Analytické vzory

© Radek Ošlejšek
Fakulta informatiky MU
oslejsek@fi.muni.cz

Pozadí vzniku vzorů

- Návrh znovupoužitelného softwaru je obtížný
  - nalezení vhodných objektů a abstrakcí
  - pružnost, modularita, elegance $\Rightarrow$ znovupoužití
  - trvá delší dobu, než se objeví, cesta pokusů a omylů
- Úspěšné návrhy přesto existují
  - vykazují opakující se třídy a struktury
- Jak popsat tyto opakující se struktury ?

Softwarové vzory

- malé
  - kódovací vzory: Smalltalk Best Practice Patterns
  - refaktorizace
- střední
  - návrhové vzory
  - analytické vzory
- velké
  - kostry, soustavy
  - architektonické vzory

Původ vzorů

- architekt Christopher Alexander
  - pojem „vzor" cca 1977-1979)
- Kent Beck and Ward Cunningham, Textronix, OOPSLA'87
  - použil Alexanderovy ideje o „vzorech" pro návrh Smalltalk GUI
- Erich Gamma, Ph.D. teze, 1988-1991
- James Coplien, *Advanced C++ Idioms book*, 1989-1991
- Gamma, Helm, Johnson, Vlissides ("Gang of Four“ - GoF) *Design Patterns: Elements of Reusable Object-Oriented Software*, 1991-1994
- PLoP konference a knihy, 1994-....
- Buschmann, Meunier, Rohnert, Sommerland, Stal, *Pattern – Oriented Software Architecture: A System of Patterns* (“POSA”)
- Martin Fowler – Analysis Patterns , 1997
- ...

Ch. Alexander – jazyk vzorů

- Co je to, co dává budově *Kvalitu*?
  - svoboda, život, pohodlí, harmonie
- Vzor: řešení problému v daném kontextu
  - *vstupní přechod*
  - *gradient soukromí*
  - *světlo na dvou stranách každého pokoje*
- Propojené vzory ⇒ *Jazyk vzorů*
  - 2534 vzorů, hrubé až po jemnozrnné

The Timeless Way of Building (1979)

A Pattern Language (1977)

Analytické vzory

- Analysis Patterns, Martin Fowler, AW 1997
- Data Modeling Patterns, David Hay
- Java Modeling in Color with UML, Peter Coad
- The Data Model Resource Book (two volumes), Len Silverston

- Modeling principles:
  - *Patterns are a starting point, not a destination.*
  - *Models are not right or wrong, they are more or less useful.*

Další vzory

- Existuje celá řada dalších vzorů pro různé oblasti
  - GoF design pattern
  - Coad, North, Mayfield, *Object Models: Strategies, Patterns, and Applications*. Second Edition. Prentice Hall, 1997.
  - Anti-patterns
  - Java J2EE patterns
  - Real-time design patterns
  - ...

Analytické vzory Fowler

- 9 souborů vzorů, každý soubor se týká určité doménové oblasti:
  - Accountability – organizační struktura a odpovědnosti osob
  - Observations and Measurements – záznamy údajů a měření
  - Observations for Corporate Finance – analýza složitých finančních vztahů a výsledků ve firmách
  - Refering to Objects – identifikace objektů
  - Inventory and Accounting – sledování toků peněz a inventury
  - Planning – plánování akcí a protokoly pro opakované plány
  - Trading – koupě a prodej produktů
  - Derivative Contracts – ceny odvozené od cen jiných subjektů apod.
  - Trading Packages – „trading“ pro velké systémy
- Každý soubor obsahuje evoluční řadu vzorů, od základních přes jejich kombinace až ke komplexním vzorům
- Celkem 65 analytických vzorů a 21 pomocných
- Analytické vzpory se občas odkazují na návrhové vzory

