenie metamodelu jazyka UML 2.0, jeho opisu pomocou semiformálnych techník a uvedenie rozšírenia metamodelu UML podporujúceho modelovanie vlastností. Metamodel UML je rozšírený o množinu prvkov modelu vlastností identifikovaných v kapitole 4.

5.1 Metamodel UML

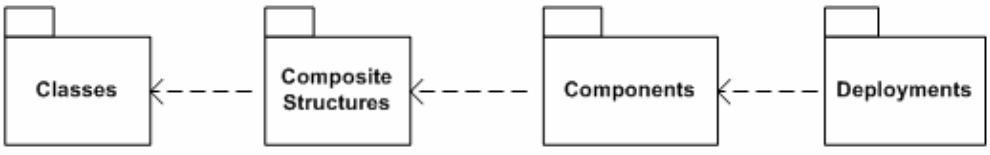
Jazyk UML [8] je rovnako ako ostatné špecifikácie združenia OMG založený na štvorvrstvovej architektúre, ktorá je všeobecne uznávaným konceptom metamodelovania. Jednotlivé vrstvy a ich funkcie sú nasledovné:

Vrstva	**Oblasť**	**Definuje**	**Príklad**
meta meta model	všeobecná infraštruktúra pre metamodelovanie	jazyk pre špecifikovanie meta modelu	*MetaClass*, *MetaAttribute*, *MetaOperation*
meta model	špecifikácia UML	jazyk na definovanie modelu	`Trieda, Atribút, Operácia`
model	projekt používajúci UML	jazyk na opis informácií o doméne	`Person, getName()`
user object data	runtime systém	Špecifické informácie o doméne	[Person: name = "Szabi", height = 184], "hello"

Obr. 5-1 Vrstvy modelu UML

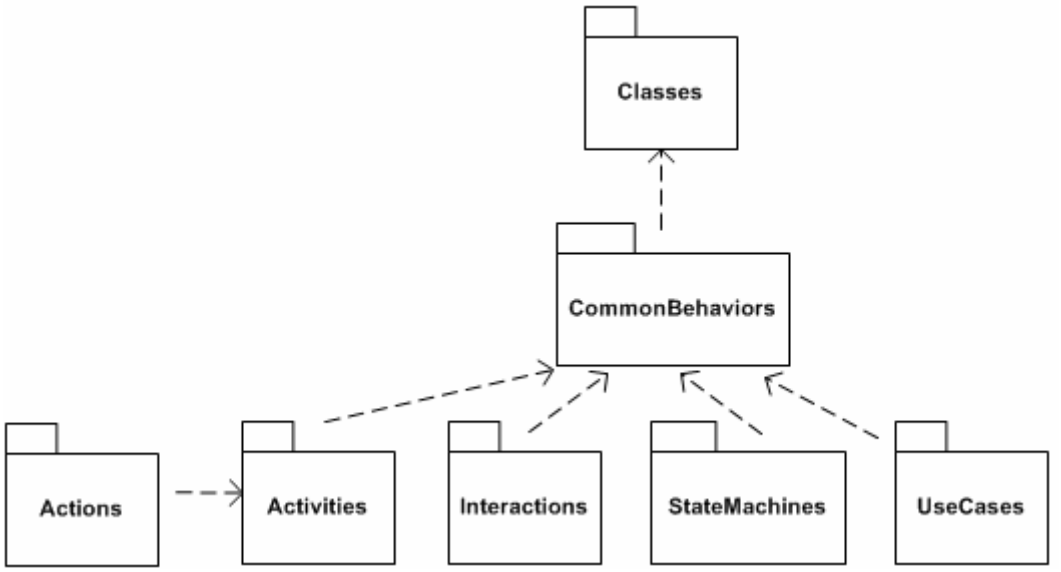
Ak je našou snahou rozšíriť model UML o prvky modelu vlastností je potrebné zamerať sa práve na metamodel, ktorý definuje skupinu prvkov, z ktorých sa skladá model UML. Koncepty jazyka UML sú zoskupené do troch hlavných častí:

Časť I: Koncepty spojené s modelovaním štruktúry



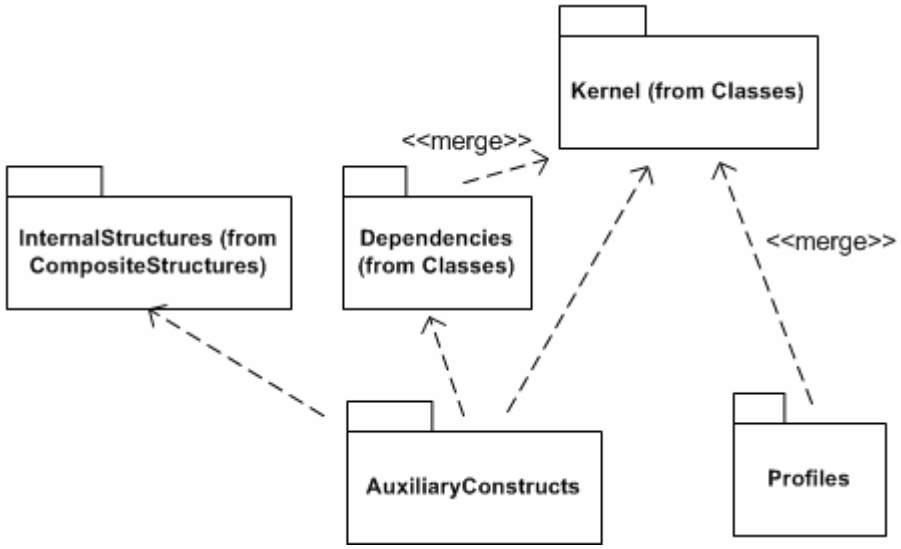
Obr. 5-2 UML balíčky podporujúce štrukturálne modelovanie

Časť II: Koncepty spojené s modelovaním správania



Obr. 5-3 UML balíčky podporujúce modelovanie správania

Časť III: Doplnkové koncepty



Obr. 5-4 UML balíčky podporujúce doplnkové konštruktory

Architektúra metamodelu spĺňa nasledujúce návrhové princípy:

- **Modularita** - princípy slabého viazania a silnej súdržnosti jednotlivých prvkov v balíčkoch
- **Vrstvenie** - rozdelenie metamodelu do viacerých vrstiev
- **Rozdeľovanie** – rozdeľovanie konceptuálnych oblastí v jednej vrstve pre ich lepšiu organizáciu.
- **Znovupoužiteľnosť** – vhodne štruktúrovaný a flexibilný metamodel
- **Rozširovateľnosť**

Z pohľadu tejto práce je najdôležitejšia práve posledná vlastnosť. Ako už bolo uvedené UML je možné rozšíriť pomocou profilov, alebo priamym rozšírením metamodelu. V druhom prípade ide vlastne o špecifikovanie nového jazyka pomocou znovupoužitia infraštruktúry jazyka UML a jej obohatením/ochudobnením o metatriedy, metavzťahy a obmedzenia.

Rozšírenie UML pomocou metamodelu je vhodné v prípadoch keď:

- je potrebná modifikácia samotného metamodelu
- doména je ustálená a obsahuje dobre definovanú množinu prvkov
- model domény nie je prevádzaný do iných domén
- neexistuje potreba kombinácie s inými doménami

Nevýhody tohto prístupu sú nasledovné:

- časová náročnosť vytvorenia špecifikácie rozšírenia
- absencia podpory rozšíreného metamodelu existujúcimi modelovacích nástrojmi bez ich rozšírenia
- strata možnosti kombinácie a prevádzania modelu do iných domén

5.2 Opis metamodelu UML

Metamodel UML je popísaný pomocou semiformálnych techník, ktoré zvyšujú jeho presnosť a zároveň udržiavajú čitateľnosť. Techniky popisujú metamodel z troch pohľadov pomocou kombinácie textu a grafickej notácie. Výhodou tohto prístupu je

- zlepšenie popisu
- redukcia nekonzistencie a dvojznačnosti
- validácia metamodelu pomocou doplnkových techník
- zvýšenie čitateľnosti popisu metamodelu

Syntax metamodelu UML definuje jednotlivé konštrukcie jazyka, spôsob akým sú vytvárané výrazy a to pomocou modelu opísaného podmnožinou UML. Pre každý balíček je syntax definovaná v sekcii *Abstraktná syntax.* Statická sémantika definuje spôsobom akým majú byť jednotlivé prvky jazyka navzájom prepojené a to pomocou OCL (object constraint language) a prirodzeného jazyka. Dynamická sémantika definuje význam správne vytvorených konštrukcií pomocou prirodzeného jazyka avšak môže obsahovať aj dodatočnú notáciu v závislosti od modelu, ktorý opisujeme. Statická sémantika pre jednotlivé balíčky je obsiahnutá v sekcii *Formálne pravidlá*, dynamická v sekcii *Sémantika*.

Ako už bolo povedané metamodel UML je dekomponovaný do jednotlivých balíčkov, ktorých štruktúra je rovnaká a čiastočne bola naznačená v predošlej kapitole. Podrobnejší popis uvádzam v nasledovnom texte a je dôležitý najmä preto, že navrhované rozšírenie metamodelu spočíva v doplnení resp. vytvorení nových balíčkov.

5.2.1 Abstraktná syntax

Táto sekcia obsahuje všeobecný popis prvkov nachádzajúcich sa v príslušnom balíčku. Následne prezentuje jednotlivé prvky diagramom tried jazyka UML ako metatriedy definujúce samotné prvky a vzťahy medzi nimi. Diagram tiež vyjadruje niektoré z formálnych

pravidiel ako napríklad kardinalita vzťahov, či nutnosť usporiadania jednotlivých podprvkov. Na záver je každý prvok opísaný prirodzeným jazykom obsahujúcim jeho atribúty spolu s ich krátkym vysvetlením a zoznam metatried na druhej strane asociácií spojených s príslušným prvkom.

5.2.2 Formálne pravidlá

Statická sémantika metatried UML, (okrem kardinality a usporiadania) je definovaná ako množina ich invariantov, ktoré musia byť splnené. Je to sada obmedzení definovaná nad atribútmi a asociáciami prostredníctvom OCL výrazov a prirodzeného jazyka. Rovnakým spôsobom sú definované v subsekciách dodatočné operácie na metatriedach. V prípade, že všetka statická sémantika je definovaná na predkovi sekcia obsahuje výraz *'No extra well-formed rules'*

5.2.3 Sémantika

Dynamická sémantika prvkov je definovaná pomocou prirodzeného jazyka. Jednotlivé prvky sú zoskupené do logických častí a definované spolu, avšak iba v prípade ak ide o konkrétne metatriedy.

5.2.4 Štandardné prvky

Sekcia obsahuje zoznam stereotypov metatried definovaných v predchádzajúcich sekciách spolu s dodatočným informáciami v prirodzenom jazyku. Ďalej formálne pravidlá pre jednotlivé stereotypy definované podobne ako v sekcii *Formálne pravidlá.* Zvyšné štandardné prvky ako obmedzenia a označené hodnoty sú na tomto mieste vypísané avšak definované v prílohe.

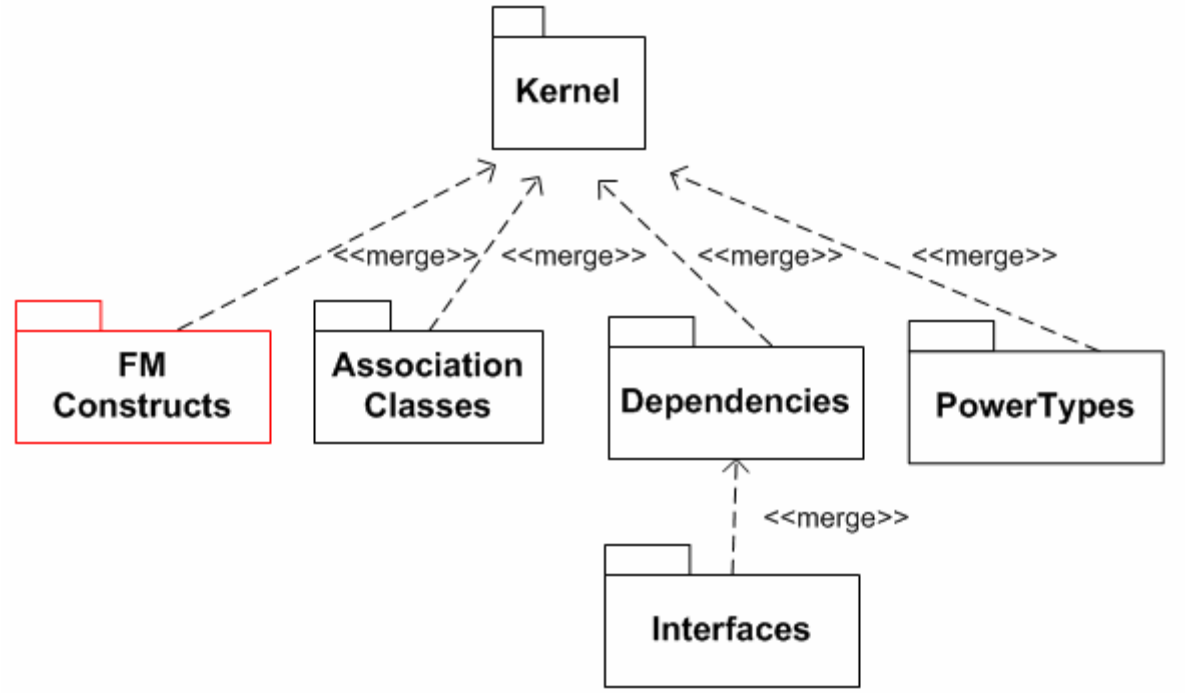
5.2.5 Poznámky

Obsahom tejto sekcie sú zdôvodnenia metamodelovacích rozhodnutí, účel použitia príslušných prvkov a prípady použitia v prirodzenom jazyku.

