Prednáška 7:
Preťažovanie a prekonávanie metód

Ján Lang

kanc. 4.34, lang@fiit.stuba.sk, http://www2.fiit.stuba.sk/~lang/zoop/

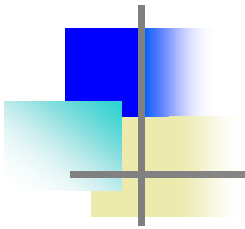
Ústav informatiky, informačných systémov a softvérového inžinierstva
Fakulta informatiky a informačných technológií
Slovenská technická univerzita v Bratislave
31. október 2023

Dedenie (inheritance)

- Generalizácia, abstrakcia (zovšeobecnenie, zovšeobecňovanie) – sú fundamentálne princípy dedenia
  - Ovocie (jablko, hruška,...)
  - Zelenina (paprika, mrkva,...)
- Abstrahovanie – uberanie špecifických vlastností, čŕt – orientované na spoločné
- Konkretizácia (antonymum) – pridávanie špecifických vlastností, čŕt
  - Buldozér, žeriav, autobus,... (Dopravný prostriedok)
- Úroveň abstrakcie implikuje pozíciu v hierarchii

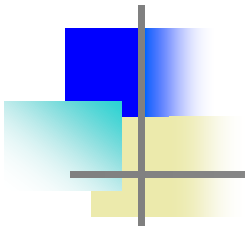
- Príklad na dedenie obdĺžnik a kváder, kváder a obdĺžnik?

Dedenie (inheritance)



- Príklad na dedenie obdĺžnik a kváder, kváder a obdĺžnik?
- Ako bude vyzerať hierarchia dedenia?
- Ktory z typov bude nadtypom v hierarachii dedenia?
- Kváder?
- Obdlžnik?

- ...na zamyslenie:
- Kam zaradíme *nájomníka* v hierarchii dedenia (človek, obyvateľ, turista, *...nájomník?*)
- kruh vs elipsa, elipsa vs kruh?



Príklad

- Trieda `Obdlžnik`
- Konštruktor s 2D parametrami `sirka, vyska`
- Verejná metóda `dlzkaUhlopriecky()` vracajúca `double`
  - `double pom;`
  - `pom=(sirka*sirka)+(vyska*vyska);`
  - `return Math.sqrt.(pom);`
- Verejná metóda `getSirka()` vracajúca hodnotu atribútu `sirka`
- Špeciálny podtyp trieda `Kváder`
- Konštruktor s 3D parametrami `sirka, vyska, hlbka`
  - `super(sirka, vyska);`
  - `this.hlbka = hlbka;`
- Verejná metóda `dlzkaUhlopriecky()` vracajúca `double`

Dedenie (inheritance)

- Generalizácia, abstrakcia (zovšeobecnenie, zovšeobecňovanie) – sú fundamentálne princípy dedenia
  - Ovocie (jablko, hruška,...)
  - Zelenina (paprika, mrkva,...)
- Abstrahovanie – uberanie špecifických vlastností, čŕt – orientované na spoločné
- Konkretizácia (antonymum) – pridávanie špecifických vlastností, čŕt
  - Buldozér, žeriav, autobus,... (Dopravný prostriedok)
- Úroveň abstrakcie implikuje pozíciu v hierarchii

- ...ako by to bolo v prípade, kruhu a elipsy - metóda nakresli()?

Dedenie (inheritance)

- Generalizácia, abstrakcia (zovšeobecnenie, zovšeobecňovanie) – sú fundamentálne princípy dedenia
  - Ovocie (jablko, hruška,...)
  - Zelenina (paprika, mrkva,...)
- Abstrahovanie – uberanie špecifických vlastností, čŕt – orientované na spoločné
- Konkretizácia (antonymum) – pridávanie špecifických vlastností, čŕt
  - Buldozér, žeriav, autobus,... (Dopravný prostriedok)
- Úroveň abstrakcie implikuje pozíciu v hierarchii

