Přehled metodiky vývoje GIS aplikací

(vytvořeno pro seminář na FIMU: Vybrané kapitoly z GIS, podzimní semestr)

Lekce 4: Datové modely

Obsah

1. Datové modelování - proč vytvářet datové modely?
2. Tři úrovně datových modelů
3. Konceptuální datový model (analytický model tříd), entity, vztahy mezi entitami, atributy, datové typy, primární klíče, unikátní klíče
4. Logický datový model (LDM), tabulky, vazby mezi nimi, sloupce, postup vytváření LDM
5. Fyzický datový model
6. Datové typy
7. Formální náležitosti datového modelu
8. Příklady datových modelů
9. Zásady tvorby datového modelu
10. Normální formy datového modelu
11. Na závěr: DTO a jejich využití
12. Podklady pro další samostudium

Datové modelování - proč vytvářet datové modely?

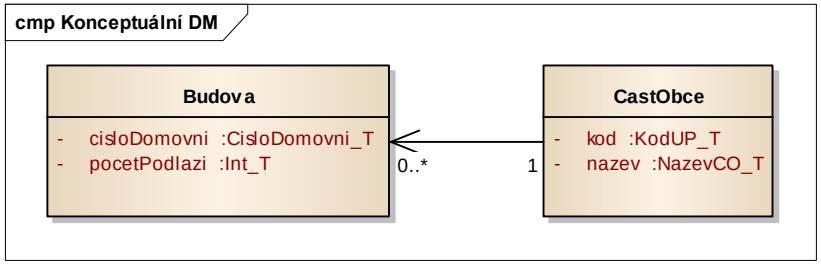
- Datové modelování (Anatoliy Kybkalo, Datové modelování, Unicorn College, 2013):
  - Datové modelování je oborem, který se zabývá návrhem struktury a organizace dat.
  - Je to proces, při kterém převádíme reálné objekty na objekty datové, s cílem zachytit v datovém modelu realitu, o které si chceme uchovávat informace.
  - Návrh správné databáze není vždy jednoduchý úkol a mnoho databázových architektů se potýká s problémy již v počátcích projektu, kdy je potřeba identifikovat základní entity, které budou obsazeny v databázi.
  - Při návrhu databáze není pouze jedno správné řešení, ale je jich hned několik.
- Proč vytvářet datové modely:
  - Protože chci poznat realitu (WOI, world of interest), její objekty a vztahy mezi nimi
  - Protože chci *dobře* navrhnout uložení dat v informačním systému
  - Protože chci *dobře* navrhnout funkce informačního systému
    - funkce pracují s XML daty, proč vytvářet datové modely?

Tři úrovně datového modelu

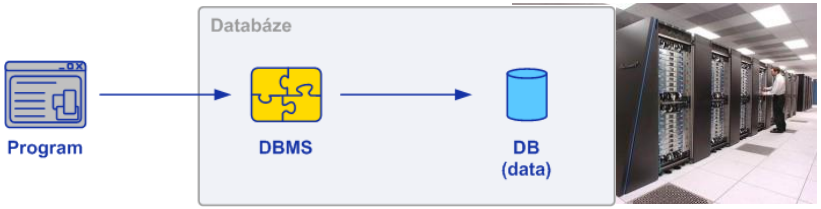
objekt reálného světa

analýza

konceptuální datový model (model tříd)

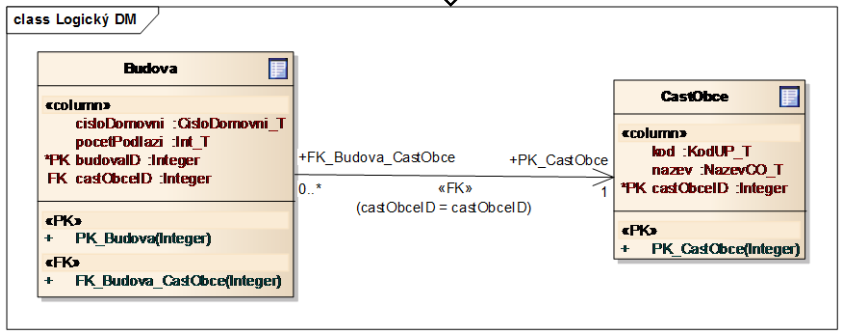


design




```
CREATE TABLE Budova (
                    cisloDomovni  CisloDomovni_T,
                    pocetPodlazi  Int_T,
                    budovaID      Integer NOT NULL,
                    castObceID    Integer);
CREATE TABLE CastObce (
                    kod       KodUP_T,
                    nazev     NazevCO_T,
                    castObceID  Integer NOT NULL);
```

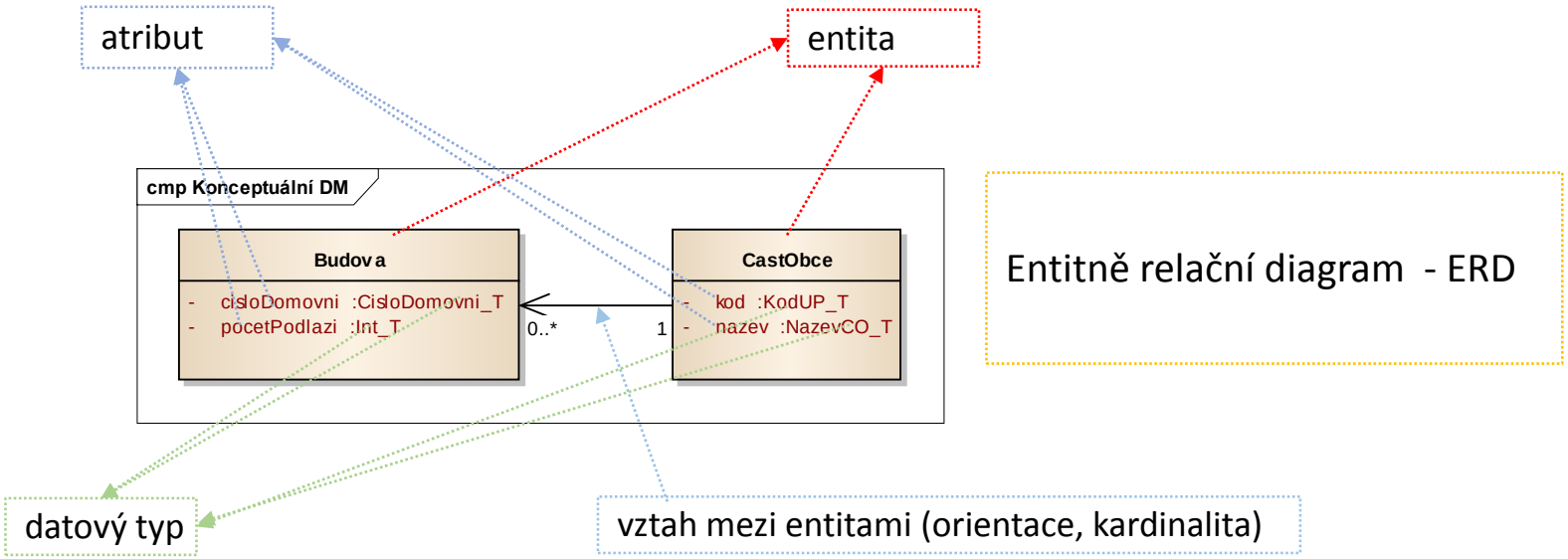
programování
(implementace)