5.3 Rozšírenie metamodelu jazyka UML o prvky modelu vlastností

Rozšírenie metamodelu UML je vykonané pridaním nových metatried a metavzťahov reprezentujúcich prvky diagramu vlastností definovaných v kapitole 4. Pri rozširovaní metamodelu bola braná do úvahy implementačná nezávislosť modelu vlastností a preto ako základ rozšírenia boli použité metatriedy bez preddefinovanej sémantiky, resp. so sémantikou zodpovedajúcej modelu vlastností. Nové prvky boli pridané tak aby neovplyvnili vzťahy v pôvodnom metamodeli UML, t.j. tak aby model vytvorený na základe pôvodného metamodelu bol korektný aj v prípade použitia rozšíreného metamodelu. Rozšírenie je opísané pomocou abstraktnej syntaxe, sémantiky v prirodzenom jazyku a stereotypov štandardných prvkov.

Pridané prvky metamodelu sú združené v balíčku FM Constructs, ktorý rozširuje Kernel, základný balíček metamodelu UML (Obr. 5-5 Balíček Kernel). Kernel reprezentuje základné modelovacie koncepty jazyka vrátane tried, asociácií a ďalších balíčkov.



Obr. 5-5 Balíček Kernel

FM Element

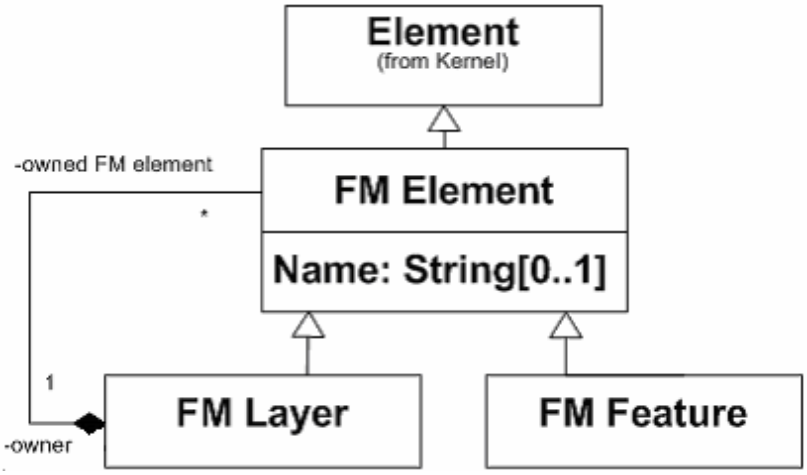
Základom balíčka FM Constructs je abstraktná metatrieda FM Element, ktorá predstavuje prvok modelu vlastností. Jednou z hlavných požiadaviek na integráciu modelu vlastností je zachovanie implementačnej nezávislosti jeho prvkov. V prípade jazyka UML ide najmä o nezávislosť od objektovo orientovanej paradigmy. FM Element preto vznikol rozšírením základného prvku metamodelu UML, prvku Element.

Concept

Abstraktná metatrieda FM Element obsahujúca jediný atribút meno prvku je ďalej špecializovaná jednotlivými prvkami modelu vlastností. Hlavným prvkom modelu je samotná *vlastnosť*, na ktorú sa viažu všetky prvky modelu vlastností. Vlastnosť je reprezentovaná ako metatrieda FM Feature s preddefinovaný stereotypom <<concept>>, ktorý označuje koncept.

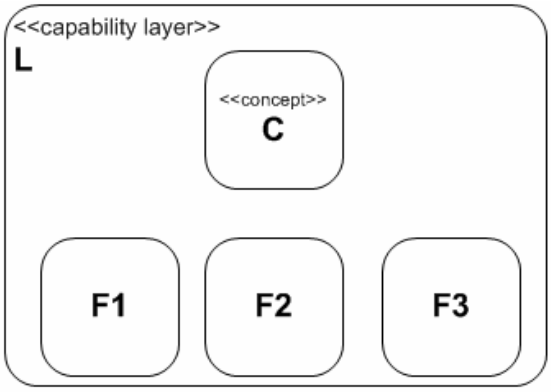
Layer

Vrstva modelu vlastností zastúpená metatriedou FM Layer umožňuje rozdelenie prvkov vlastností do rôznych skupín (vrstiev). Metóda FORM, ktorá existenciu vrstiev do modelu vlastností zaviedla, definuje štyri typy vrstiev, ktoré môžu byť do metamodelu zavedené ako stereotypy prvku FM Layer. Jedná sa o vrstvu schopností (<<capability layer>>), vrstvu prevádzkového prostredia (<<operation environment layer>>), vrstvu doménových technológií (<<domain technology layer>>) a vrstvu implementačných techník (<<implementation technique layer>>).



Obr. 5-6 Základný diagram balíčka FM Constructs

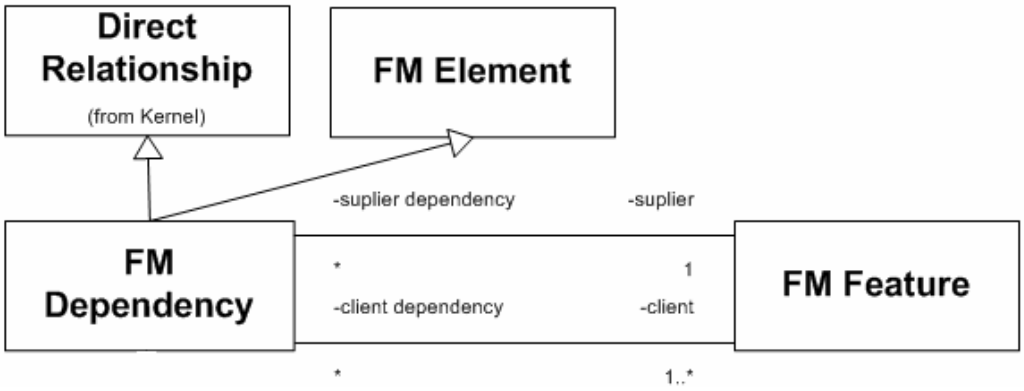
Graficky je vlastnosť vyjadrená ako zaoblený štvorec, vrstvy sú v modely vlastností reprezentované ako zaoblené obdĺžniky (Obr. 5-7)



Obr. 5-7 Koncept, vlastnosť, vrstva rozšíreného metamodelu UML

FM Dependency

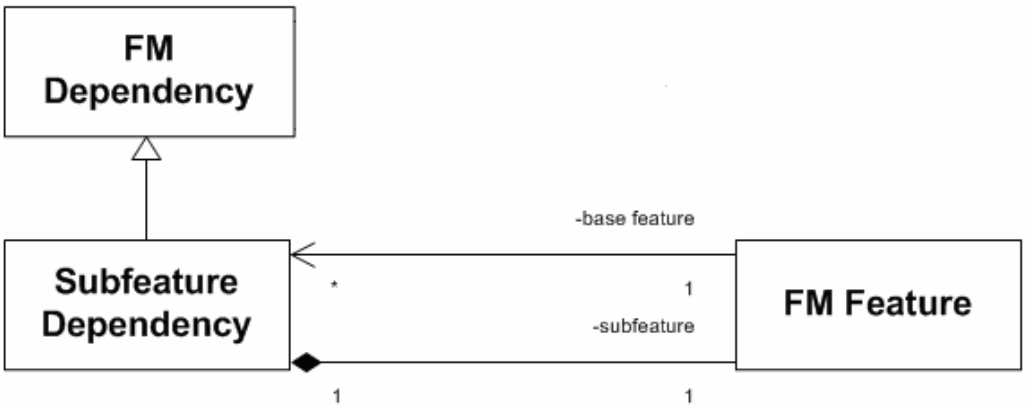
Základným prvkom vyjadrujúcim vzťahy medzi vlastnosťami je metavzťah FM Dependency. FM Dependency predstavuje vzťah supplier/client, v ktorom po zahrnutí nezávislej vlastnosti v roli supplier do konceptu je možné zahrnúť do konceptu aj závislé vlastnosti v roli client. Presné podmienky zahrnutia vlastností v roli client definujú jednotlivé špecializácie metavzťahu. Prvok FM Dependency rozširuje metavzťah Direct Relationship z balíčka Kernel a samozrejme základný prvok modelu vlastností FM Element. (Obr. 5-8)



Obr. 5-8 Abstraktné vzťah rozšíreného metamodelu UML

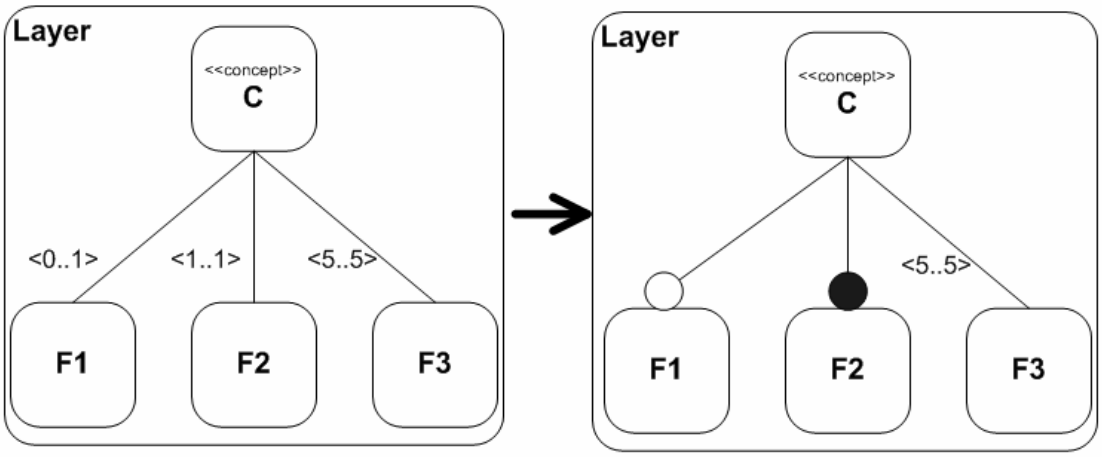
Subfeature Dependency

Vzťahy medzi dvoma vlastnosťami definovanými v metóde FODA (povinné vlastnosti, alternatívne vlastnosti) boli neskôr upravené po grafickej stránke notáciou Czarneckého-Eiseneckera a následne v rozšírenej notácii Czarneckého-Eiseneckera zovšeobecnené pomocou kardinality ich vzťahu. Tento vzťah závislosti dvoch vlastností je v metamodeli reprezentovaný metavzťahom Subfeature Dependency (Obr. 5-9).



Obr. 5-9 Vlastnosť Subfeature Dependency rozšíreného metamodelu UML

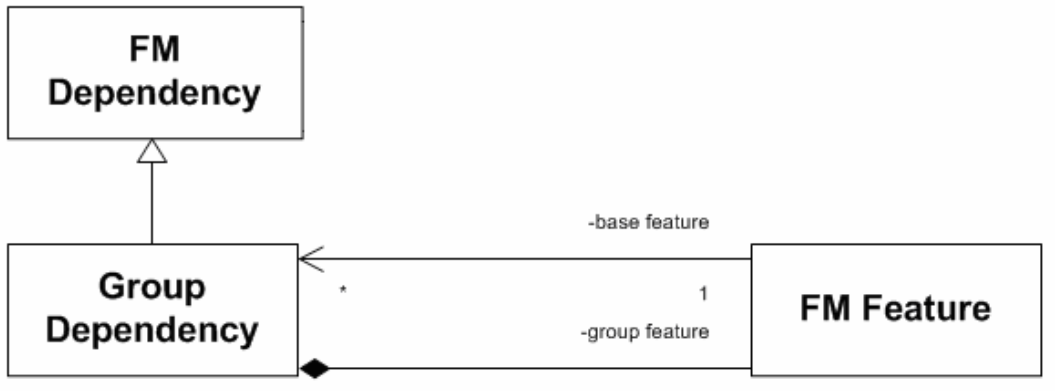
Pôvodné grafické notácia nezovšeobecných vzťahov môžu byť vyjadrené ako stereotypy s rozdielnou grafickou notáciou vzťahu.