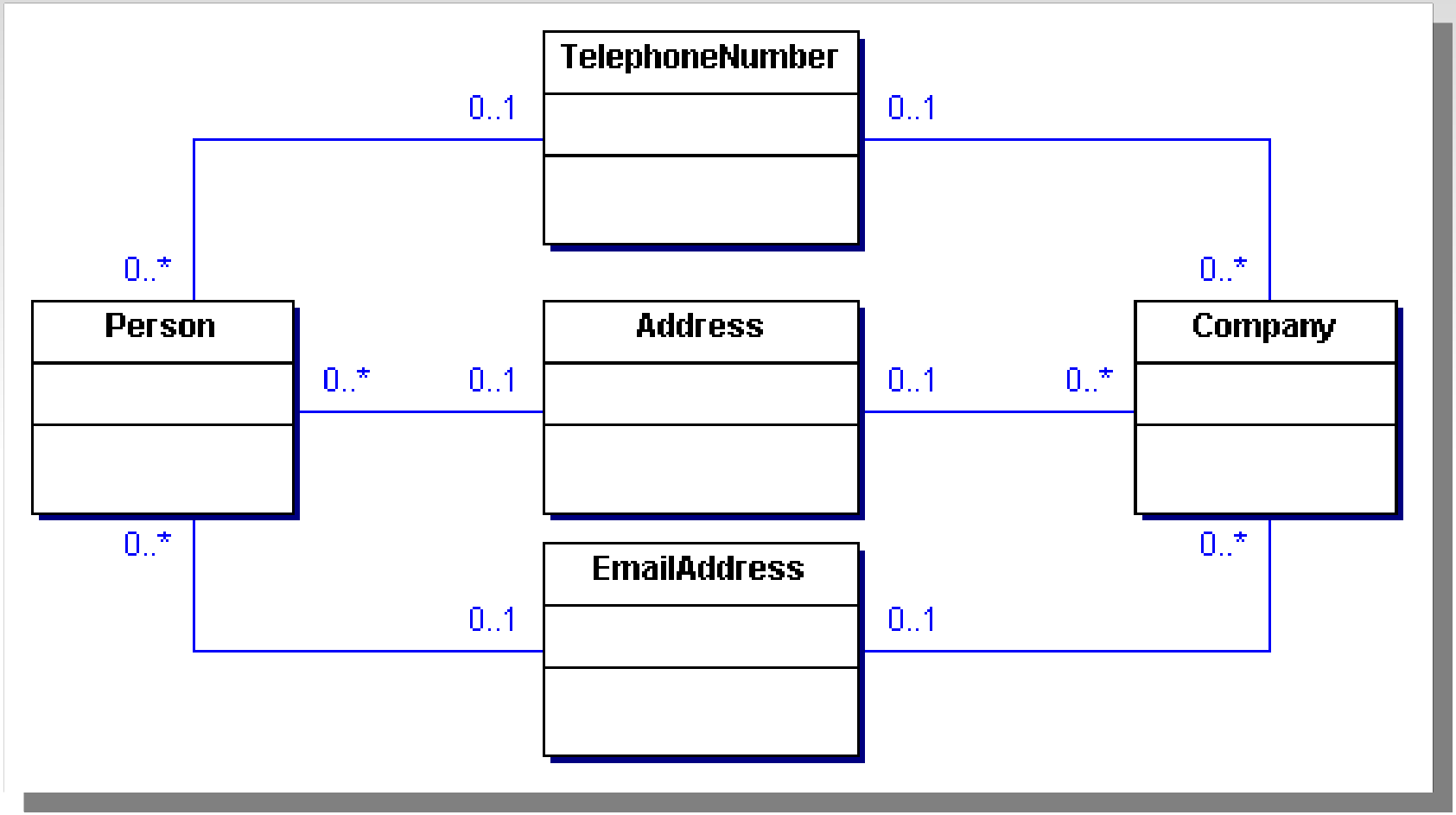
Soubor vzorů Accountability

- Modelování zodpovědností mezi osobami a organizacemi.
  - Organizační struktura společnosti
  - Hierarchie řízení (nadřízenost-podřízenost mezi zaměstnanci)
  - ...
- Často se používají v podnikových informačních systémech

Soubor vzorů Accountability

- **Party** – základní vzor pro řešení odpovědností
- **Organization Hierarchies** – jednoduchá organizační struktura
- **Organization Structure** – složitější organizační struktura
- **Accountibility** – kombinace Party a Organization Structure
- **Accountibility Knowledge Level** – typy zodpovědností a omezení
- **Party Type Generalization** – hierarchie typů účastníků
- **Hierarchic Accountibility** – hierarchie zodpovědností
- **Operating Scopes** – rozsahy zodpovědností, místa apod.
- **Post** – odpovědnosti delegované na pracovní pozice namísto konkrétních osob