fyzický datový model (implementační datový model)

logický datový model (technologický datový model)

Konceptuální datový model



- Postup
  - vymezení entit, jejich přesná definice
  - pozor: gnoseologické kategorie obecné x jednotlivé = entita x instance entity
  - určení vztahů mezi entitami, orientace z nadřízené do podřízené entity (tabulka reprezentující v logickém DM podřízenou entitu bude obsahovat cizí klíč)
  - nahrazení vztahů m:n asociativními entitami (pokud to nechci dělat až v LDM)
  - určení významných atributů a jejich definice
  - určení datových typů atributů, jejich formální popis a jejich průběžná konsolidace
  - doplnění všech atributů, jejich definic a jejich datových typů
  - určení speciálních atributů - unikátních (business) klíčů
- Diagram (ERD) není datový model, datový model je uložen v repository (úložišti) CASE nebo (také) v hlavě analytika
- Diagram slouží jednak jako jedna z možností vytváření prvků modelu, jednak pro prezentaci modelu
- Diagramů může (musí) být v jednom modelu více!
- Údržba konceptuálního datového modelu (?)

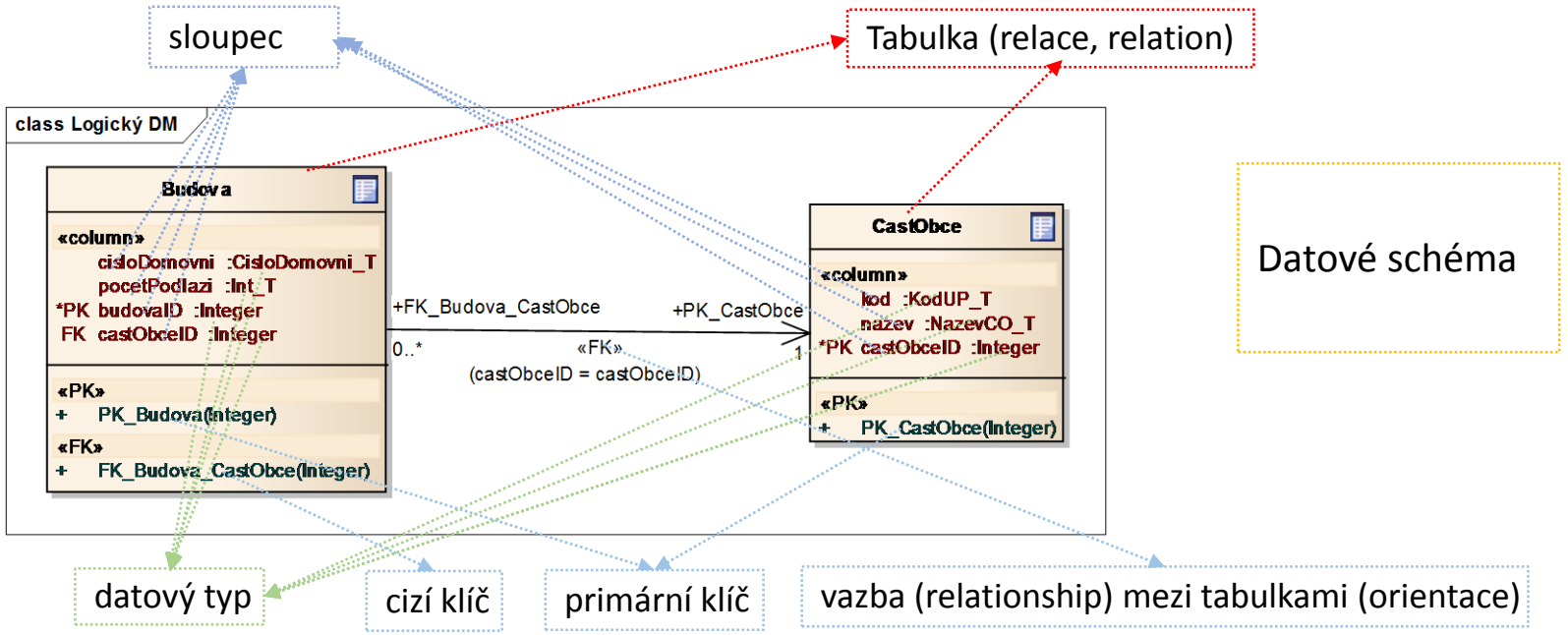
Konceptuální datový model – „kuchařka“

Zdroj langer.zam.slu.cz/teaching/db/P02-Konceptualni%20modelovani.pps

1. Zvolte jednu (primární) entitu ze specifikace požadavků.
2. Určete atributy, jejichž hodnoty se mají pro tuto entitu zaznamenávat. Označte případné klíče (identifikátory) a vytvořte ukázková data.
3. Popište slovně navrženou entitu, její atributy a klíče.
4. Prověřte funkční vztahy (závislosti) atributů a v případě potřeby entitu normalizujte.
5. Prověřte atributy navržené entity a zjistěte, zda je třeba zaznamenávat informace o jednom či více atributech v nové samostatné entitě.
6. Je-li vhodné vytvořit další entitu, zakreslete ji do diagramu a vraťte se na krok 2.
7. Spojte entity vztahy, pokud tyto existují. Popište slovně vztahy mezi entitami z obou stran.
8. Prověřte seznam atributů a určete, zda některé z nich potřebují být identifikovány prostřednictvím dvou (či více) entit. Pokud ano, umístěte atribut na příslušný vztah, který spojuje dané entity.
9. Prověřte, zda v diagramu nemáte „smyčky“, které mohou indikovat nadbytečné (odvozené) vztahy. Pokud je vztah skutečně redundantní, odstraňte ho.
10. Vytvořte ukázková data.
11. Předveďte navržený model uživateli - pokud je to třeba, upřesněte diagram.