Obr. 5-10 Stereotypy vlastnosti Subfeature Dependency

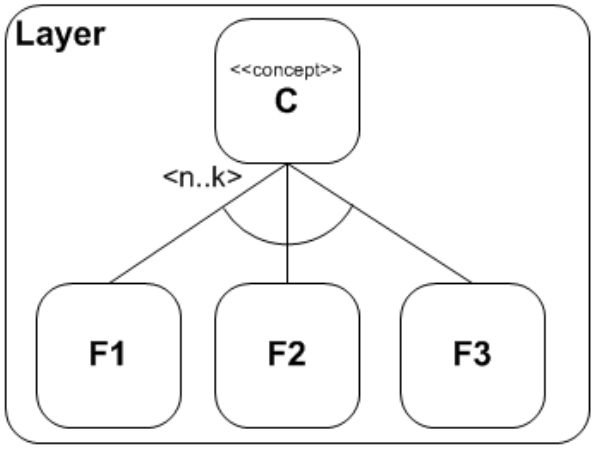
Group Dependency

Vzťah medzi vlastnosťou a skupinou vlastností bol prvý krát uvedený v metóde FODA, išlo o tzv. alternatívne vlastnosti. Neskôr bol Czarneckým-Eiseneckerom definovaný nový typ skupinového vzťahu tzv. alebo-vlastnosti. Čiastočné zovšeobecnenie týchto vzťahov na úrovní ich kardinalít uvedené Riebishom bolo neskôr nahradené v rozšírenej notácii Czarneckého-Eiseneckera jediným vzťahom, ktorého kardinalita určuje sémantiku vzťahu. Uvedený vzťah je v rozšírenom metamodeli jazyka UML reprezentovaný prvkom Group Dependency. Prvok predstavuje špecializáciu všeobecného vzťahu závislosti medzi vlastnosťami FM Dependency, v ktorom sa vlastnosti môžu vyskytovať v roli nezávislej základnej vlastnosti (base feature), alebo v roli závislej skupinovej vlastnosti (group feature).



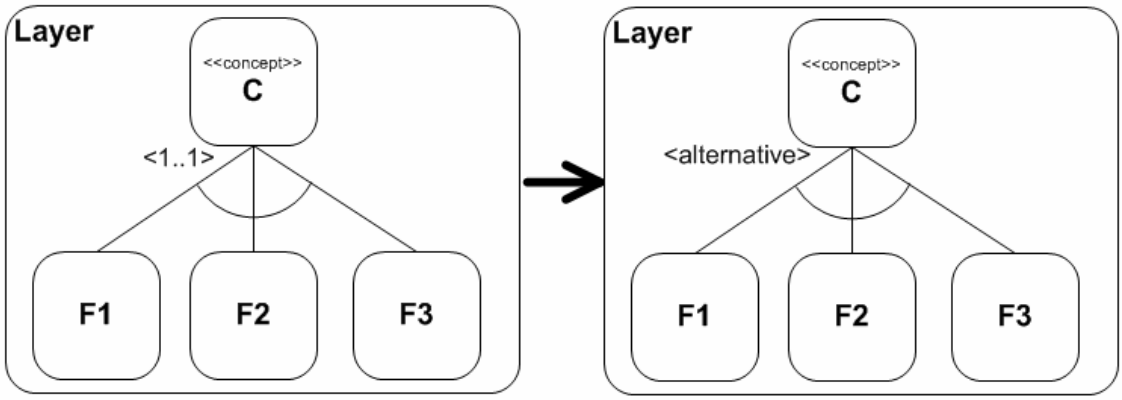
Obr. 5-11 Vzťah Group Dependency rozšíreného metamodelu UML

Grafická notácia vzťahu vychádza z notácie alternatívnych vlastností metódy FODA. Skupina vlastností je označená prázdnym oblúkom spolu s príslušnou kardinalitou vzťahu určujúcou sémantiku vzťahu.

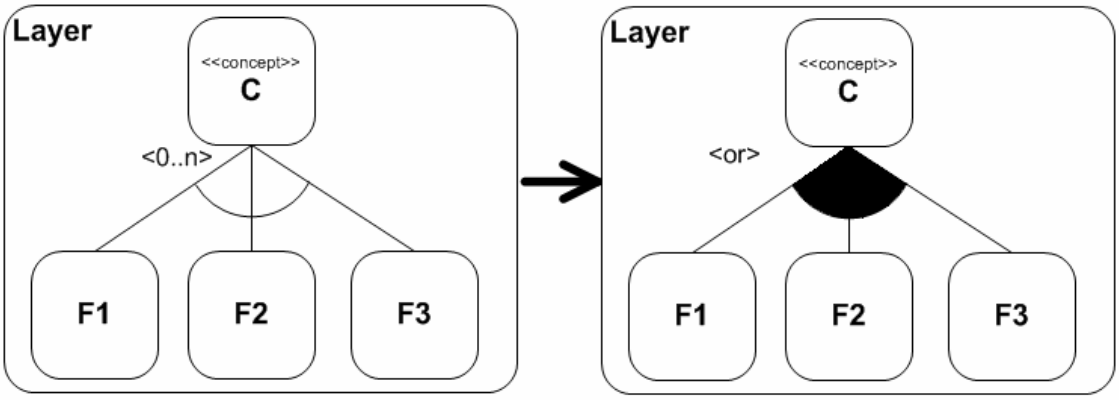


Obr. 5-12 Príklad modelovania vzťahu Group Dependency

Tak ako v prípade povinných a voliteľných vlastností aj alternatívne a alebo-vlastnosti môžu byť vyjadrené ako stereotypy zovšeobecneného vzťahu s grafickou notáciou odpovedajúcou metóde FODA, resp. základnej notácii Czarneckého-Eiseneckera. Príklady zobrazenia týchto špeciálnych vzťahov sú na Obr. 5-13 a Obr. 5-14



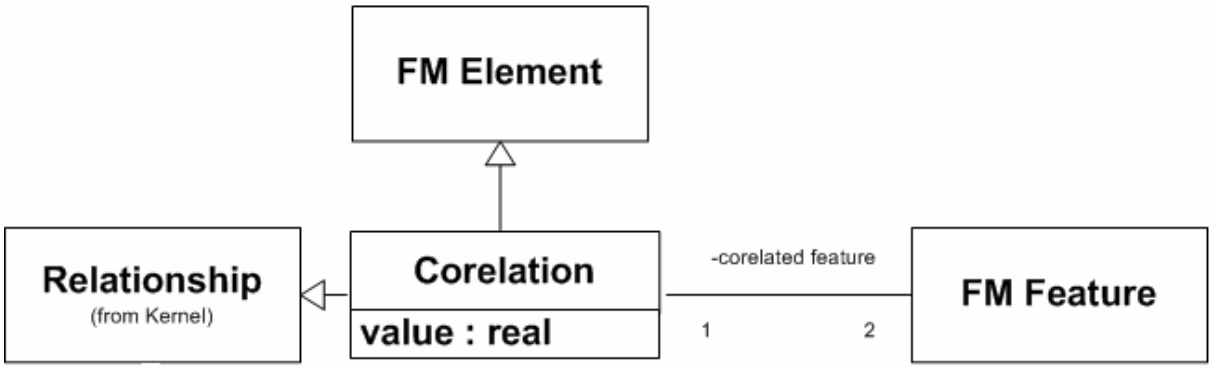
Obr. 5-13 Stereotyp alternative vzťahu group dependency



Obr. 5-14 Stereotyp or vzťahu group dependency

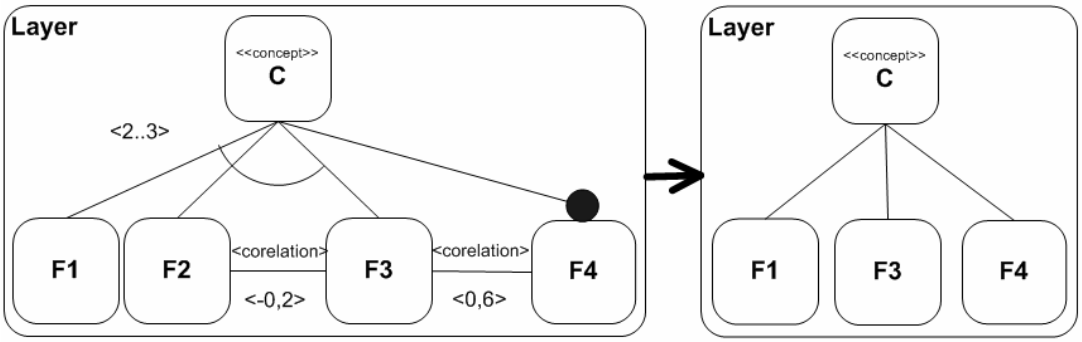
Corelation

Korelácia medzi dvoma vlastnosťami predstavuje významný príspevok metódy návrhových priestorov modelovaniu vlastností. Korelácia reprezentuje vzťah medzi vlastnosťami, ktorý odporúča zahrnutie, resp. nezahrnutie dvoch vlastností spoločne do konceptu. Na rozdiel od predchádzajúcich vzťahov sú vlastnosti vystupujúce vo vzťahu rovnocenné a preto prvok Corelation reprezentujúci vzťah korelácia je odvodený od metavzťahu Relationship a samozrejme tiež od prvku FM Element (Obr. 5-15). Váha odporúčania je daná atribútom value. V metóde návrhových priestorov je táto hodnota z intervalu <-1;1>, kde hraničné hodnoty nazývané tiež ako silná korelácia už nevyjadrujú odporúčanie, ale nutnosť zahrnúť, resp. nezahrnúť obe vlastnosti do konceptu. Keďže vzťah silnej korelácie už má svoju obdobu vo vzťahu SubfeatureDependency je korelácia integrovaná do metamodelu UML ako vzťah s atribútom v intervale <-1;1).



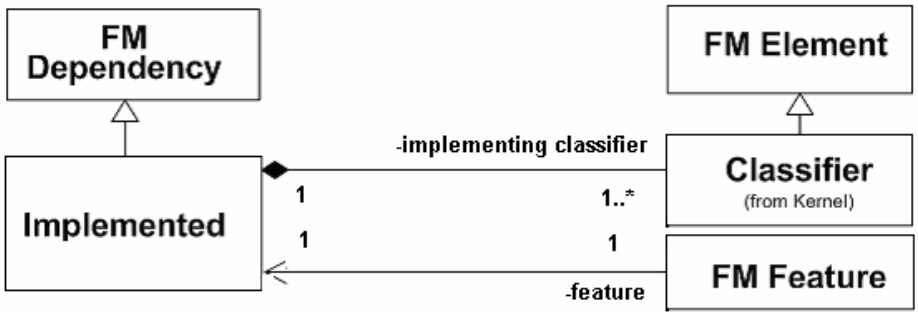
Obr. 5-15 Prvok corelation rozšíreného metamodelu UML

So vzťahom korelácia súvisí možnosť vytvárať profily, preddefinovanú konfiguráciu vlastností, tak ako to je opísané v samotnom modelovaní pomocou návrhových vzorov. Takýto profil umožňuje z modelu vlastností jednoduchým spôsobom určiť, ktoré vlastnosti budú zahrnuté do konceptu. Príklad využitia korelácie je demonštrovaný na (Obr. 5-16). Vlastnosť F4 je povinná a preto musí byť zahrnutá do konceptu. Z vlastností F1, F2 a F3 majú byť do konceptu zahrnuté ľubovoľné dve, alebo všetky tri. Pri rozhodovaní, ktoré vlastnosti zahrnúť je nápomocní práve korelácia, ktorá na základe zahrnutia vlastnosti F4 odporučí zahrnutie vlastnosti F3 a následne vylúči vlastnosť F2.



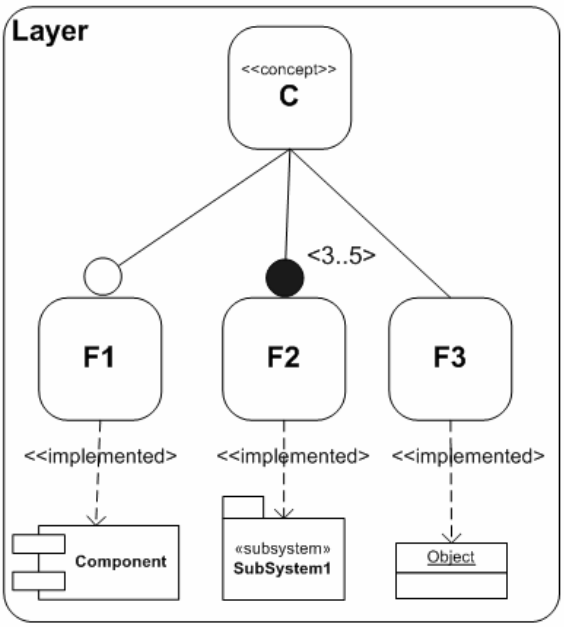
Obr. 5-16 Príklad využitia korelácie

Vzťah implemented uvedený v metóde návrhových vzorov predstavuje prechod od implementačne nezávislých vlastností ku inštanciám, ktoré príslušnú vlastnosť implementujú. Vzťah je odvodený od prvku FM Dependency, pričom závislosť medzi vlastnosťou a inštanciou je riešený rovnako ako u povinných vlastností, t.j. po zahrnutí vlastnosti do konceptu je automaticky zahrnutá aj inštancia. Tá je reprezentovaná prvkom Classifier z balíčku Kernel, ktorý popisuje inštancie s vlastnosťami. Aby bolo možné prvok Classifier a jeho špecializácie zahrnúť do vrstvy modelu vlastností, bol k jeho rodičovským prvkom pridaný základný prvok modelu vlastností FM Element (Obr. 5-17).