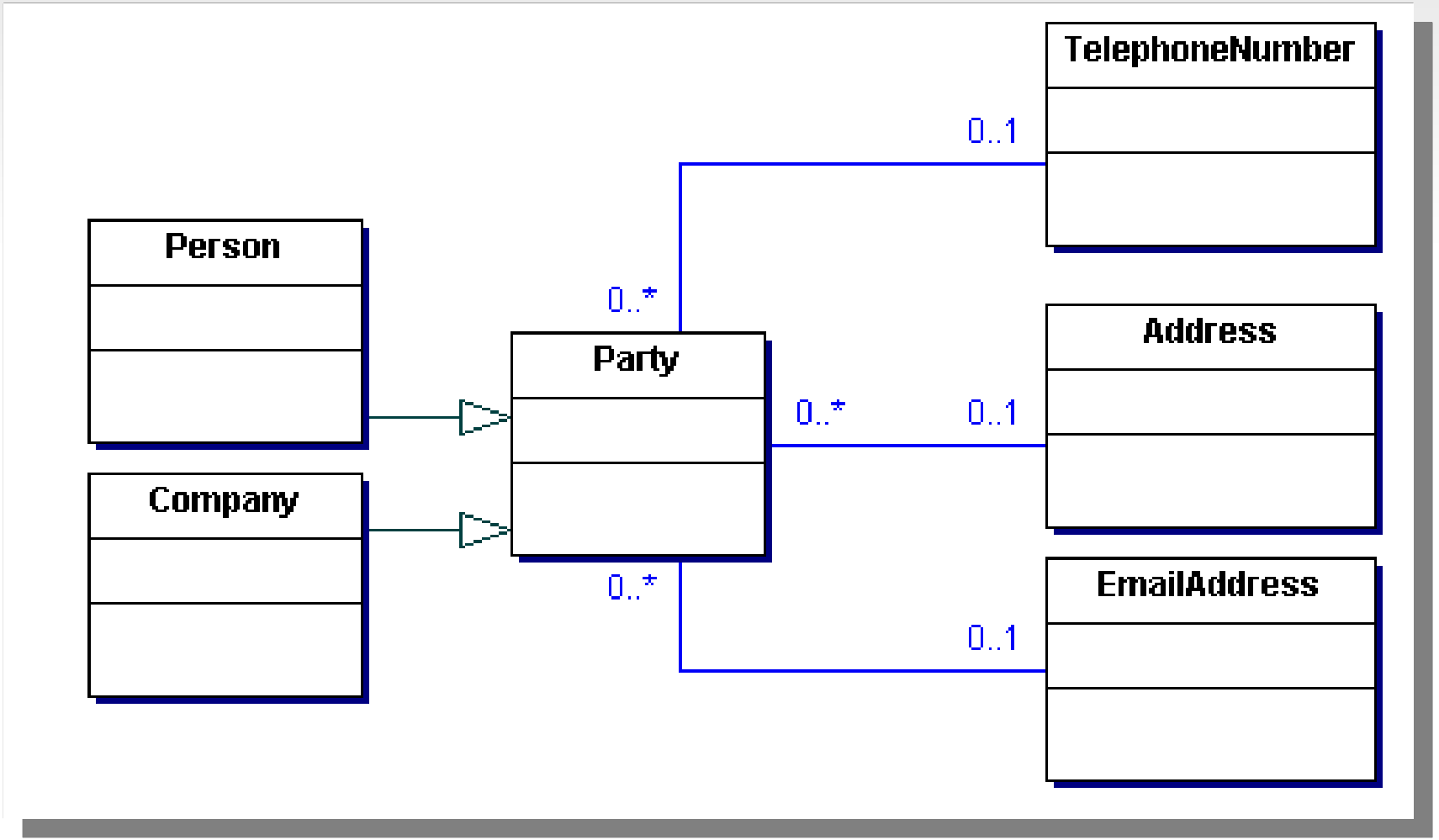
Party – motivace



příklad modelu adresáře (address book),
typický výsledek modelu problémové oblasti