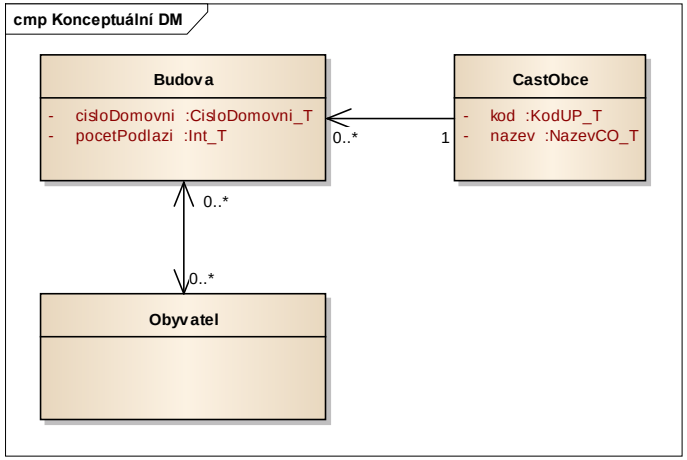
Další návod: Zendulka,
http://www.fit.vutbr.cz/study/courses/DSI/public/pdf/nove/DodatekA_metodologieERD.pdf

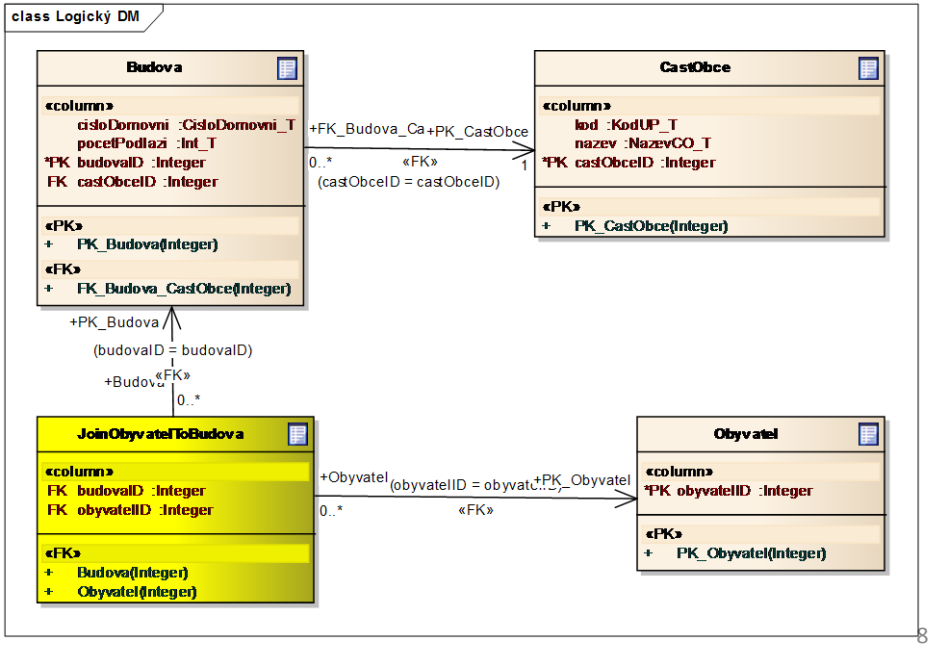
Logický datový model (relační přístup)



- Postup
  - automatické vygenerování LDM z konceptuálního DM pomocí CASE
  - vytvoření asociačních entit ze vztahů m:n (pokud jsem neudělal už v rámci konceptuálního datového modelu nebo pokud to neudělá CASE sám)
  - pro Enterprise Architect: změna datových typů vygenerovaných sloupců
- Údržba logického DM – ANO! (a bude líp)
  - neudržovat jen fyzický DM v DDL skriptech, ale odvozovat jej z LDM!
- Trochu algebry: tabulka je n-nární relace na doménách D1, D2, ... , Dn (přesněji je to dvojice R = (R, R*), kde R = R(A1:D1, A2:D2, ... An:Dn) je schéma relace R* ⊆ D1 x D2 x ... x Dn je tělo relace. A1, …, An jsou atributy. Schéma relace zapisujeme často zjednodušeně ve tvaru R(A1, A2, ... , An).

Konceptuální a logický datový model – vztahy kardinality m:n





Fyzický datový model pro relační databáze

Fyzický (physical) datový model (PDM) tvoří příkazy DDL (data definition language, část SQL, analytik musí zásady znát!), například:

```
CREATE TABLE Budova
(
        cisloDomovni  CisloDomovni_T,
        pocetPodlazi  Int_T,
        budovaID      Integer NOT NULL,
        castObceID    Integer
)
;
CREATE TABLE CastObce
(
        kod        KodUP_T,
        nazev      NazevCO_T,
        castObceID  Integer NOT NULL
)
;
ALTER TABLE Budova ADD CONSTRAINT PK_Budova
        PRIMARY KEY (budovaID)
;
ALTER TABLE CastObce ADD CONSTRAINT PK_CastObce
        PRIMARY KEY (castObceID)
;
ALTER TABLE Budova ADD CONSTRAINT FK_Budova_CastObce
        FOREIGN KEY (castObceID) REFERENCES CastObce (castObceID)
;
```

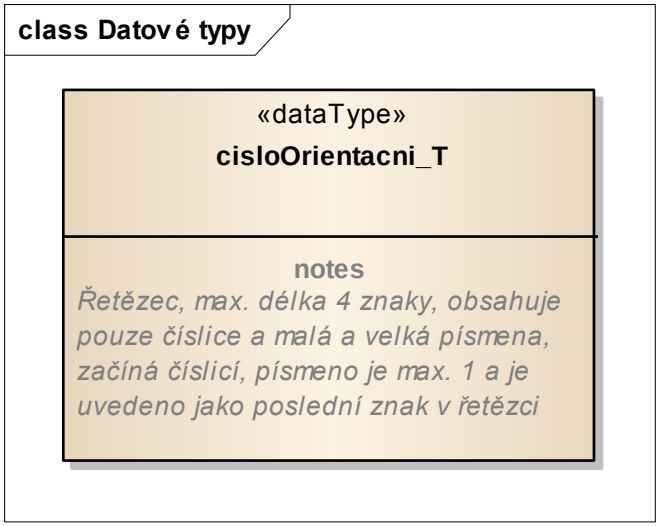
- Postup
  - automatické vygenerování PDM z LDM pro konkrétní DB pomocí CASE
  - doplnění PDM o další objekty (DB view, sekvence, …)