Obr. 5-17 Vzťah implemented rozšíreného metamodelu UML

Príklad modelu vlastností spolu s inštanciami, ktoré jednotlivé vlastnosti implementujú je znázornený na Obr. 5-18



Obr. 5-18 Príklad modelu vlastností s implementovanými vlastnosťami

6 Integrácia modelovania vlastností na základe profilu UML

Prvky modelu vlastností môžu byť integrované nielen pomocou rozšírenia metamodelu UML. Ďalším spôsobom je tzv. ľahké rozšírenie prostredníctvom profilov jazyka UML. Obsahom tejto kapitoly je opis mechanizmu rozšírenia pomocou profilov, priblíženie ich výhod/nevýhod a typických situácií kedy je ich použitie vhodné.

6.1 Profil jazyka UML

Profil jazyka UML 2.0 je stereotypovaný balíček obsahujúci stereotypy (stereotypes), označené hodnoty (tagged values) a obmedzenia (constraints), ktoré sú aplikované na základné prvky, atribúty a vzťahy metamodelu. Profil umožňuje prispôsobenie jazyka UML jednotlivým platformám či doménam pomocou:

- definovania nových metatried z existujúceho metamodelu (stereotypy)
- definovania nových metaatribútov (označené hodnoty)
- doplnenia obmedzení na použitie metamodelu a konštruktorov
- doplnenia sémantiky ktorá v metamodeli neexistuje
- doplnenia sémantiky na miestach, ktoré nie sú v metamodeli dostatočne špecifikované
- definovania novej notácie pre už existujúce symboly
- použitia terminológie prispôsobenej určitej platforme alebo oblasti

Prednosti rozšírenia jazyka UML pomocou profilov sú nasledovné:

- možnosť použitia existujúcich modelovacích nástrojov bez ich úpravy
- podpora kombinácie viacerých profilov aplikovaných na jeden model
- podpora výmeny modelu medzi rôznymi profilmi
- podpora dynamickej zmeny aplikácie profilu na model v prípade zmeny perspektívy počas vývoja
- zavedenie profilu je relatívne časovo nenáročné

Nevýhody použitia profilov ako rozširujúceho mechanizmu plynú z jeho definície:

- absencia možnosti meniť existujúci metamodel jazyka UML
- absencia možnosti odstrániť existujúce obmedzenia
- sémantika profilu nemusí presne zodpovedať modelovanej doméne

Zámerom profilov jazyka UML je poskytnúť priamy mechanizmus adaptovania existujúceho metamodelu konštruktorom, ktoré sú špecifické pre určitú doménu. Ich použitie je vhodné v prípade, že existuje veľké množstvo pohľadov a zmien v doméne, je potrebná kombinácia s inými doménami, prípadne vymieňať existujúci model s ďalšími doménami.

6.2 Profil jazyka UML podporujúci modelovanie vlastností

Obsahom tejto kapitoly je špecifikácia profilu podporujúceho modelovanie vlastností na základe identifikovanej množiny prvkov modelu vlastností vedeného v kapitole 4. Jednou z výhod tohto prístupu je podpora existujúcich modelovacích nástrojov, preto je súčasťou

kapitoly vizualizácia profilu v prostredí Enterprise Architect a doplnenie špecifikácie príkladmi vytvorenými v tomto nástroji.

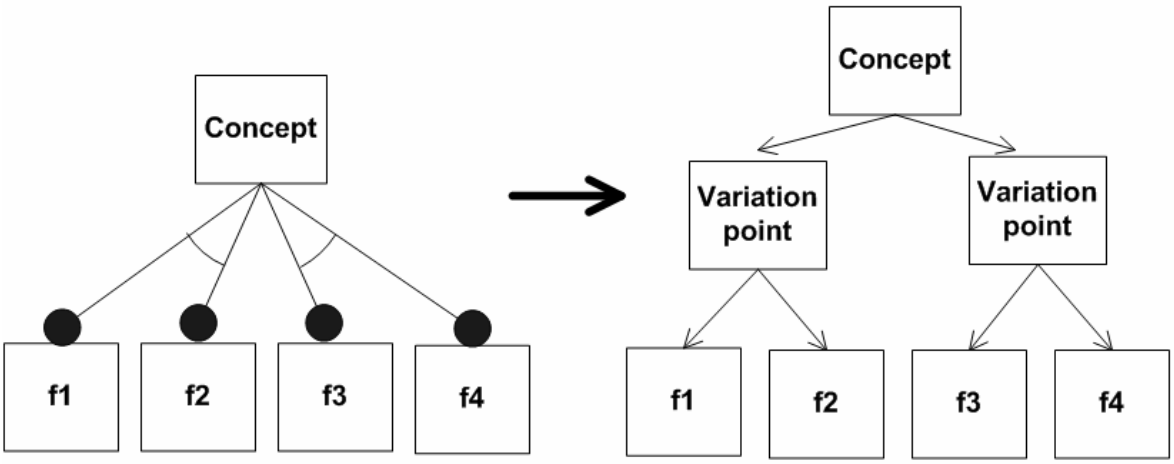
V súčasnosti neexistuje normatívna definícia profilu jazyka UML. Všeobecne môžeme za profil jazyka UML považovať špecifikáciu, ktorá spĺňa nasledovné kritériá [14]:

- identifikuje podmnožinu prvkov metamodelu UML
- špecifikuje štandardné elementy ktoré rozširujú identifikovanú podmnožinu metamodelu UML. Štandardným elementom sa rozumie stereotyp UML, označená hodnota alebo obmedzenie.
- špecifikuje sémantiku vyjadrenú v prirodzenom jazyku obohacujúcu identifikovanú podmnožinu metamodelu UML.
- špecifikuje „dobre formulované pravidlá“ (well-formedness rules) ako opis obmedzení v jazyku OCL, ktoré dopĺňajú už existujúce pravidlá metamodelu UML.

6.2.1 Identifikovaná podmnožina metamodelu UML

Identifikovaná podmnožina metamodelu UML predstavuje skupinu elementov, ktoré sú prostredníctvom stereotypov mapované na vytvorený metamodel diagramu vlastností. Týmto prístupom je zabezpečená plná podpora vytvoreného profilu pre modelovanie vlastností.

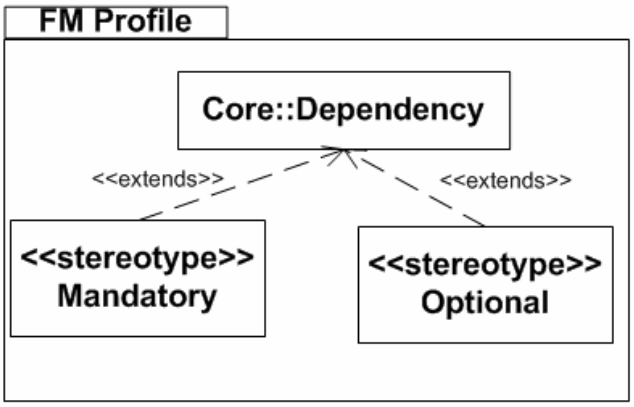
Priame mapovanie všetkých elementov metamodelu diagramu vlastností nie je možné vzhľadom na absenciu n-nárneho vzťahu v modelovacích nástrojoch. Problematické je vyjadrenie vlastností s viacerými skupinami alternatívnych a alebo-vlastností pomocou binárnych vzťahov, kedy nie je možné identifikovať príslušnosť vlastností k jednotlivým skupinám. Riešením je použitie variačných bodov, ktoré reprezentujú skupiny vlastností.



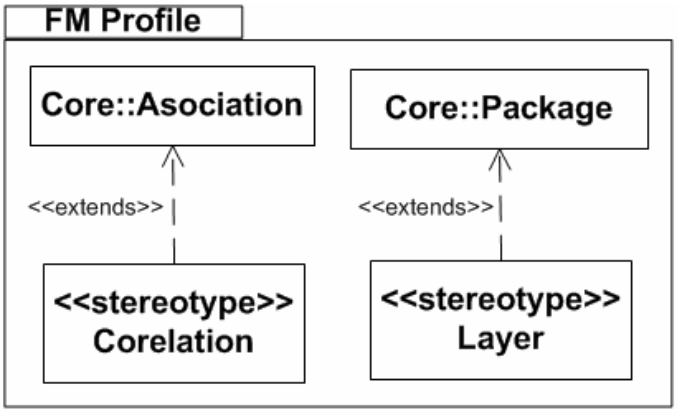
Obr. 6-1 Vlastnosť s viacerými skupinami alebo-vlastností vyjadrená pomocou variačných bodov

Profil podporujúci takto upravené diagramy vlastností rozširuje UML v balíčkoch Core, Components a Deployments. Použité sú metatriedy Artifact, Asociation, Component, Dependency, Realization, Package a implicitne všetky ich rodičovské metatriedy

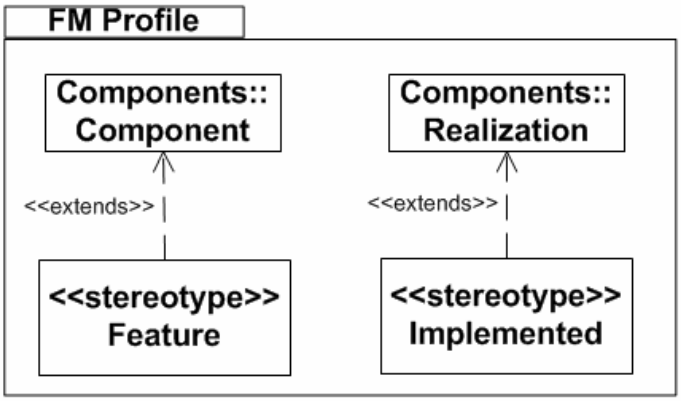
Na opis modelu profilu je použitý diagram tried. Stereotypy sú vyjadrené pomocou prvku Classifier, pričom kľúčové slovo <<stereotype>> zdôrazňuje fakt, že ide o metatriedu stereotypu. Medzi metatriedami UML a stereotypmi je vzťah dependency stereotypovaný ako <<extends>>, vyjadrujúci rozšírenie príslušnej metatriedy. Všetky stereotypy vystupujú v tomto vzťahu v roli client. Konkrétna previazanosť metatried UML a stereotypov profilu modelovania vlastností je znázornená na Obr. 6-2, Obr. 6-3, Obr. 6-4.



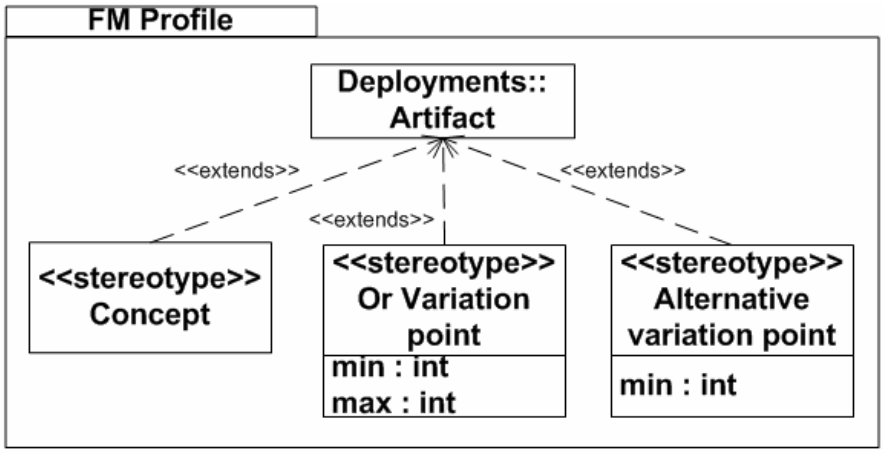
Obr. 6-2 Rozšírenie metatriedy dependency



Obr. 6-3 Rozšírenie metatried asociation a package



Obr. 6-4 Rozšírenie metatried components a realization



Obr. 6-5 Rozšírenie metatriedy artifact

6.2.2 Prvky profilu modelu vlastností

Koncept

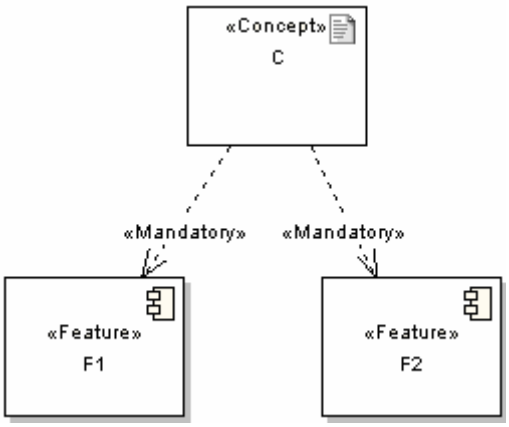
Koncept predstavuje abstraktnú, univerzálnu ideu, alebo entitu, ktorá slúži na označenie kategórií entít, udalostí a vzťahov. Je nositeľom významu, ktorý môže byť vyjadrený viacerými spôsobmi. U konceptu orientovaného na modelovanie vlastností je kladený dôraz predovšetkým na identifikovanie spoločných a rozdielnych vlastností, ktoré používateľ očakáva od aplikácií v určitej doméne, ako aj vzťahov medzi nimi. Koncept je vyjadrený ako stereotyp <<concept>> rozširujúci metatriedu Artifact. Z definície konceptu je zrejmé, že v modeli vlastností sa môže vyskytovať iba raz. Po zahrnutí podpory notácie FORM, konkrétne viacerých vrstiev, toto obmedzenie platí pre každú vrstvu jednotlivo.