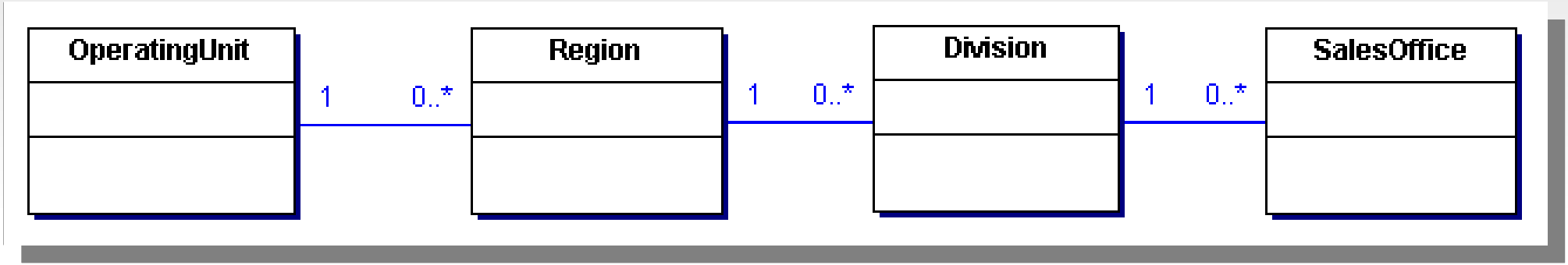
Party – vzor

- Strana, účastník
- Mnohé role, které obvykle hrají lidé, jsou někdy také hrány organizačními jednotkami.
- Party = jednotné pojmenování pro tyto role



Organization Hierarchies – motivace

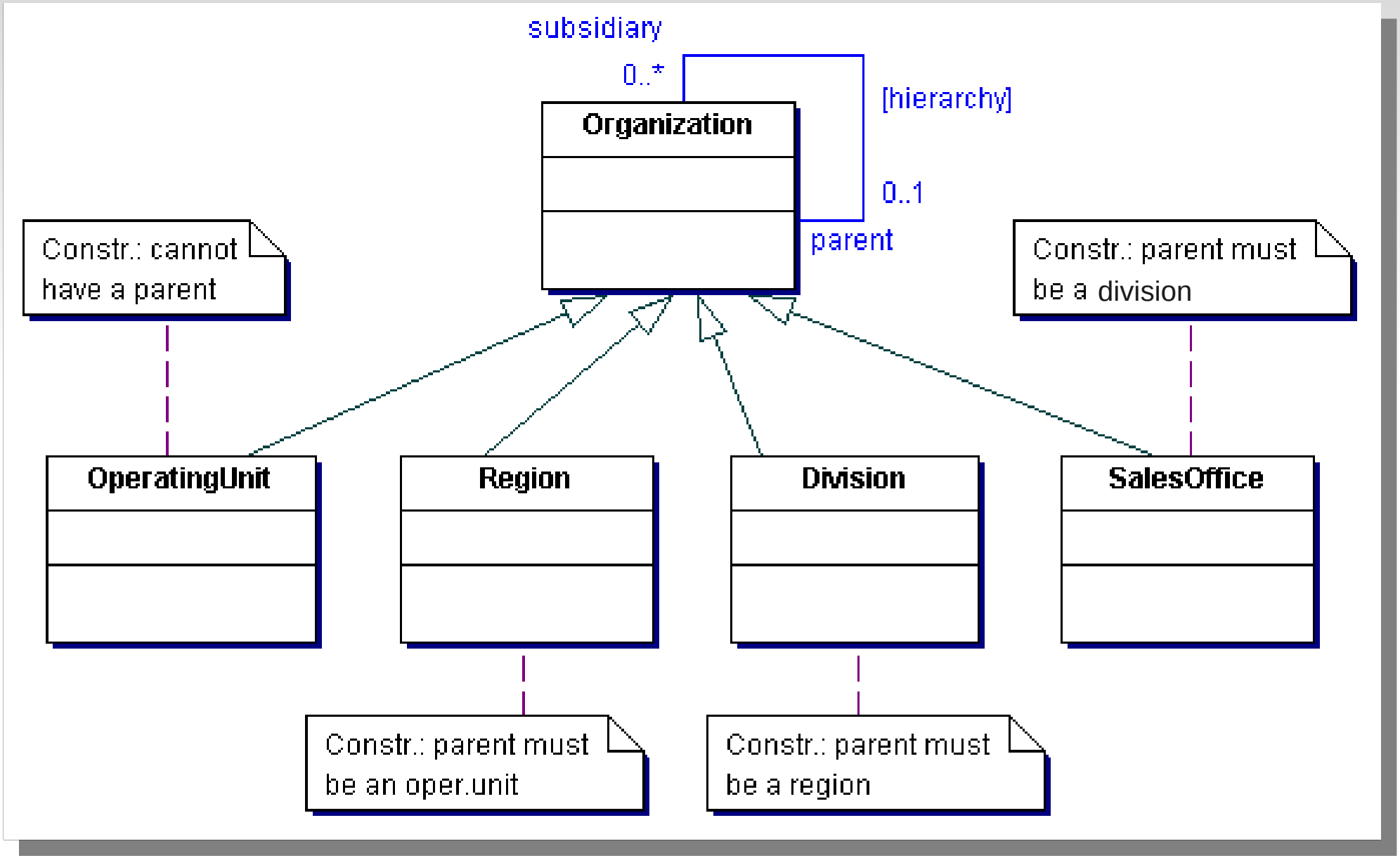
- Organizační hierarchie
- Organizace mají obvykle hierarchickou strukturu