Datové typy

- Všechny atributy entit a dalších prvků datového modelu jsou striktně popsány datovými typy.
- Datový typ je prvek typu *class*, stereotypu *dataType*.
- Pro název datového typu je použita notace CamelCase (slova spojená bez mezery, počáteční písmena slov velké), navíc má příponu _T.
- Pravidla, která splňuje atribut daného datového typu, jsou popsána buď v popisu (notes) datového typu, nebo v dokumentech, které jsou připojeny k datovému typu (pomocí links nebo files).
- Doporučení: využít definice z informačního systému datových prvcích (ISDP, https://www.sluzby-isvs.cz/ISDP/DefaultSSL.aspx)

Příklad definice datového typu

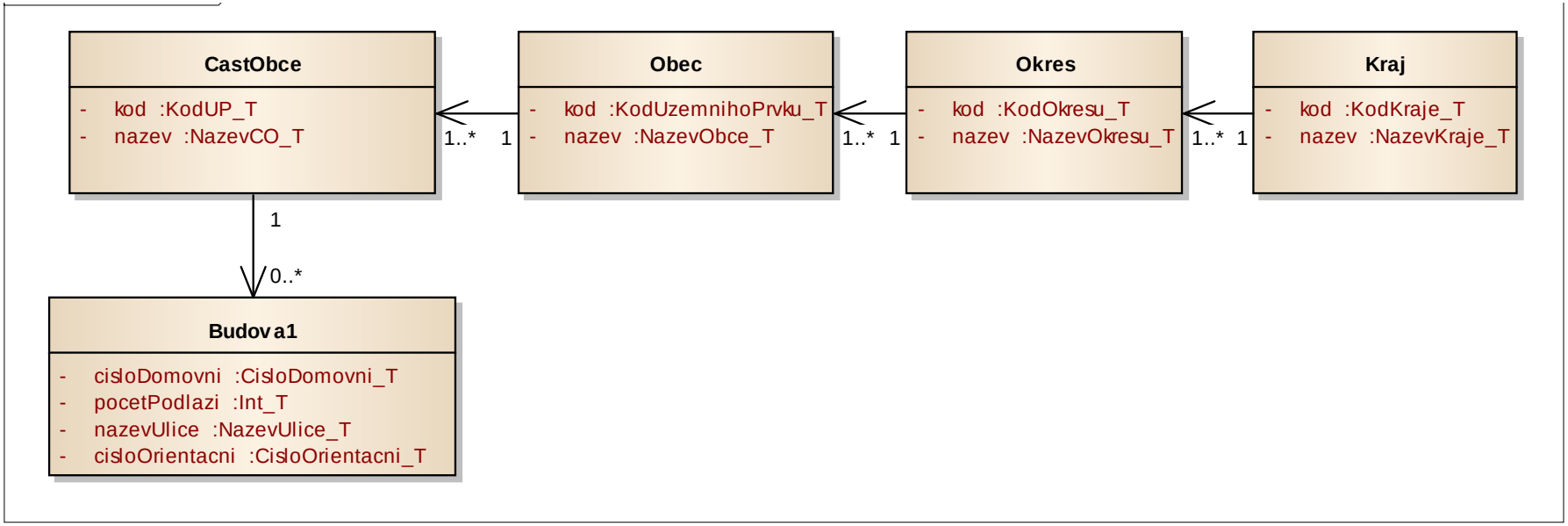
CisloOrientacni_T	
Typ	DataType
Package	Datove typy
Popis	Řetězec, max. délka 4 znaky, obsahuje pouze číslice a malá a velká písmena, začíná číslicí, písmeno je max. 1 a je uvedeno jako poslední znak v řetězci
Formát a omezení	String 4
	pouze malá a velká písmena a číslice, 1-4



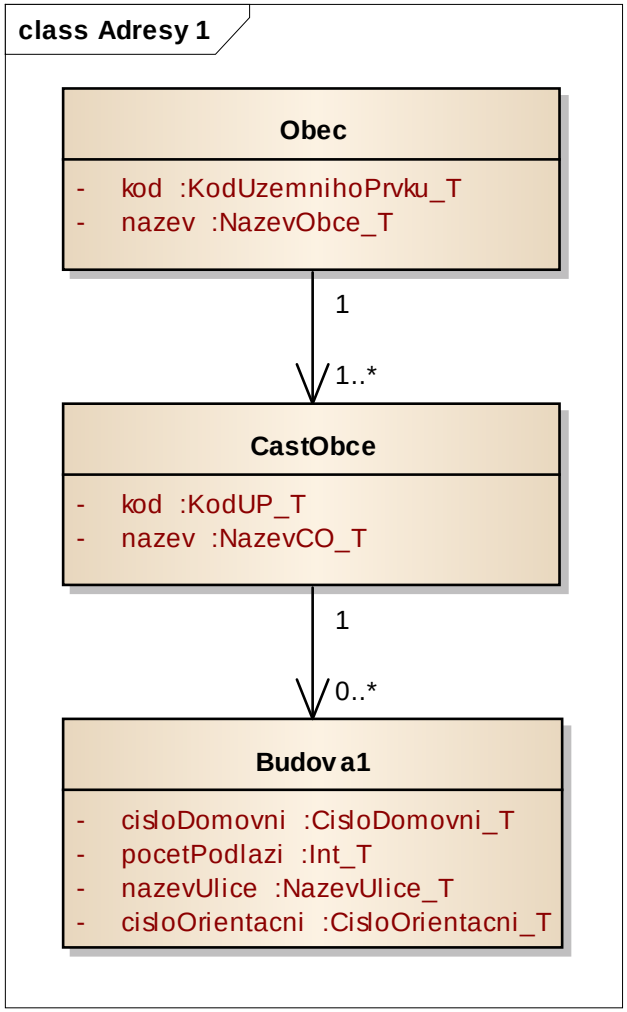
Formální náležitosti modelů

- Předepsaná struktura repository
- Názvy objektů (entit, tabulek, datových typů a dalších) v CamelCase (CastObce)
  - (názvy entit v jednotném čísle, názvy tabulek v množném čísle)
- Názvy atributů (a také operací, metod) v camelCase (cisloDomovni)
- Názvy bez diakritiky
- V názvech minimalizovat zkratky
- Neuvádět typy objektů v jejich názvech (přípona _T v datových typech je z tohoto pohledu chybně)
- Při prezentaci modelu rozdělit diagramy tak, aby každý z nich obsahoval okolo devíti objektů
  - nesdělovat čtenáři „podívej, jak složitý (datový) model jsem vytvořil“, ale „podívej, jak umím (datový) model přehledně prezentovat“
  - diagramy obsahují skupiny z nějakého pohledu příbuzných objektů
  - pokud se objekt vyskytuje ve více diagramech (například z důvodu znázornění vazeb je to nutné), výskyt objektů v kmenovém a ostatních diagramech je vhodné barevně odlišit