Vlastnosť

Vlastnosť je v profile vyjadrená ako stereotyp <<feature>> rozširujúci metatriedu Component. Samotný spôsob implementácie vlastnosti je daný vzťahom <implemented>>, v ktorom okrem komponenty môžu vystupovať aj ostatné elementy v závislosti od špecifikácie konkrétnej vlastnosti. Vzťahy medzi vlastnosťami sú určené pomocou ďalších prvkov profilu.

Povinná vlastnosť

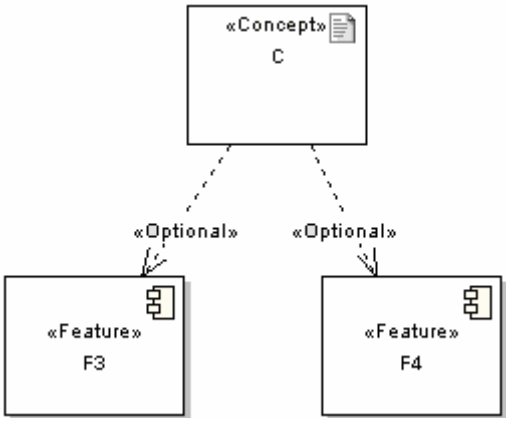
Povinná vlastnosť je reprezentovaná vzťahom mandatory medzi konceptom, vlastnosťami a variačnými bodmi. Element <<concept>> vystupuje vždy v roli nezávislého elementu a je do konceptu zahrnutý implicitne. Element <<feature>> môže byť iba v roli závislého prvku, variačné body v oboch roliach. Vzťah je vyjadrený v profile ako stereotyp <<mandatory>> rozširujúci metatriedu dependency, ktorá vyjadruje všeobecnú závislosť dvoch elementov jazyka UML. Grafické vyjadrenie povinnej vlastnosti v profile je znázornené na Obr. 6-6



Obr. 6-6 Povinná vlastnosť v profile UML

Voliteľná vlastnosť

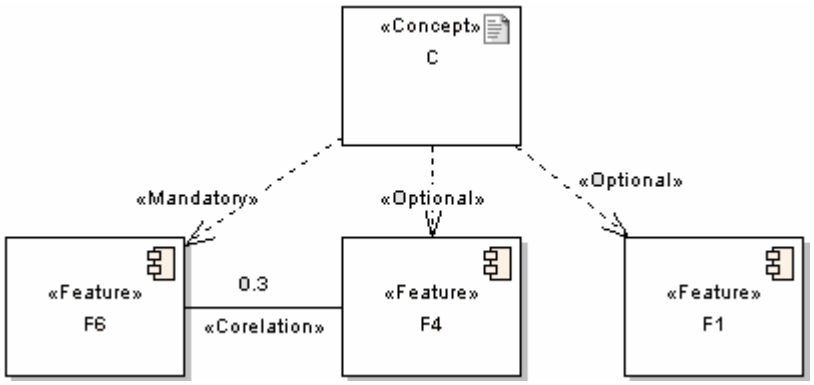
Voliteľná vlastnosť predstavuje závislosť dvoch elementov. Môže existovať medzi konceptom, vlastnosťami a variačnými bodmi. V prípade zahrnutia nezávislého elementu môže byť do konceptu zahrnutý aj závislý element. Voliteľná vlastnosť je reprezentovaná v profile ako stereotyp <<optional>> rozširujúci metatriedu dependency, ktorá vyjadruje všeobecnú závislosť dvoch elementov jazyka UML. Príklad modelovania voliteľnej vlastnosti pomocou profilu je znázornený na Obr. 6-7



Obr. 6-7 Voliteľná vlastnosť v profile UML

Korelácia

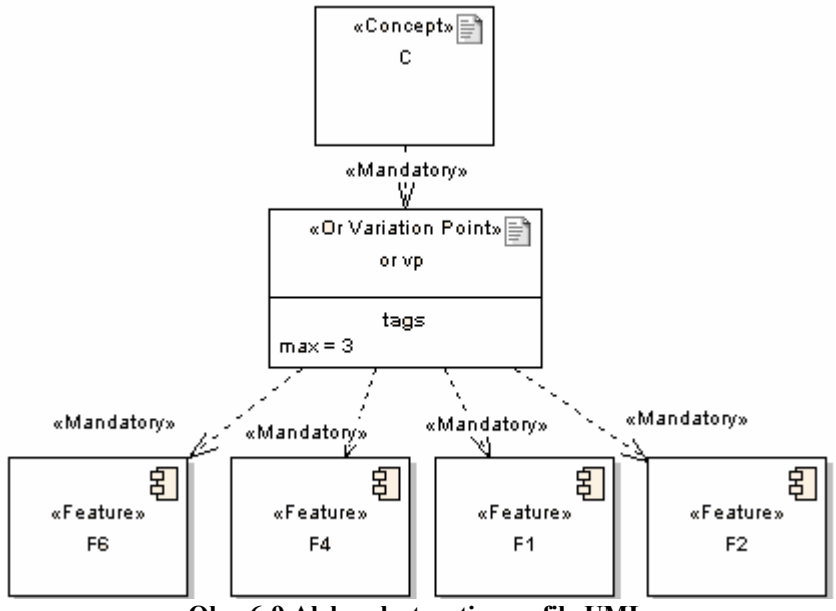
Korelácia ako vzťah vyjadrujúci preferovaný spôsob kombinácie dvoch vlastností je reprezentovaný v profile ako stereotyp <<corelation>> rozširujúci metatriedu asociation, ktorá vyjadruje všeobecný vzťah medzi dvoma elementmi jazyka UML. Váha odporúčania je určená kardinalitou vzťahu v intervale <-1,1). (-1 vlastnosť sa neodporúča zahrnúť, 1- vlastnosť sa odporúča zahrnúť). Pri vytváraní inštancie konceptu korelácie medzi jednotlivými vlastnosťami nie sú záväzné a nemajú väčšiu prioritu ako vzťahy určujúce povinné a voliteľné vlastnosti. Vzťah korelácie medzi vlastnosťami je možné vidieť na Obr. 6-8



Obr. 6-8 Korelácia v profile UML

Alebo vlastnosti

Vzhľadom na absenciu n-nárneho vzťahu v modelovacích nástrojoch nie je možné bez použitia variačných bodov reprezentovať vlastnosť s viacerými skupinami alebo-vlastností. Skupina alebo-vlastností je preto reprezentovaná ako jeden element alebo-variačný bod. Zavedenie alebo variačného bodu, slúži na vyjadrenie alebo-vlastností. V profile je reprezentovaný ako stereotyp <<or variation point>> rozširujúci metatriedu artifact. Alebo-vlastnosť predstavuje vzťah medzi nezávislou vlastnosťou a skupinou závislých vlastností. Zahrnutie nezávislej vlastnosti do konceptu implikuje zahrnutie <0,max> vlastností v skupine, kde max je označená hodnota určujúca maximálny počet vlastností, ktoré je možné zahrnúť do konceptu. Príklad modelu vlastností s použitím alebo-variačného bodu je znázornený na Obr. 6-9

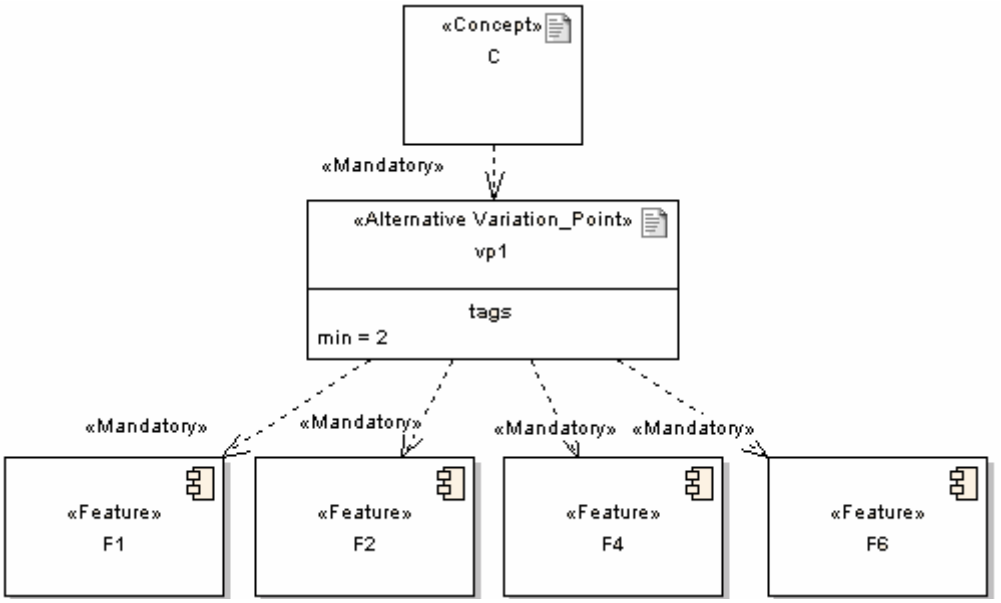


Obr. 6-9 Alebo-vlastnosti v profile UML

Alternatívne vlastnosti

Vzhľadom na absenciu n-nárneho vzťahu v modelovacích nástrojoch, tak ako pri alebo vlastnostiach je skupina alternatívnych vlastností reprezentovaná ako jeden element – alternatívny variačný bod. Alternatívny variačný bod je vyjadrený v profile ako stereotyp <<alternative variation point>> rozširujúci metatriedu artifact. Jeho

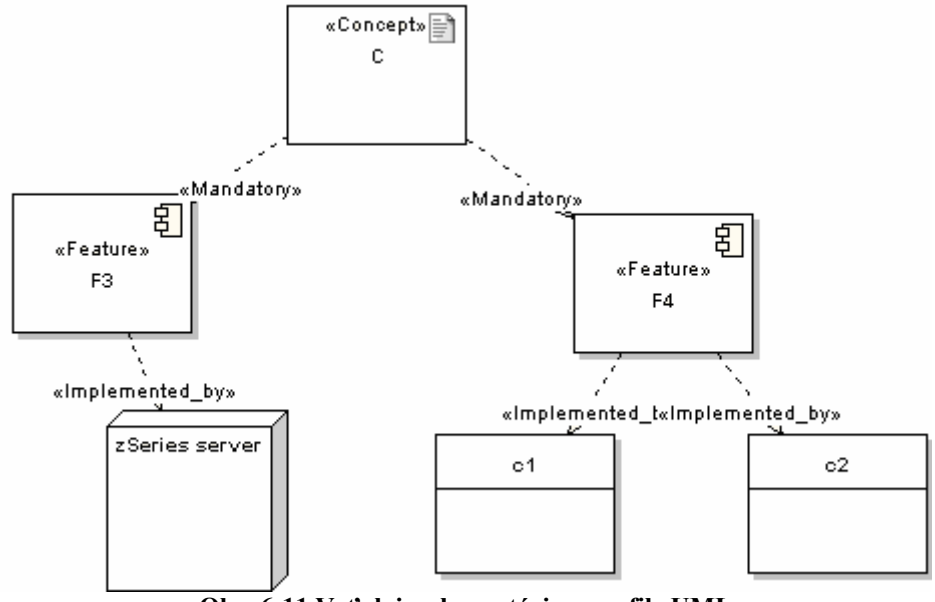
označená hodnota min vyjadruje počet závislých vlastností, ktoré je potrebné zahrnúť do konceptu. Spôsob vyjadrenia alternatívnych vlastností je možné vidieť na Obr. 6-10



Obr. 6-10 Alternatívne vlastnosti v profile UML

Vzťah implementácia

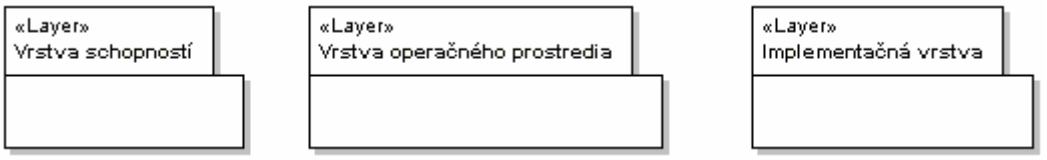
Vzťah implementácia predstavuje vzťah medzi vlastnosťou a elementom (elementmi) UML, prostredníctvom ktorých je táto vlastnosť implementovaná. Vzťah je reprezentovaný v profile ako stereotyp <<implemented_by>> rozširujúci metatriedu realization, ktorá tomuto vzťahu sémanticky najviac zodpovedá. Typ implementujúceho elementu závisí od konkrétnej vlastnosti, napr. funkčné vlastnosti systému sú implementované ako komponenty, resp. triedy. Príklad modelu vlastností obsahujúci vzťah implementácie je znázornený na Obr. 6-11