Princíp zviazanosti a súdržnosti

- ...*low in coupling and high in cohesion* – málo zviazaný vysoko súdržný
- Súdržnosť – miera do akej prvky triedy patria dohromady
  - Všetok súvisiaci kód by mal byť „blízko" pri sebe. Úsilie o vysokú súdržnosť. Zoskupiť všetok súvisiaci kód dohromady čo najviac. Rieši sa to v rámci **triedy**
- Zviazanosť – miera do akej sú triedy na sebe vzájomne závislé
  - Všetky triedy by mali byť sebestačné a na sebe čo najmenej závislé. Rieši sa to na úrovni **tried**
- Mnohokrát problematické
- Vyjadrenie vzťahov medzi triedami vhodné diagramom tried v jazyku UML

Princíp zviazanosti a súdržnosti

Metriky

- **CBO - Coupling between Objects**
  Coupling between objects (CBO) is a count of the number of classes that are coupled to a particular class i.e. where the methods of one class call the methods or access the variables of the other. These calls need to be counted in both directions so the CBO of class A is the size of the set of classes that class A references and those classes that reference class A. Since this is a set - each class is counted only once even if the reference operates in both directions i.e. if A references B and B references A, B is only counted once.

Chidamber, S. and Kemerer. C. (1994). A metric suite for object oriented design, IEEE Transactions on Software Engineering

Princíp zviazanosti a súdržnosti

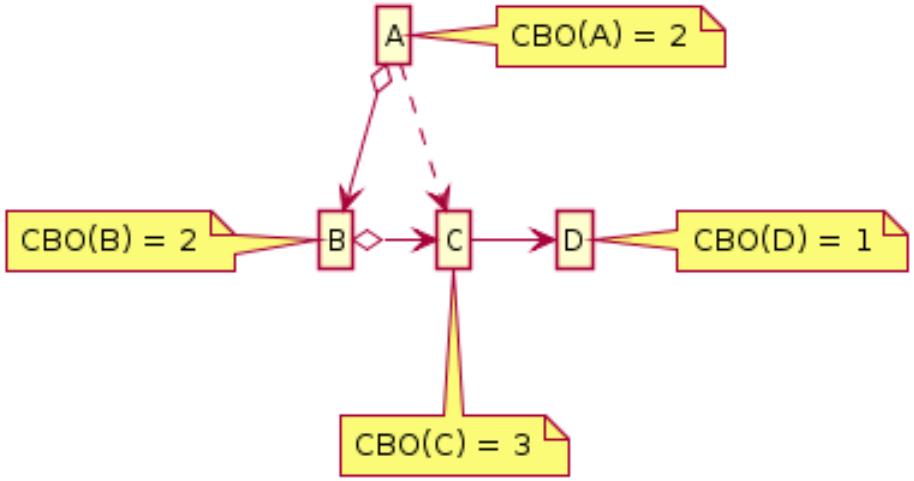
Metriky

- CBO - Coupling between Objects
- **CBO medzi balíkmi**
- Princíp metriky: previazanie modulu je počet unikátnych tried volaných z daného modulu. Previazanie celého projektu je potom priemerovaná hodnota, t.j. súčet previazania jednotlivých modulov delený počtom modulov v projekte.
- Remodularizácia tried

Princíp zviazanosti a súdržnosti

Metriky

- **CBO** doesn't care about the direction of a dependency. D has a CBO of 1 because C depends on it, even though D depends on no other classes. B and C are similar cases.
- Coupling can be via attributes (composition), associations, local variables, instanciations or injected dependencies (arguments to methods).



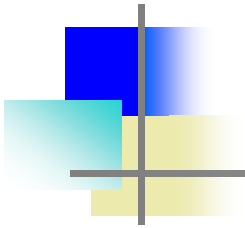
https://stackoverflow.com/questions/27515541/cbo-coupling-between-object

Princíp zviazanosti a súdržnosti

Metriky

- **CBO - Coupling between Objects**
- CBO should be as low as possible for three reasons -
- Increased coupling increases interclass dependencies, making the code less modular and less suitable for reuse.
- More coupling means that the code becomes more difficult to maintain
- The more links between classes the more complex the code and the more difficult it will be to test