Příklad 1: budova – část obce – obec – okres – kraj



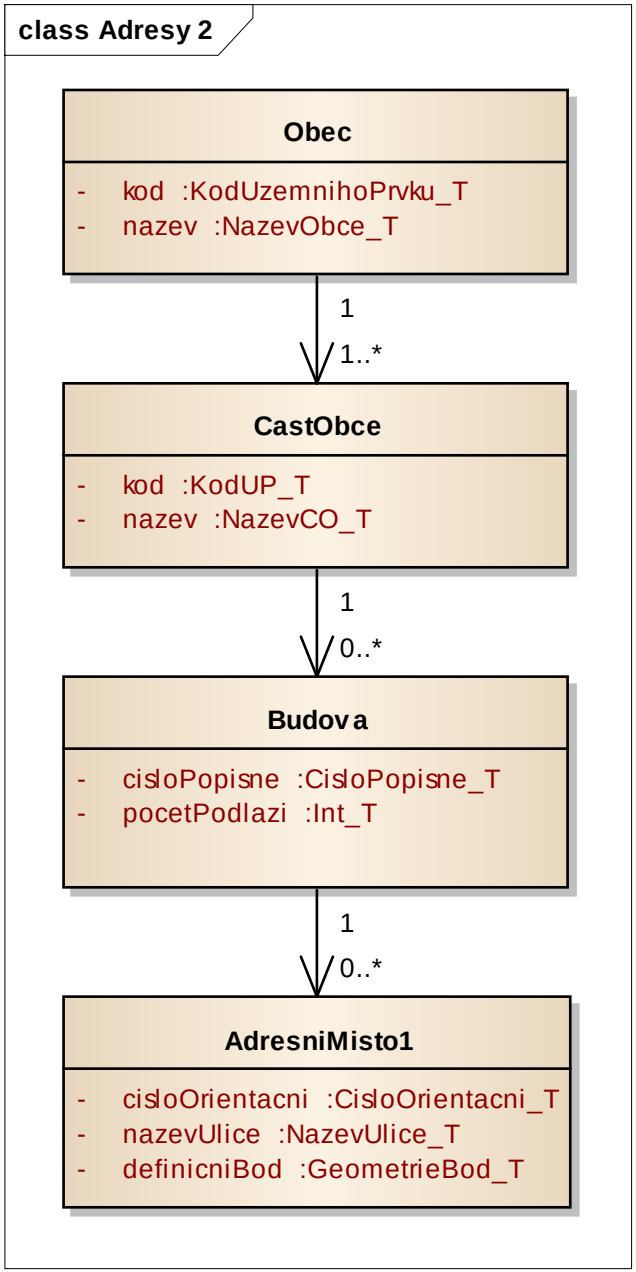
Příklad 2a: Adresa



Problém?

Každý vchod budovy má svoji adresu. Dokonce mohou být v různých ulicích.

Příklad 2b: Adresa



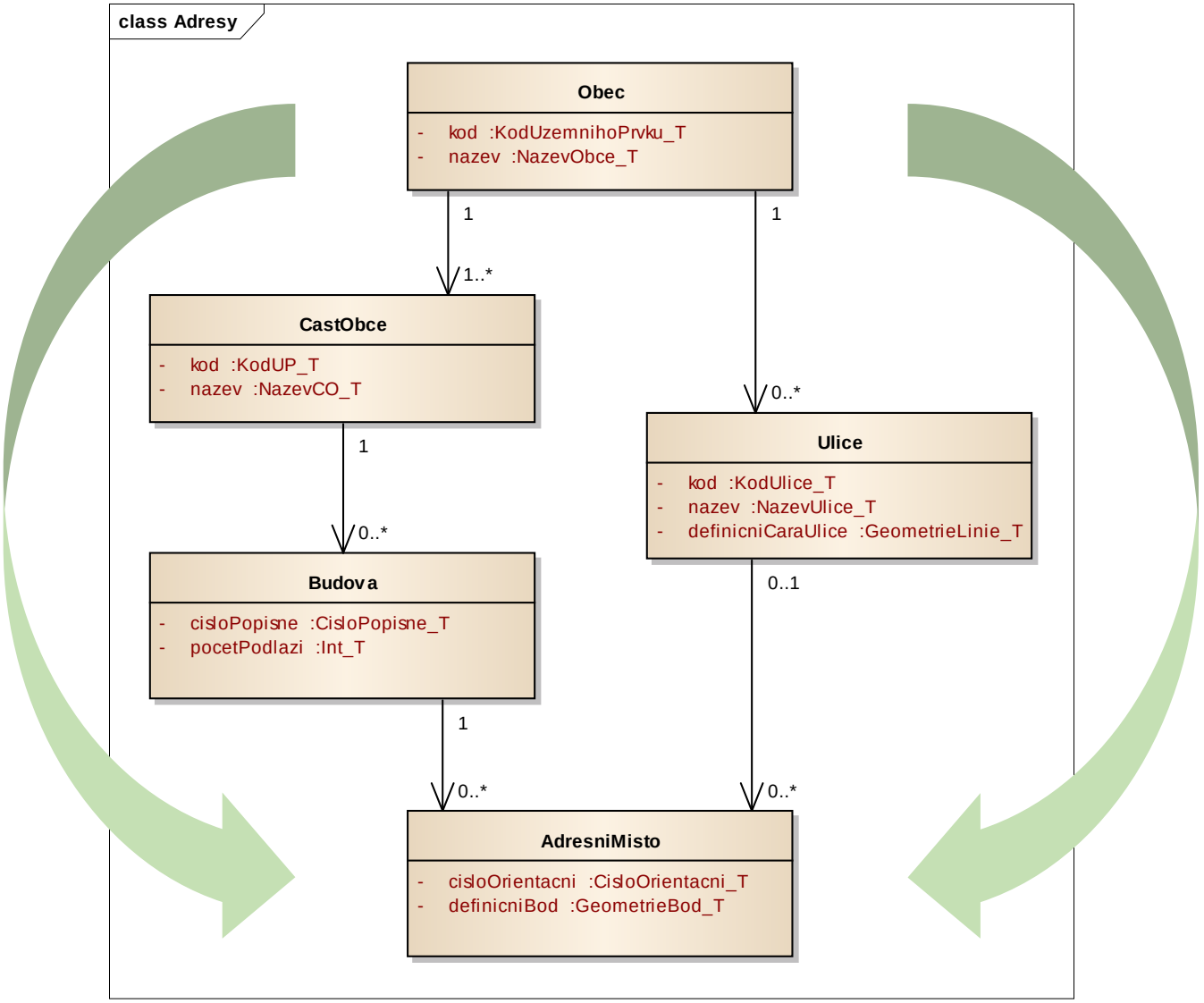
Problém?