Obr. 6-11 Vzťah implementácie v profile UML

Vrstva modelu vlastností

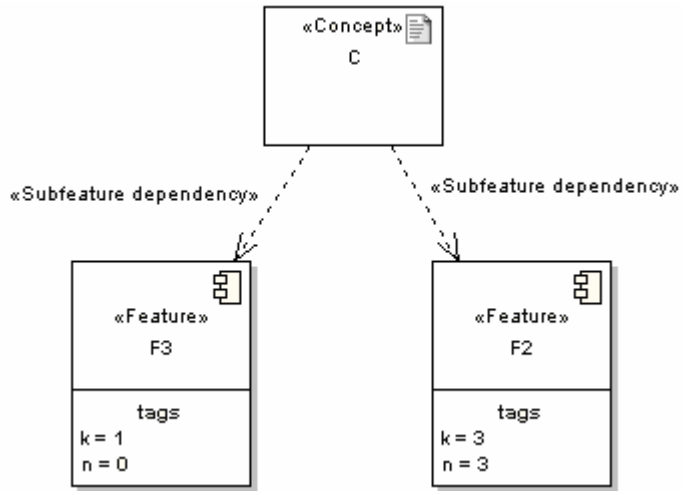
Vrstvy zavedené do modelovania vlastností metódou FORM, je možné modelovať v navrhovanom profile pomocou elementu stereotypu <<layer>>. Ten rozširuje metatriedu package, ktorá umožňuje združovať viaceré elementy do jednej skupiny. Medzi vlastnosti nachádzajúcimi sa v rôznych vrstvách môžu existovať všetky doposiaľ uvedené vzťahy. Grafická reprezentácia vrstiev profilu je znázornená na Obr. 6-12.



Obr. 6-12 Vrstvy v profile UML

Kardinalita vlastností

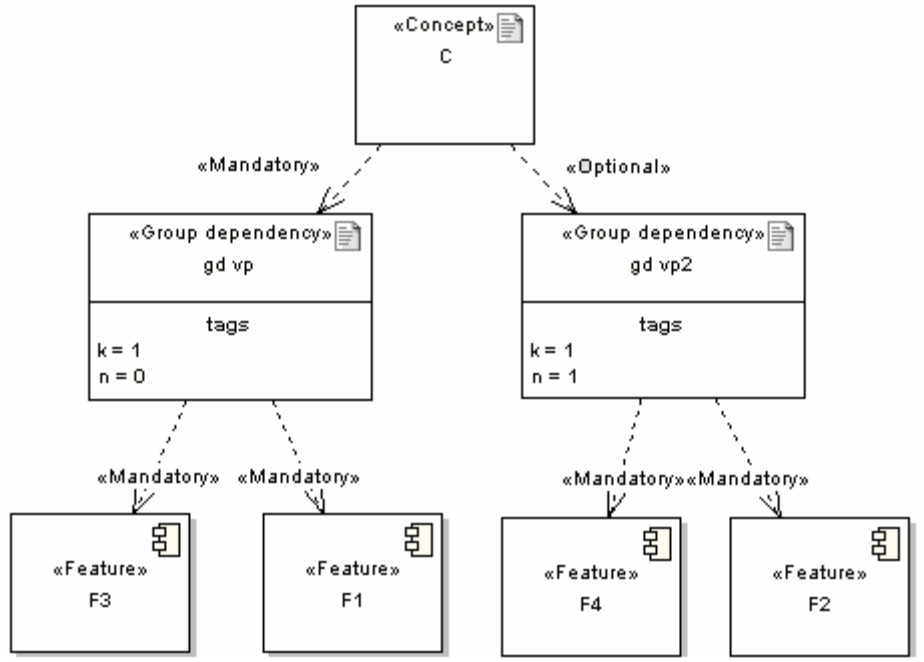
Všeobecná kardinalita vlastností uvedená v rozšírenej notácii Czarneckého-Eiseneckera je reprezentovaná stereotypom <<subfeature dependency>> metatriedy dependency, ktorá vyjadruje všeobecnú závislosť dvoch elementov jazyka UML. Kardinalita vlastností je špecifikovaná pomocou dvoch označených hodnôt n,k, pričom dvojica n=0, k=1 predstavuje voliteľnú vlastnosť a dvojica n=1, k=1 povinnú vlastnosť. Spôsob vyjadrenia všeobecnej kardinality vlastností je možné vidieť na Obr. 6-13



Obr. 6-13 Kardinalita vlastností v profile UML

Kardinalita skupiny vlastností

Vzhľadom na absenciu n-nárneho vzťahu v modelovacích nástrojoch je všeobecný vzťah medzi vlastnosťami v skupine reprezentovaný pomocou stereotypu <<group dependency>>, špecializujúci metatriedu artifact. Všeobecná kardinalita skupiny vlastností uvedená v rozšírenej notácii Czarneckého-Eiseneckera je špecifikovaná pomocou dvoch označených hodnôt n,k, pričom dvojica n=0, k=1 predstavuje voliteľnú vlastnosť a dvojica n=1, k=1 povinnú vlastnosť. Spôsob vyjadrenia všeobecnej kardinality vlastností je možné vidieť na Obr. 6-14



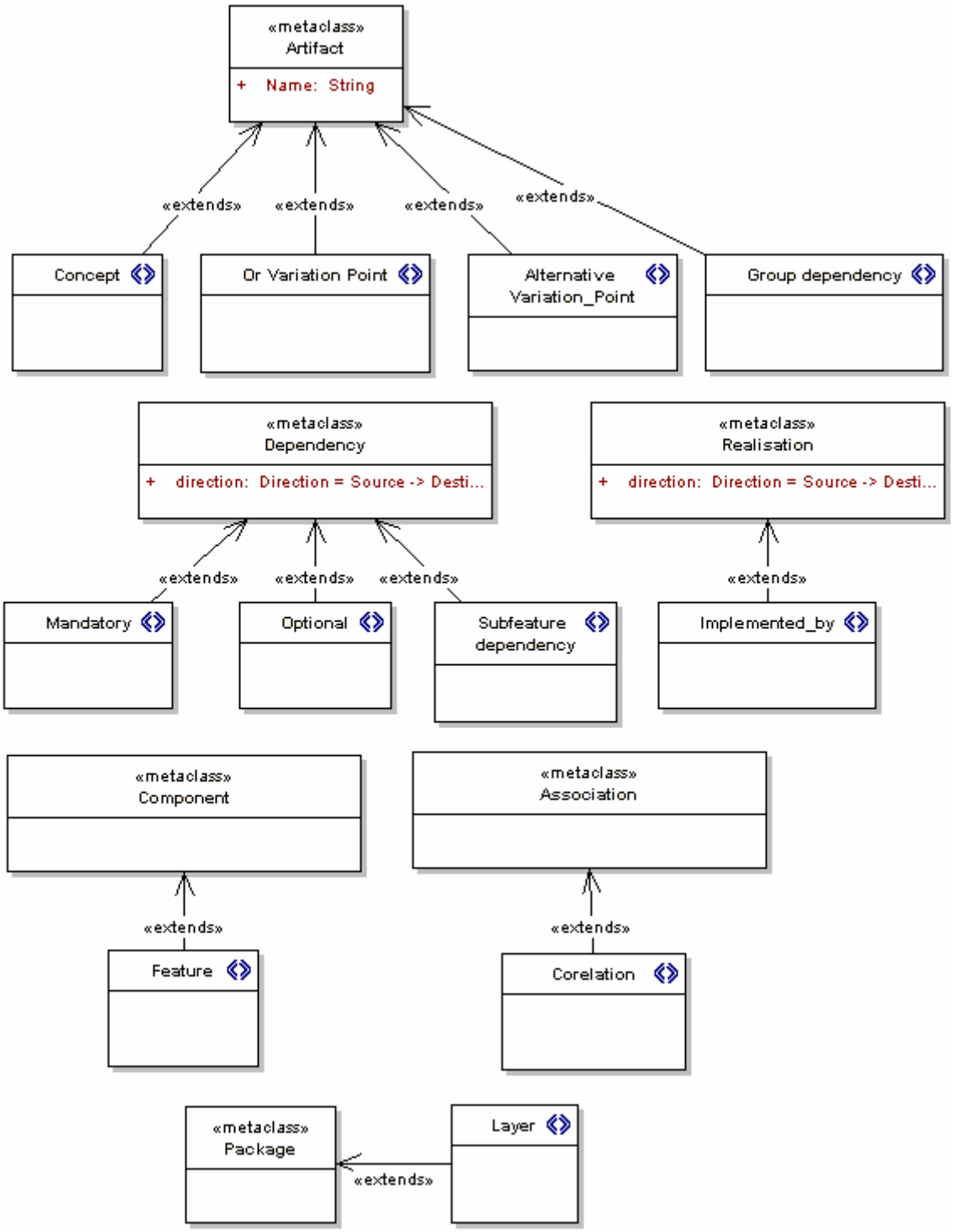
Obr. 6-14 Kardinalita skupiny vlastností v profile UML

6.3 Vizualizácia profilu

Overenie funkčnosti uvedeného profilu bolo realizované jeho implementáciou v modelovacom nástroji Enterprise Architect, ktorý podporuje vytváranie profilov jazyka UML priamo v grafickom prostredí nástroja. Model profilu je znázornený na Obr. 6-15 a jeho elektronická špecifikácia vo formáte XML je súčasťou priloženého CD.

Zavedením vytvoreného profilu do projektu je vytvorená v prostredí záložka obsahujúca všetky definované stereotypy. Tvorba diagramu vlastností je tak z pohľadu používateľa rovnaká ako tvorba ľubovoľného diagramu jazyka UML. Príklady diagramov vlastností vytvorených pomocou uvedeného profilu je možné nájsť časti 7.2, kde boli použité na opis grafickej notácie jednotlivých prvkov.

Keďže diagram profilov je súčasťou jazyka UML, vytvorenie navrhovaného profilu je možné v zásade v ľubovoľnom modelovacom nástroji, ktorý podporuje jazyk UML 2.0. Navrhnutý profil bol implementovaný tiež v plugine Omondo [15] v prostredí Eclipse. Motiváciou na implementáciu profilu prostredníctvom pluginu Omondo bola najmä rozšírenosť bezplatného prostredia Eclipse. V jeho neprospech hovorí relatívne zložité používateľské rozhranie pri tvorbe profilu a jeho následného použitia, ktoré rozhodlo v prospech nástroja Enterprise Architect. Vytvorený profil v prostredí Omondo je súčasťou priloženého CD.



Obr. 6-15 Profil UML podporujúci modelovanie vlastností

7 Zhodnotenie

V rámci diplomovej práce som analyzoval najznámejšie prístupy k modelovaniu vlastností a to najmä z pohľadu notácie modelu vlastností. Identifikoval som spoločné a rozdielne prvky, resp. ich reprezentácie v modeli vlastností Na základe analýzy som definoval množinu vybraných prvkov, ktoré boli v ďalších častiach práce integrované do UML. Identifikovaná množina spĺňa požiadavku podpory rôznych prístupov k modelovaniu vlastností a súčasne obsahuje prvky, ktoré tieto prístupy čo najviac zovšeobecňujú.

Vzhľadom na zachovanie abstraktnosti modelu vlastností bola identifikovaná množina prvkov integrovaná do UML tzv. významným rozšírením na úrovni metamodelu. Základom rozšírenia sú prvky jadra metamodelu UML s vysokou úrovňou abstrakcie, čím bola zabezpečená požiadavka implementačnej nezávislosti modelu vlastností. Nevýhodou uvedeného prístupu je absencia možnosti overiť rozšírený metamodel v niektorom z existujúcich modelovacích nástrojov bez jeho úpravy Identifikovaná množina prvkov modelu vlastností bola preto integrovaná do UML aj tzv. ľahkým rozšírením na základe profilov jazyka UML. Takto vytvorený profil umožňuje modelovať vlastnosti v súčasných modelovacích nástrojoch pomocou existujúcich prvkov, avšak za cenu straty abstraktnosti modelu vlastností.

Najvýznamnejším príspevkom tejto práce je vytvorenie rozšíreného metamodelu UML podporujúceho modelovanie vlastnosti. V porovnaní s existujúcimi prístupmi k integrácii modelovania vlastností založených na ľahkom rozšírení ide o kvalitatívny posun v oblasti zachovania abstrakcie modelu. Ďalším pokračovaním tejto práce môže byť spomínaná modifikácia modelovacích nástrojov, tak aby podporovali rozšírený metamodel.

8 Použitá literatúra

[1] K. Kang, S. Cohen, J. Hess, W. Nowak, and S. Peterson. Feature-Oriented Domain Analysis (FODA) Feasibility Study. Technical Report, CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, November 1990

[2] K. Czarnecki, S. Helsen, U. Eisenecker. Staged Configuration Using Feature Models. Proceedings Of Software Product Line Conference 2004.