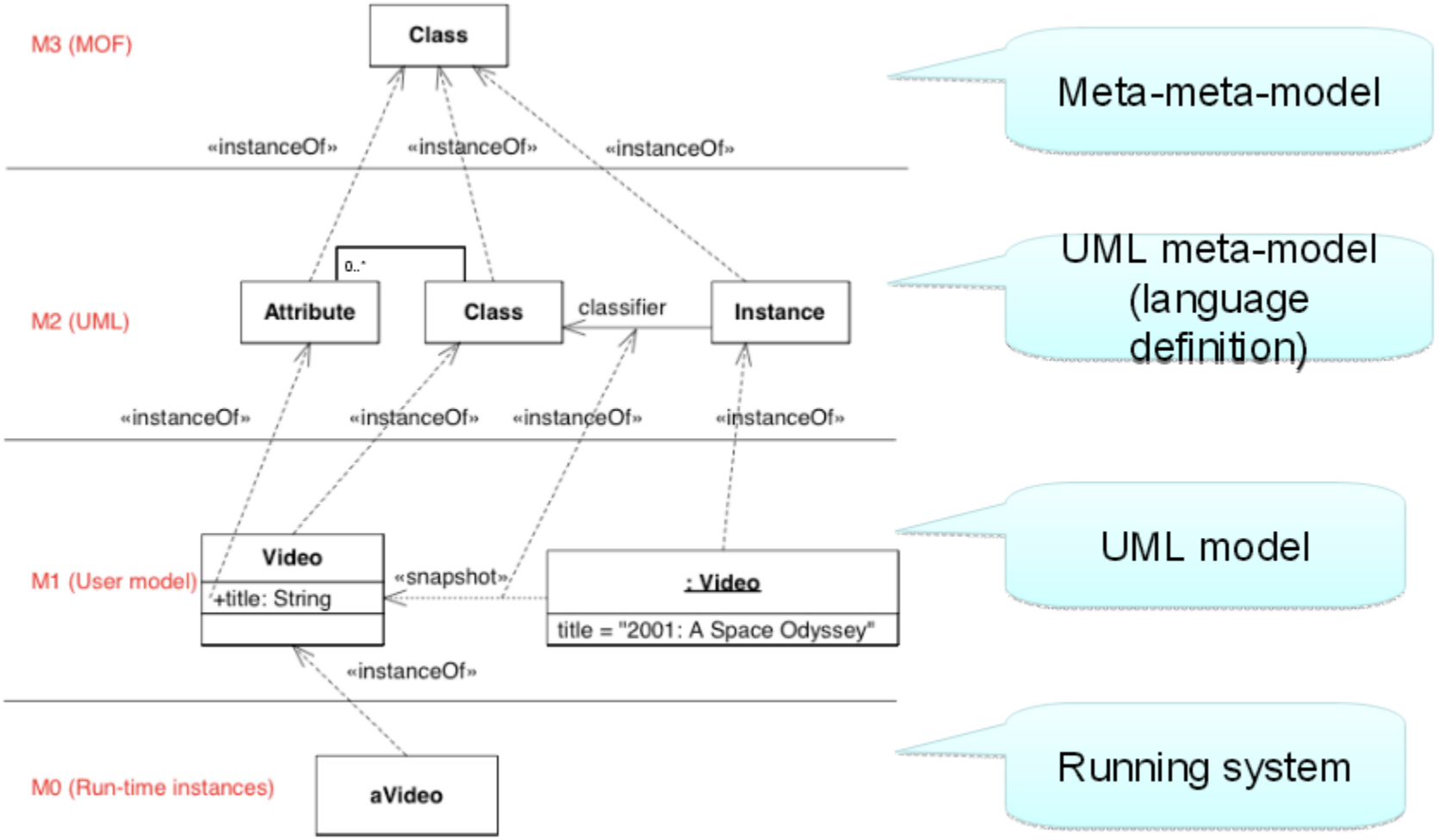
- Príklad Banka (špecifické podtypy účtov, - FrontEnd pohľad na vec, otázka BackEnd-u... Organizácia, preskupovanie v rámci architektúry, viac balíkov...