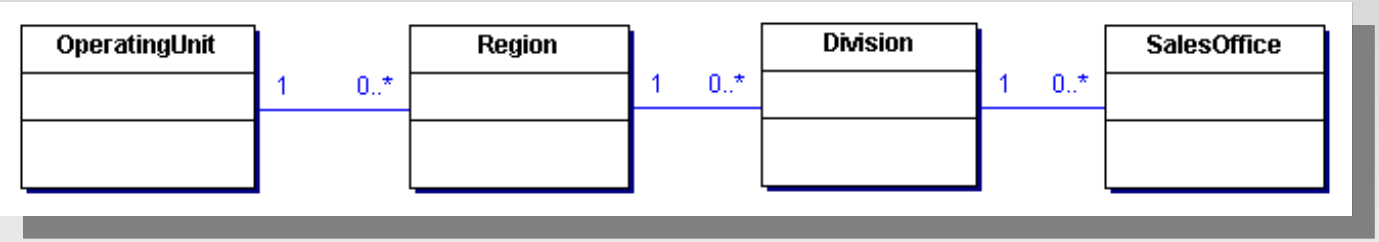
příklad organizační struktury nadnárodní společnosti

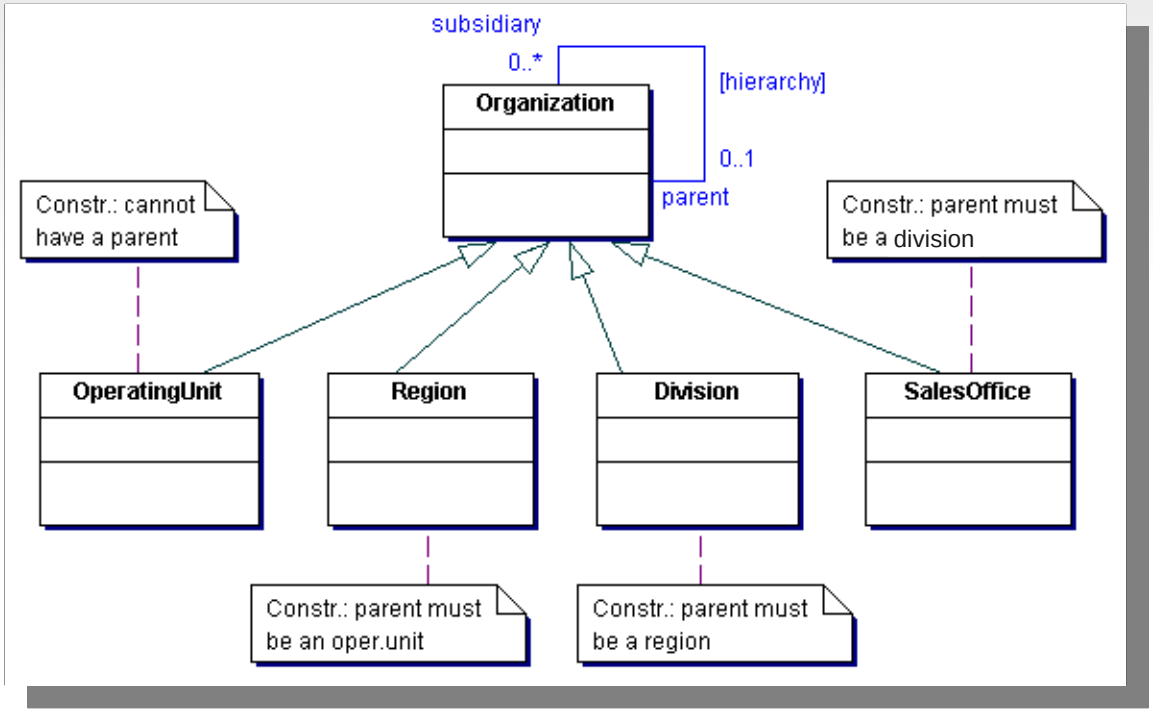
- málo flexibilní: zrušení např. regionu z důvodu redukce stupňů řízení vede ke změně modelu
- nejde znovupoužít pro jiné organizace s jinou strukturou nebo názvy organizačních jednotek