[3] Griss, M., Favaro, J., d' Alessandro, M.: Integrating feature modeling with the RSEB. In: Proceedings of the Fifth International Conference on Software Reuse (ICSR), IEEE Computer Society Press (1998) 76{85

[4] Kang, K.C., Kim, S., Lee, J., Kim, K., Shin, E., Huh, M.: FORM: A feature-oriented reuse method with domain-specific reference architectures. Annals of Software Engineering 5 (1998) 143{168

[5] Riebisch, M., et al.: Extending feature diagrams with UML multiplicities. In: Proc. of the 6th Conference on Integrated Design and Process Technology (IDPT 2002), Pasadena, California, USA, Society for Design and Process Science (2002).

[6] Geyer, L.: Feature modeling using design spaces. In: Proc. of the 1st German Product Line Workshop (1. Deutscher Software-Produktlinien Workshop, DSPL-1), Kaiserslautern, Germany, IESE (2000)

[7] Lane, T.G.: *Studying Software Architecture Through Design Spaces and Rules*, Technical Report CMU/SEI-90-TR-18, Carnegie Mellon Univ., 1990. Cited in [6]

[8] OMG Unified Modeling Language Superstructure Specification, June 2004, Version 2.0 http://www.omg.org

[9] Czarnecki, K., Eisenecker, U.W.: Generative Programming: Methods, Tools, and Applications. Addison-Wesley (2000)

[10] Clauss, M.: Modeling variability with UML. In: Proc. of Net.ObjectDays 2001,Young Researchers Workshop on Generative and Component-Based Software Engineering, Erfurt, Germany, tranSIT (2001) 226{230

[11] Valentino Vranić. Reconciling Feature Modeling: A Feature Modeling Metamodel. In Mathias Weske and Peter Liggesmeyer, editors, *Proc. of 5th Annual International Conference on Object-Oriented and Internet-Based Technologies, Concepts, and Applications for a Networked World (Net.ObjectDays 2004)*, Erfurt, Germany, September 2004. Springer

[12] Silva Robak, Bogdan Franczyk, Kamil Politowicz: Extending the UML for modeling variability for system families, Int. J. Appl. Math. Comput. Sci., 2002, Vol.12, No.2, 285–298 Institute of Organization and Management, University of Zielona Góra

[13] OMG Unified Modeling Language Infrastrucure Specification, November 2004, Version 2.0 http://www.omg.org

[14] UML Profile for CORBA Specifiaction, April 2002, http://www.omg.org

[15] Eclipse UML Studio, December 2005, http://www.omondo.com

Príloha A Špecifikácia elementov profilu

Príloha popisuje prvky vytvoreného profilu modelovania vlastností prostredníctvom opisu ich sémantiky, špecifikuje prvky metamodelu UML, ktoré sú stereotypované spolu s preddefinovanými označenými hodnotami a obmedzeniami v prirodzenom jazyku. Súčasťou špecifikácie je taktiež predstavenie notácia jednotlivých prvkov prostredníctvom príkladu modelu vlastností obsahujúceho daný prvok.

Concept

Opis

Koncept predstavuje abstraktnú, univerzálnu ideu, alebo entitu, ktorá slúži na označenie kategórií entít, udalostí a vzťahov. Je nositeľom významu, ktorý môže byť vyjadrený viacerými spôsobmi. U konceptu orientovaného na modelovanie vlastností je kladený dôraz predovšetkým na identifikovanie spoločných a rozdielnych vlastností, ktoré používateľ očakáva od aplikácií v určitej doméne, ako aj vzťahov medzi nimi. Koncept je základným prvkom modelu vlastností a všetky ostatné elementy profilu podporujúceho modelovanie vlastností slúžia na jeho vyjadrenie.

Stereotyp a označené hodnoty

Koncept je vyjadrený ako stereotyp <<concept>> špecializujúci metatriedu Artifact.

Obmedzenia

- Z definície konceptu je zrejmé, že v modeli vlastností sa môže vyskytovať iba raz. Po zahrnutí podpory notácie FORM, konkrétne viacerých vrstiev, toto obmedzenie platí pre každú vrstvu jednotlivo. V jednej vrstve sa nachádza vždy iba jeden element reprezentujúci koncept.
- Koncept môže vystupovať iba vo vzťahoch mandatory a optional

Feature

Opis

Vlastnosť predstavuje významný, alebo zreteľný, používateľom viditeľný aspekt, alebo charakteristika softvérového systému v danej doméne (konceptu). Pod vlastnosťou (feature) rozumieme vlastnosť (property) systému, ktorá je dôležitá pre zákazníka a je použitá na zachytenie podobností, alebo odlišností medzi systémami. Vzťahy medzi vlastnosťami konceptom a variačnými bodmi sú bližšie vysvetlené pri popise jednotlivých vzťahov.

Stereotyp a označené hodnoty

Vlastnosť je v profile vyjadrená ako stereotyp <<feature>> špecializujúci metatriedu Component. Samotný spôsob implementácie vlastnosti je daný

vzťahom <<implemented>>, v ktorom okrem komponenty môžu vystupovať aj ostatné elementy v závislosti od špecifikácie konkrétnej vlastnosti.

Obmedzenia

- Vlastnosť môže vystupovať iba vo vzťahoch mandatory, optional a implemented.

Mandatory

Opis

Vzťah mandatory slúži na reprezentáciu povinnej vlastnosti ako závislosti dvoch elementov. Existuje medzi konceptom, vlastnosťami a variačnými bodmi. V prípade zahrnutia nezávislého elementu musí byť do konceptu zahrnutý aj závislý element. Element <<concept>> vystupuje vždy v roli nezávislého elementu a je do konceptu zahrnutý implicitne. Element <<feature>> môže byť iba v roli závislého elementu, variačné body v oboch rolách.

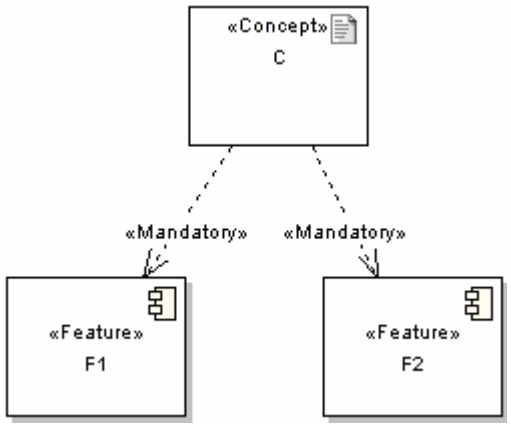
Stereotyp a označené hodnoty

Vzťah je reprezentovaný v profile ako stereotyp <<mandatory>> špecializujúci metatriedu dependency, ktorá vyjadruje všeobecnú závislosť dvoch elementov jazyka UML.

Obmedzenia

- Vzťah môže existovať iba medzi stereotypmi <<concept>>, <<feature>> a variačnými bodmi
- <<concept>> je vždy nezávislým elementom
- <<feature>> je vždy závislým elementom
- Zahrnutie nezávislého elementu do konceptu implikuje zahrnutie závislého elementu

Notácia



Optional

Opis

Vzťah optional reprezentuje voliteľnú vlastnosť ako závislosť dvoch elementov. Existuje medzi konceptom, vlastnosťami a variačnými bodmi. V prípade zahrnutia nezávislého elementu môže byť do konceptu zahrnutý aj závislý element. Element <<concept>> vystupuje vždy v roli nezávislého elementu a je do konceptu zahrnutý implicitne. Element <<feature>> môže byť iba v roli závislého elementu, variačné body v oboch rolách.

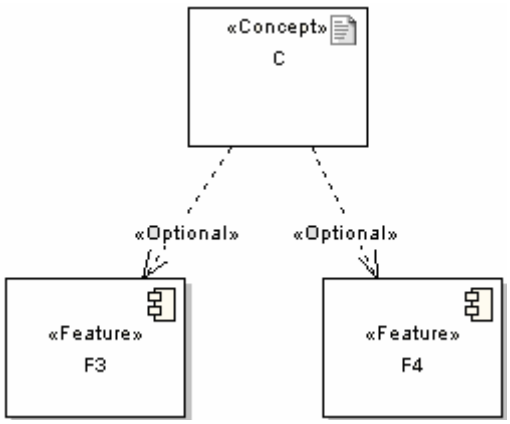
Stereotyp a označené hodnoty

Vzťah je reprezentovaný v profile ako stereotyp <<optional>> rozširujúci metatriedu dependency, ktorá vyjadruje všeobecnú závislosť dvoch elementov jazyka UML.

Obmedzenia

- Vzťah môže existovať iba medzi stereotypmi <<concept>>, <<feature>> a variačnými bodmi
- <<concept>> je vždy nezávislým elementom
- <<feature>> je vždy závislým elementom

Notácia



Corelation

Opis

Vzťah corelation vyjadruje preferovaný spôsob kombinácie dvoch vlastností. Vlastnosti v tomto vzťahu sú rovnocenné a v prípade zahrnutia jednej vlastnosti do konceptu hodnota atribútu korelácie predstavuje odporúčanie na zahrnutie druhej vlastnosti vo vzťahu. Hodnoty atribútu sa nachádzajú v intervale <0,1>. (0 – vlastnosť sa neodporúča zahrnúť, 1- vlastnosť sa odporúča zahrnúť). Pri vytváraní inštancie konceptu korelácie medzi jednotlivými vlastnosťami nie sú záväzné a v žiadnom prípade nemajú väčšiu prioritu ako vzťahy určujúce povinné a voliteľné vlastnosti.

Stereotyp a označené hodnoty

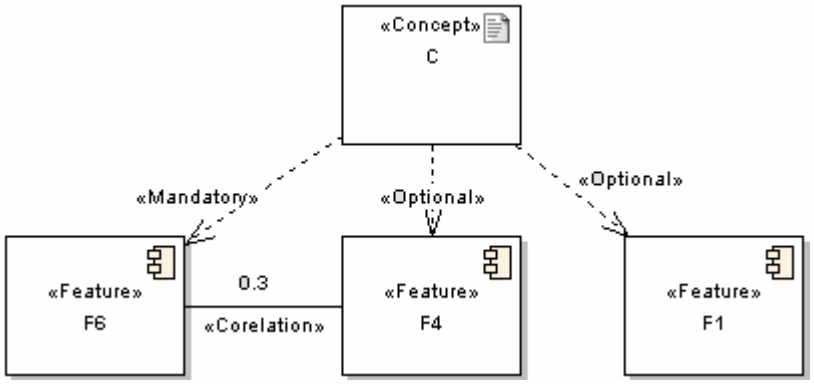
Vzťah je reprezentovaný v profile ako stereotyp <<corelation>> špecializujúci metatriedu asociation, ktorá vyjadruje všeobecný vzťah medzi dvoma elementmi jazyka UML.

Hodnota atribútu reprezentujúca silu korelácie je vyjadrená kardinalitou vzťahu. Ak je kardinalita nešpecifikovaná prednastavená je hodnota je 0

Obmedzenia

- Vzťah existuje iba medzi stereotypmi <<feature>>
- Hodnota atribútu je z intervalu (-1,1>

Notácia



Or variation point

Opis

Zavedenie alebo variačného bodu, slúži na vyjadrenie alebo-vlastností. Alebo-vlastnosť predstavuje vzťah medzi nezávislou vlastnosťou a skupinou závislých vlastností. Zahrnutie nezávislej vlastnosti do konceptu implikuje zahrnutie n závislých vlastností z intervalu <0,max>.

Stereotyp a označené hodnoty

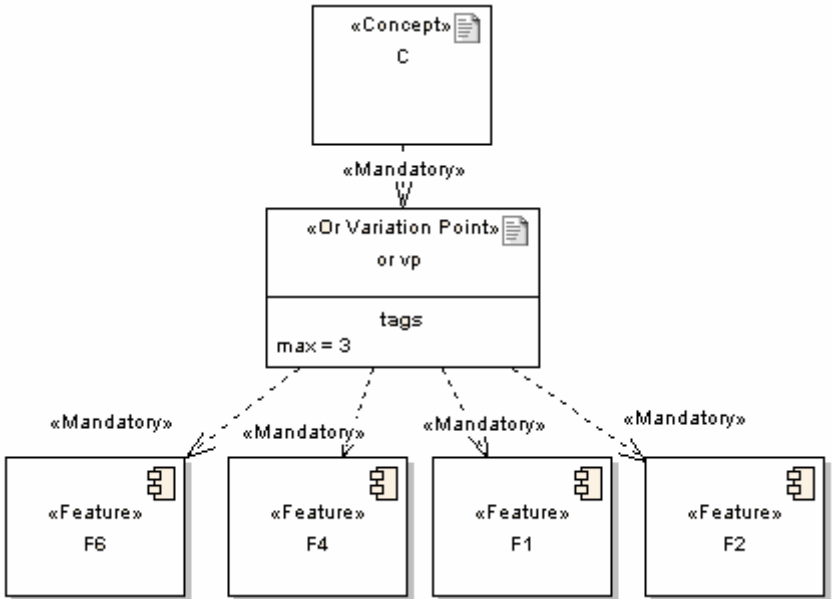
Alebo-variačný bod je reprezentovaný v profile ako stereotyp <<or variation point>> špecializujúci metatriedu artifact.