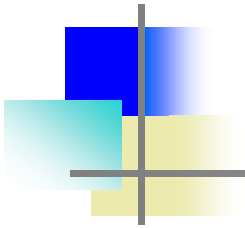
Dedenie

- Uzavretosť, otvorenosť znovu použiteľných častí
- Uzavretý – pre jeho použitie nič iné netreba, má všetko potrebné, používateľ nemá možnosť nič meniť...
- Otvorený – používateľ má možnosť meniť, upravovať, modifikovať, pridávať...

Štvorúrovňová hierarchia v UML

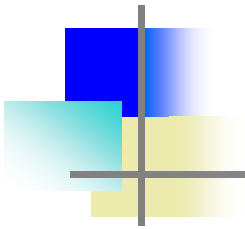


(example from UML Infrastructure formal/09-02-04, page 19)



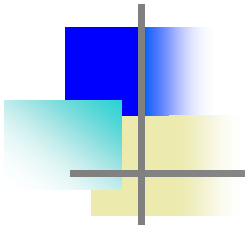
Príklad

- Vytvorte triedu `Električka`, ktorá umožní:
  - pohyb električky
  - zastavenie električky
  - otvorenie dverí
  - zatvorenie dverí
  - nastúpenie zadaného počtu cestujúcich
  - vystúpenie zadaného počtu cestujúcich
  - zistenie aktuálneho počtu cestujúcich
- zabezpečte príslušné kontroly (napr. nevystupovať cez zatvorené dvere, nedovoliť pohnúť sa kým sú dvere otvorené, nenastupovať ak je električka plná a pod.



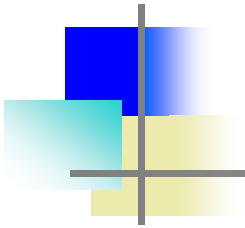
Príklad

- Napíšte triedu `Lekar` s premennými `meno, priezvisko, vek, pocetOperacii`, triedu `Sestricka` s premennými `meno, priezvisko, vek, pocetAsistencií`. A napíšte triedu `Operacia` s premennou typu `Lekar` a dvoma premennými typu `Sestricka` a metódou `vypis()`, ktorá vypíše informácie o operácii.
- ...na prednáške.



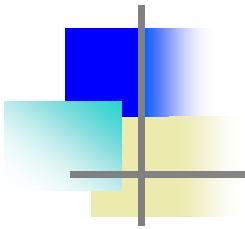
Príklad

- Navrhnite a napíšte triedu `Zelenina` s aspoň dvoma premennými a aspoň dvoma metódami a triedu `Paprika`, ktorá bude dediť od triedy `Zelenina` a bude mať navyše aspoň jednu premennú a aspoň jednu metódu a bude **prekonávať** jednu metódu triedy `Zelenina`.
- ...na prednáške.



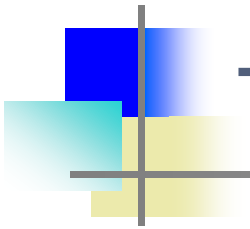
Príklad

- Pridajte do triedy `Zelenina` jednu finálnu metódu a overte, že sa nedá skryť v triede `Paprika`.
- Je možné finálnu metódu preťažiť?
- ...na prednáške.



Príklad

- Vytvorte triedu `Kufor`, ktorá umožní:
  - otvorenie,
  - zatvorenie,
  - pridanie zadaného množstva obsahu,
  - výber zadaného množstva obsahu,
  - vrátenie aktuálnej hmotnosti,
- a zároveň bude kontrolovať nekorektné operácie (pridanie nad dovolené množstvo, pokus vybrať viac ako je vložené a pod.)



Test

Semestrálny test

Semestrálny test bude 7. novembra 2023 v čase prednášky v miestnosti -1.58 (U120) FIIT STU.

TODO nezabudnite

- Aj vy môžete pomôcť vylepšiť tento predmet študentom pre nasledujúci akademický rok. Vaše odporúčanie, komentár či otázka.

  ...cez spätnoväzobný formulár.