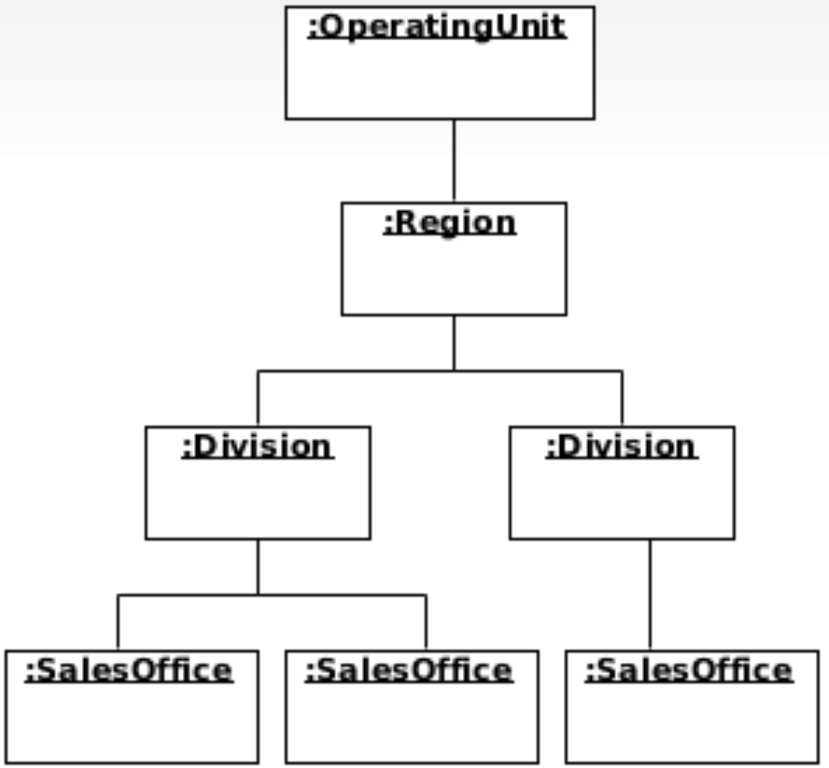
Organization Hierarchies – vzor



Organization Hierarchies – rozdíly

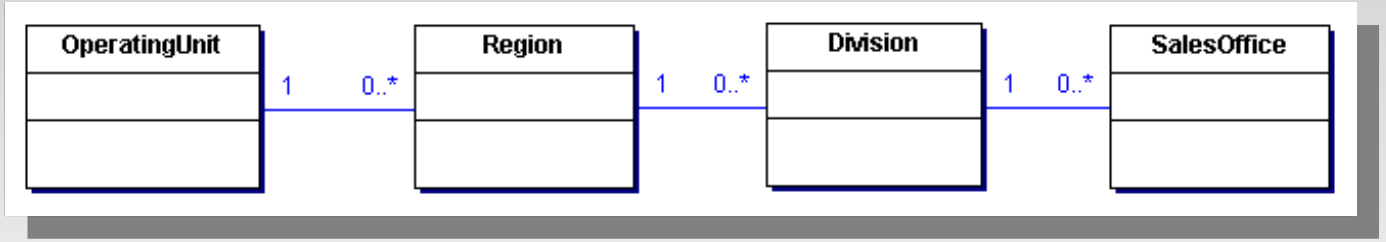