Stereotyp obsahuje označenú hodnotu max – maximálny počet závislých vlastností, ktoré je možné do konceptu zahrnúť

Obmedzenia

- Alebo-variačný bod môže byť s ostatnými elementmi iba vo vzťahoch mandatory a optional.

Notácia



Alternative variation point

Opis

Zavedenie alternatívneho variačného bodu, slúži na vyjadrenie alternatívnych vlastností. Alternatívna vlastnosť predstavuje vzťah medzi nezávislou vlastnosťou a skupinou závislých vlastností. Zahrnutie nezávislej vlastnosti do konceptu implikuje zahrnutie práve n závislých vlastností.

Stereotyp a označené hodnoty

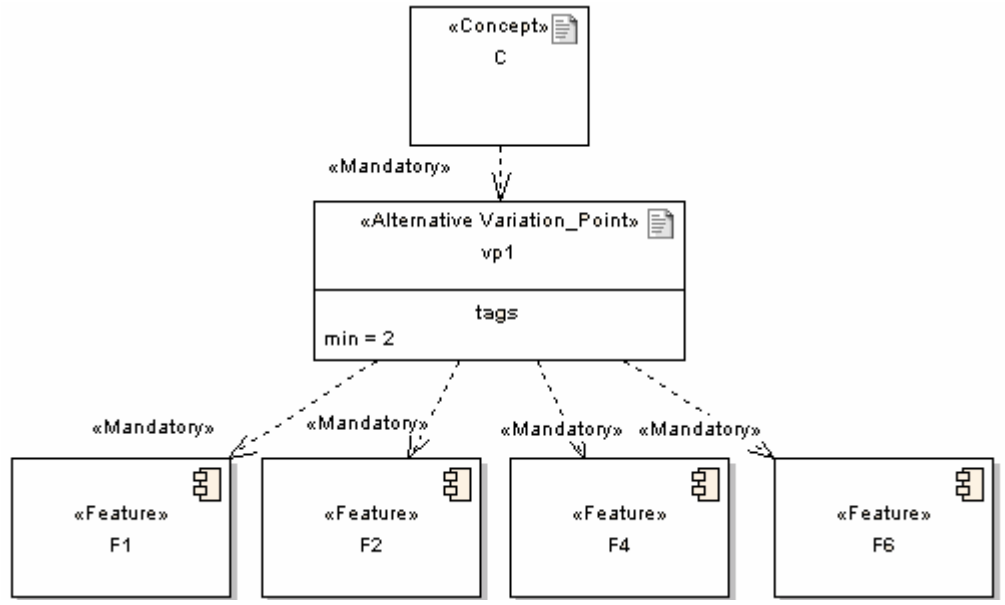
Alternatívny variačný bod je reprezentovaný v profile ako stereotyp <<alternative variation point>> špecializujúcim metatriedu artifact.

Stereotyp obsahuje označené hodnotu min vyjadrujúci počet závislých vlastností, ktoré je potrebné zahrnúť do konceptu.

Obmedzenia

- Alternatívny variačný bod môže byť s ostatnými elementmi iba vo vzťahoch mandatory a optional.

Notácia



Implemented

Opis

Implemented predstavuje vzťah medzi vlastnosťou a elementom (elementmi) UML, prostredníctvom ktorých je táto vlastnosť implementovaná. Vlastnosť vo vzťahu vždy vystupuje v roli *supplier* ostatné elementy v roli *client*. *Typ* elementu závisí od druhu vlastnosti, napr. funkčné vlastnosti systému sú implementované ako komponenta, resp. triedy.

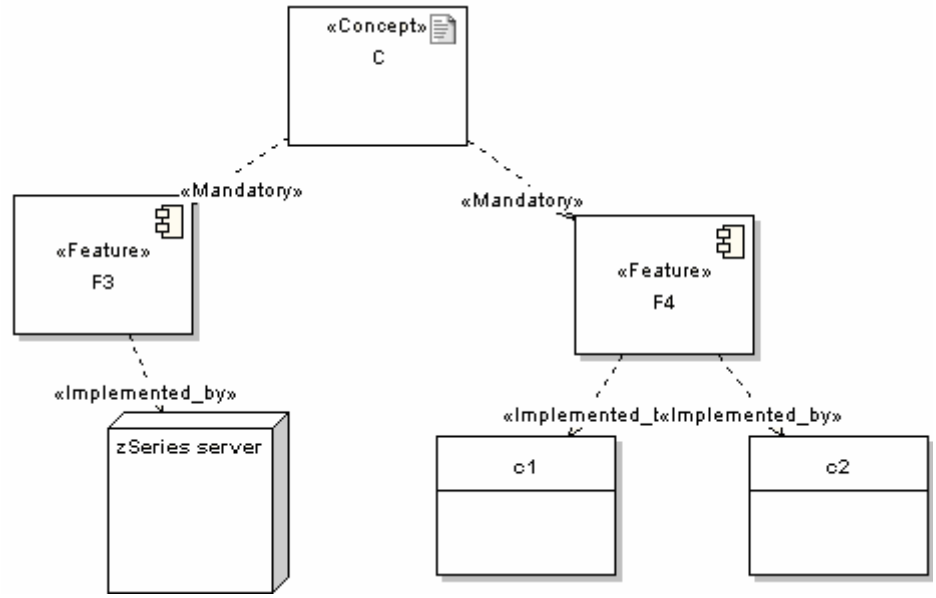
Stereotyp a označené hodnoty

Vzťah implemented je reprezentovaný v profile ako stereotyp <<implemented>> špecializujúcim metatriedu realization, ktorá tomuto vzťahu sémanticky najviac zodpovedá.

Obmedzenia

- V roli supplier sa môže nachádzať výhradne iba stereotyp <<feature>>

Notácia



Layer

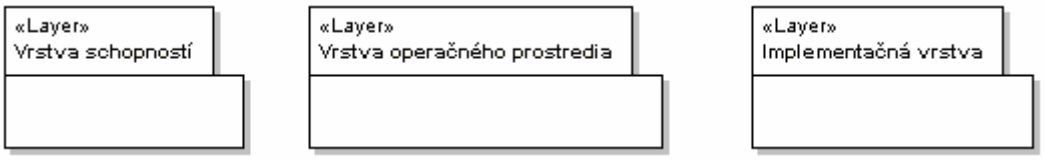
Opis

Vrstvy zavedené do modelovania vlastností metódou FORM, je možné modelovať v navrhovanom profile pomocou elementu Layer. Medzi vlastnosti nachádzajúcimi sa v rôznych vrstvách môžu existovať všetky doposiaľ uvedené vzťahy.

Stereotyp a označené hodnoty

Vrstva je reprezentovaná ako stereotyp <<layer>> špecializujúci metatriedu package, ktorá umožňuje združovať viaceré elementy do jednej skupiny.

Notácia



Subfeature dependency

Opis

Prvok subfeature dependency predstavuje všeobecnú kardinalitu uvedenú v rozšírenej notácii Czarneckého-Eiseneckera. Prvok predstavuje vzťah medzi dvoma vlastnosťami, v ktorom v prípade zahrnutia nezávislej vlastnosti do konceptu je prostredníctvom kardinality špecifikované zahrnutie/nezahrnutie závislej vlastnosti.

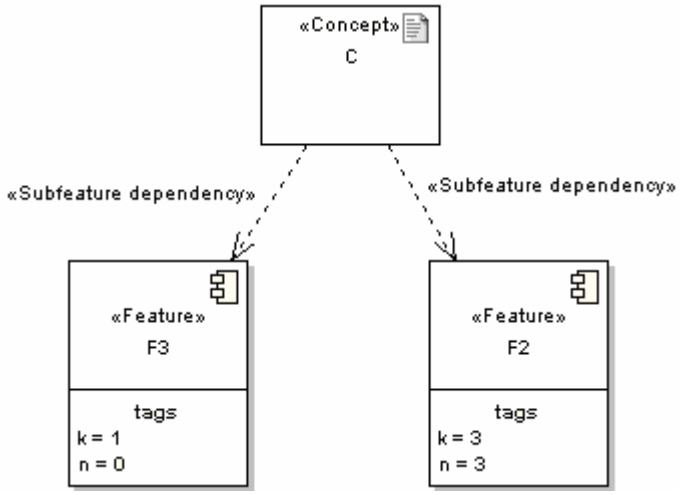
Stereotyp a označené hodnoty

Prvok je reprezentovaná stereotypom <<subfeature dependency>> metatriedy dependency, ktorá vyjadruje všeobecnú závislosť dvoch elementov jazyka UML. Kardinalita vlastností je špecifikovaná pomocou dvoch označených hodnôt n,k, pričom dvojica n=0, k=1 predstavuje voliteľnú vlastnosť a dvojica n=1, k=1 povinnú vlastnosť.

Obmedzenia

- Vzťah môže existovať iba medzi stereotypmi <<concept>>, <<feature>> a variačnými bodmi
- <<concept>> je vždy nezávislým elementom
- <<feature>> je vždy závislým elementom
- Zahrnutie nezávislého elementu do konceptu implikuje zahrnutie závislého elementu

Notácia



Group dependency

Opis

Stereotyp <<group dependency>> reprezentuje všeobecnú kardinalitu skupiny vlastností uvedenej v rozšírenej notácii Czarneckého-Eiseneckera. Prvok predstavuje vzťah medzi nezávislou vlastnosťou a skupinou vlastností. V prípade zahrnutia nezávislej vlastnosti je do konceptu zahrnutých n vlastností z intervalu špecifikovanom označenými hodnotami prvku.

Stereotyp a označené hodnoty

Prvok je reprezentovaný stereotypom <<group dependency>> špecializujúcim metatriedu artifact a obsahuje dve označené hodnoty:

- n – minimálny počet vlastností, ktoré musia byť zahrnuté do konceptu po zahrnutí nezávislej vlastnosti.

- k – maximálny počet vlastností, ktoré môžu byť zahrnuté do konceptu po zahrnutí nezávislej vlastnosti.

Špeciálnymi prípadmi tohto vzťahu sú nasledujúce kombinácie označených hodnôt:

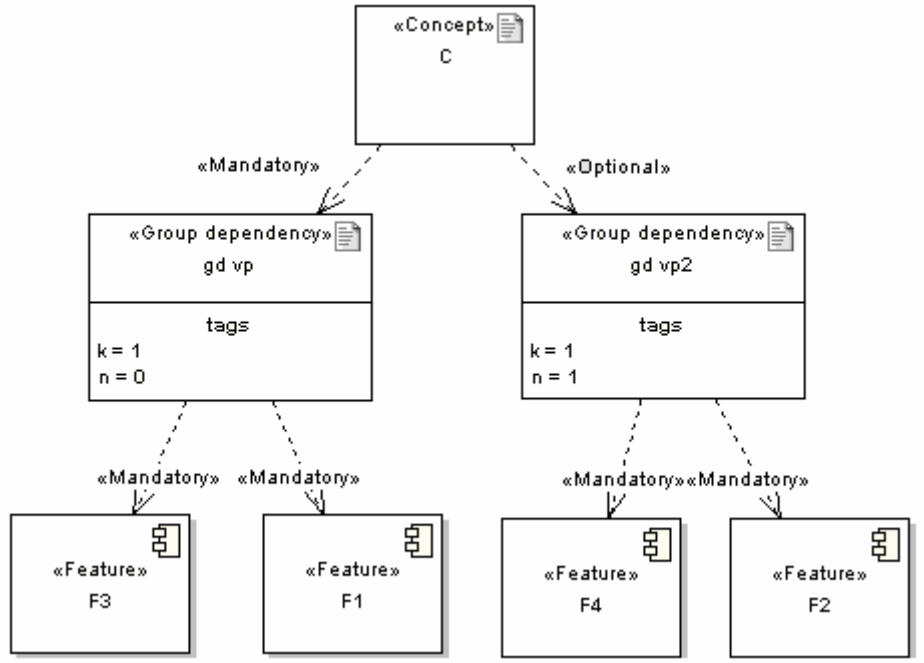
n=1, k=* - predstavuje alternatívne vlastnosti

n=0, k=* - predstavuje alebo – vlastnosti

Obmedzenia

- Prvok group dependency môže byť s ostatnými elementmi iba vo vzťahoch mandatory a optional.

Notácia



Príloha B Elektronické médium

Súčasťou diplomovej práce je elektronické médium (CD) obsahujúce nasledovné časti:

Install\

Inštalácie podporných a modelovacích nástrojov pre Windows:

- Acrobat Reader 6
- Enterprise Architect 6
- Omondo Eclipse Studio Edition 2.1
- Eclipse SDK 3.1
- Eclipse EMF 2.1
- Eclipse GEF 3.1
- Eclipse UML 2.1

Docs\

Výstupy diplomovej práce:

- Diplomová práca (diplomova_praca.pdf)
- Profil UML (profil.xml)
- Model vlastností vytvorený pomocou profilu v prostredí EA (model.eap)