Překlep v názvu ulice:
ulice Otokara Březiny x ulice Otakara Březiny

Změna názvu ulice.

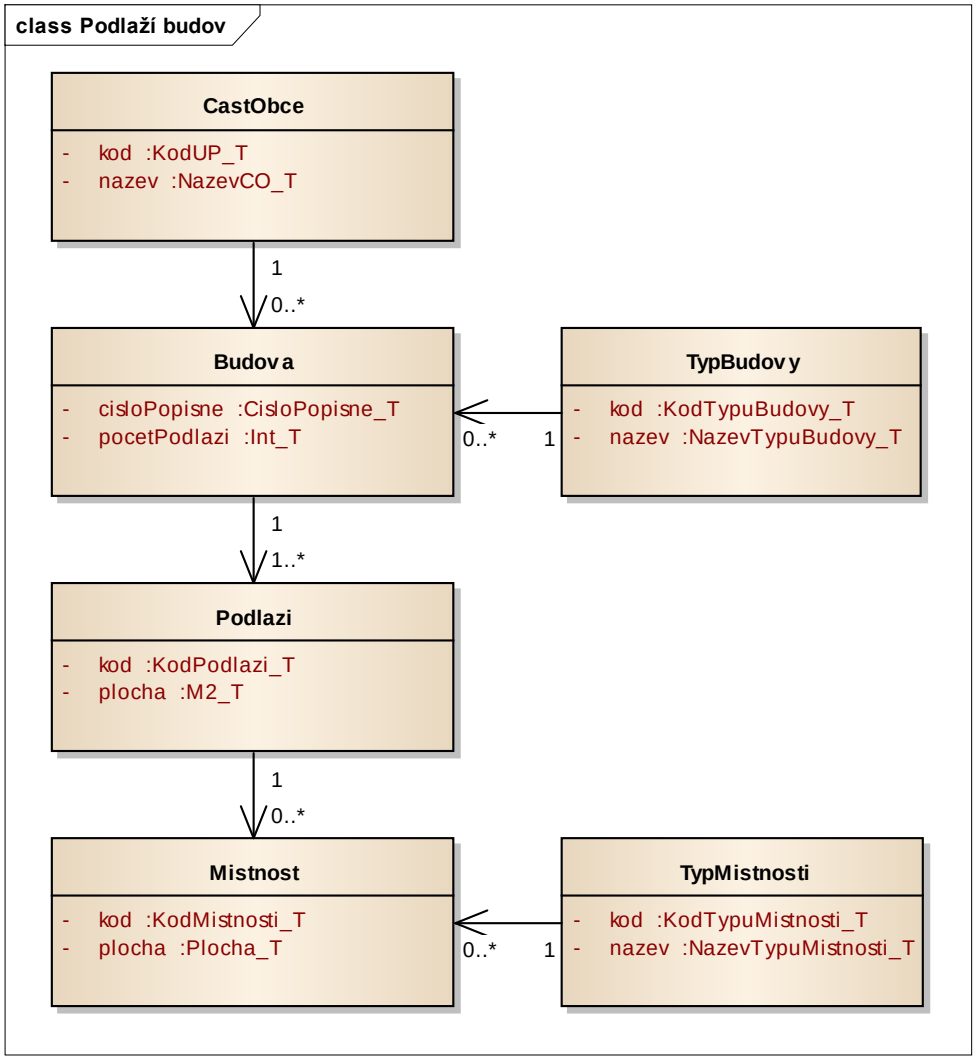
Definiční čára ulice.

Příklad 2c: Adresa



smyčka v datovém modelu

Příklad 3: místnost – podlaží – budova – část obce



Zásady tvorby datového modelu

Modeluj

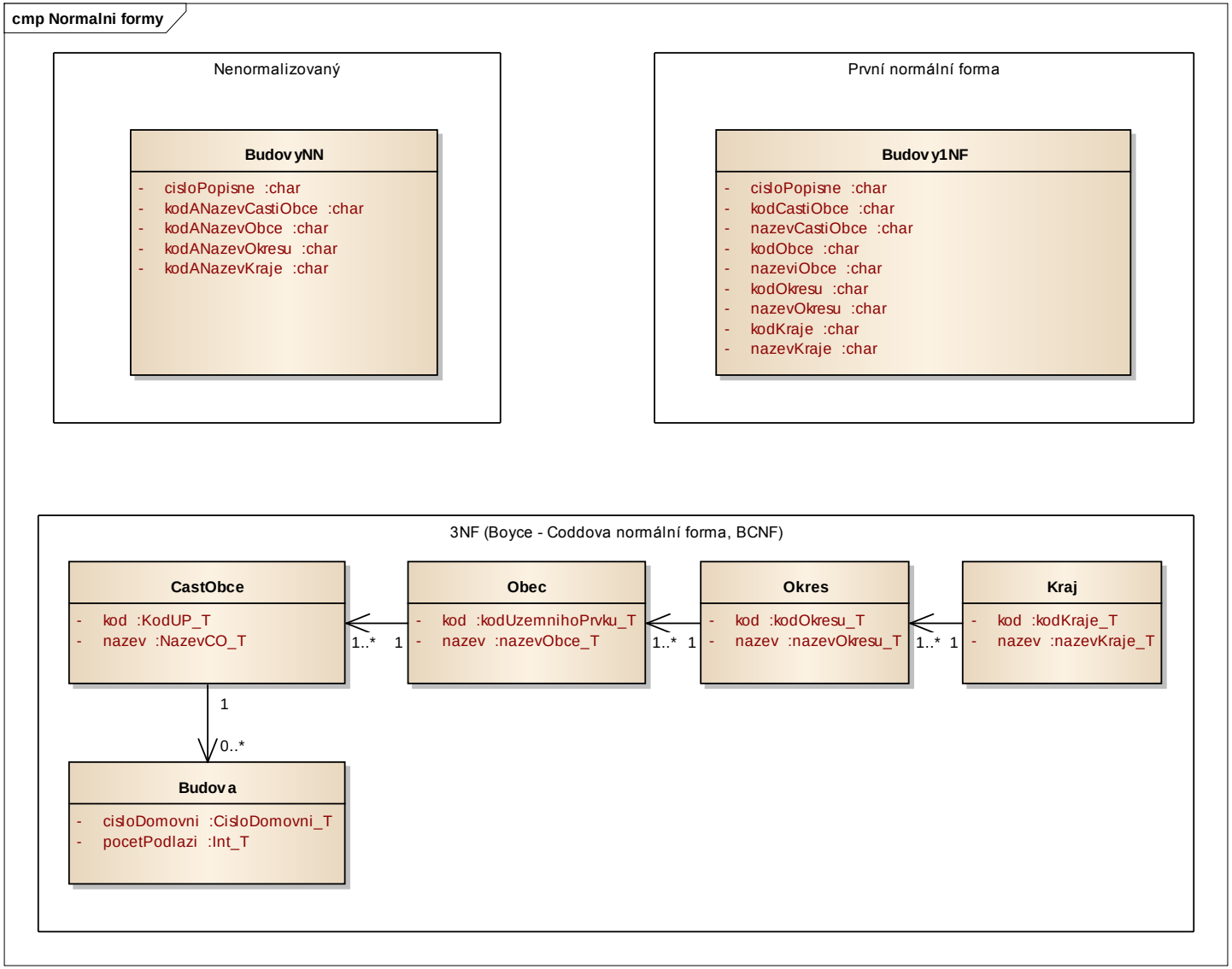
- Pochybuj!
- Diskutuj v týmu, ptej se, čti, googluj! Všechno už bylo namodelováno.
- Rozhoduj, hodnoť varianty podle účelu DM!
- Uvědomuj si stále, že krása i kvalita je v jednoduchosti!
- Pokud ti model někde drhne, je to tvoje chyba, není to tak složité, jak se ti to jeví!

Připrav si balíček první pomoci před tím, než model ukážeš programátorovi ☺

Minimalizuj chyby

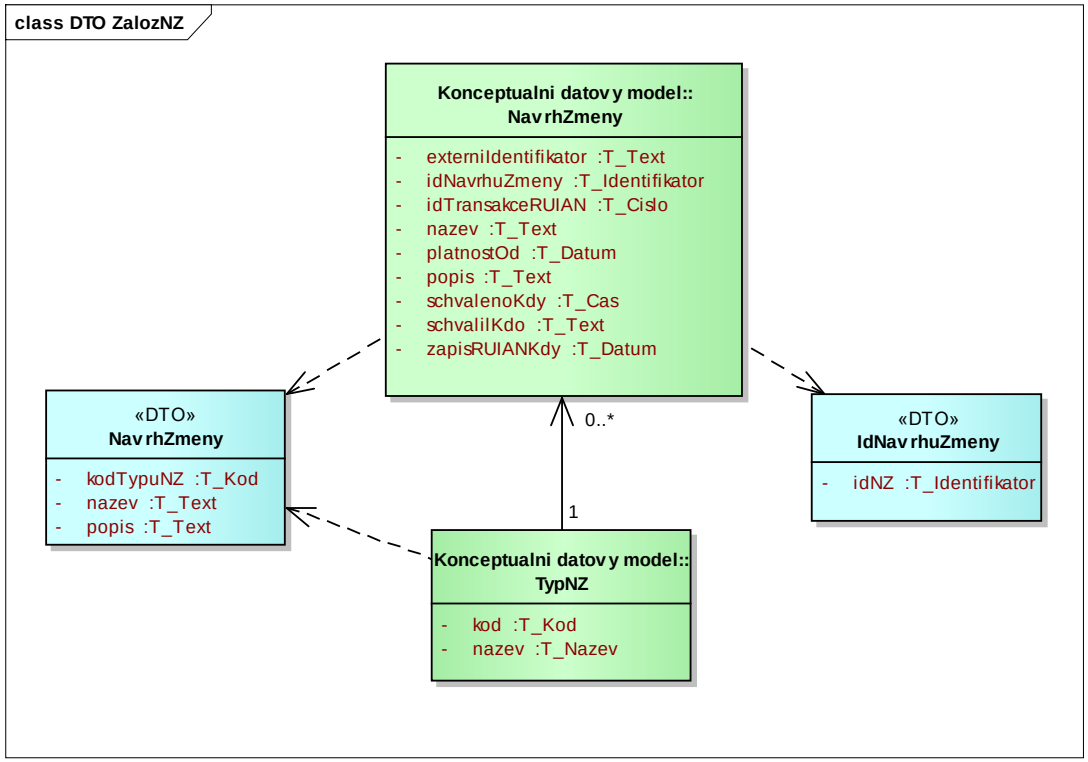
- Nechybí ti entita? Hlavnímu analytikovi ISKN chyběl list vlastnictví.
- Nechybí ti atribut v nějakém vztahu? V RPP analytikům Accenture chybělo pořadí právního předpisu.
- Nenamodeloval jsi náhodou jednoduchý datový model všeho? „Nedomodelováno“, příklady: PTE, obecný model GIS, kmenové evidence v ČSSZ.
- Nenamodeloval jsi náhodou příliš složitý datový model? „Přemodelováno“.
- Nemáš v modelu příliš detailů (entit, atributů) bez struktury?
- Nemáš v modelu příliš hlubokou strukturu objektů (entit, atributů)?
- Vyšlo ti při konverzi konceptuální DM – logický DM něco jiného, než jsi chtěl?
- Zdá se ti konceptuální DM nadbytečný?
- Zdá se ti datové modelování zbytečné? Však on někdo pro xml to xsd napíše.

Normální formy datového modelu

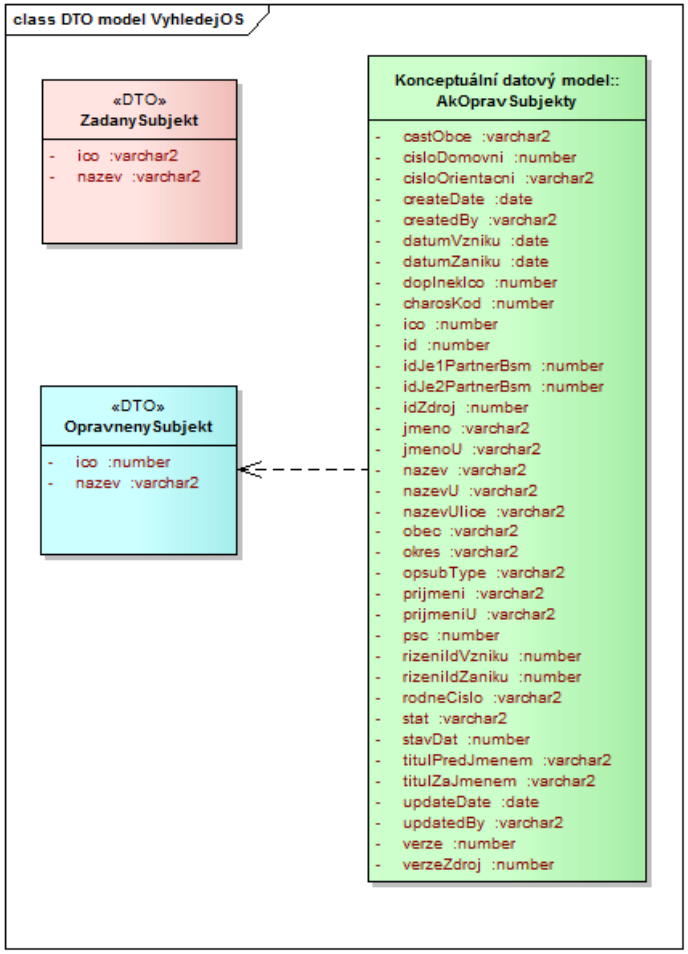


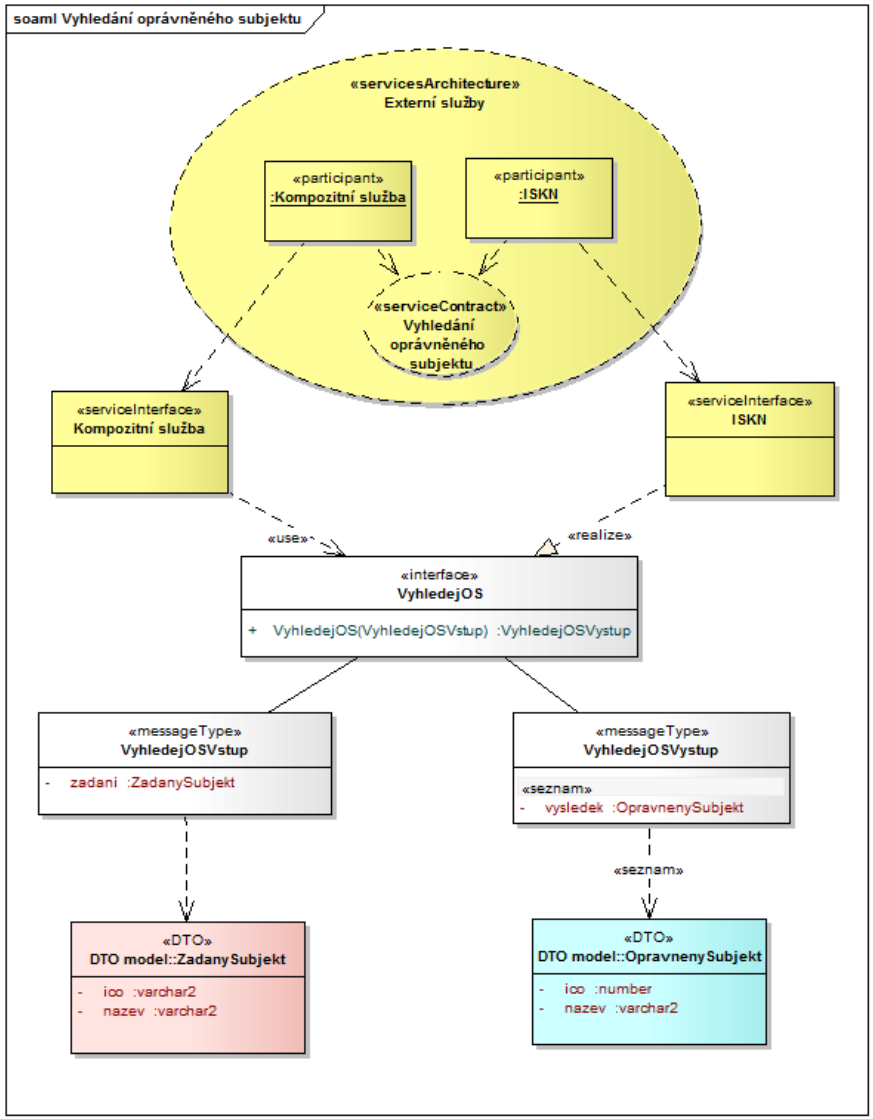
Na závěr: data transfer objekty (DTO) a jejich využití

- Objekt, který přenáší data mezi procesy.
- To se obvykle děje přes vzdálené rozhraní (např. webové služby). Je vhodné snížit náročnost této operace snížením počtu volání služeb -> agregace dat do jednoho datového objektu.
- DTO umožňuje provést denormalizaci dat
- DTO mohou a nemusí být odvozeny z datového modelu (např. uživatelem zadané parametry služby).
- DTO je modelován prvkem typu *class*, stereotypu *DTO*. Formálně se podobá prvku modelu typu entita, může mít však atribut stereotypu *seznam*.
- Použití: například u webových služeb.



Příklad využití DTO při modelování webové služby





Podklady pro další samostudium

- Anatoliy Kybkalo, Datové modelování, Bakalářská práce, Unicorn College, 2013 (http://www.unicorncollege.cz/european-it-center/kybkalo-anatoliy/attachments/PB_Kybkalo_Anatoliy_1-4146-1.pdf, vedoucí práce Ing. Miroslav Žďárský)
- Doc.Ing.Jaroslav Zendulka,CSc.: Databázové systémy a návrh databází http://www.fit.vutbr.cz/study/courses/DSI/public/.cs#Heading2
- http://langer.zam.slu.cz/teaching/db/P02-Konceptualni%20modelovani.pps, autor neuveden