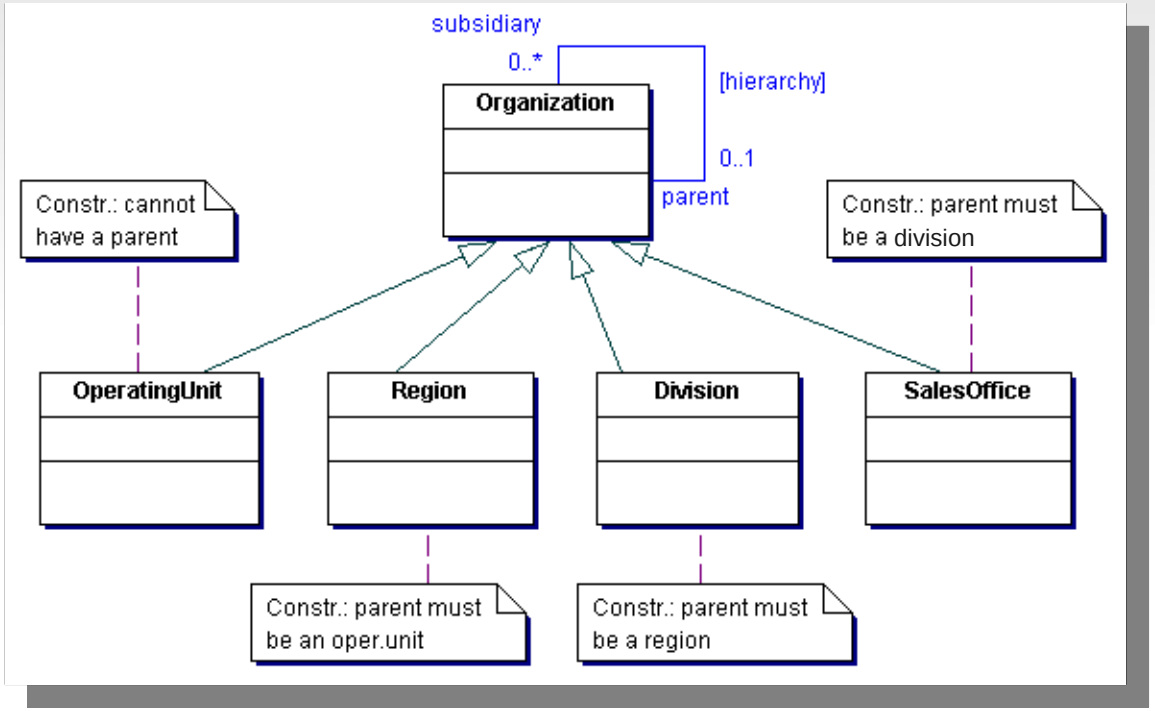




- Q: Kterému modelu odpovídá diagram objektů?
- A: Oběma

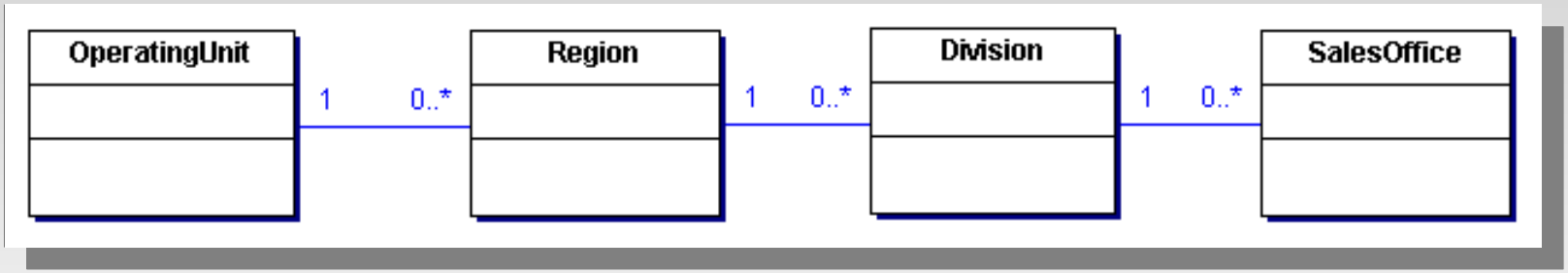
Organization Hierarchies – rozdíly (pokr.)





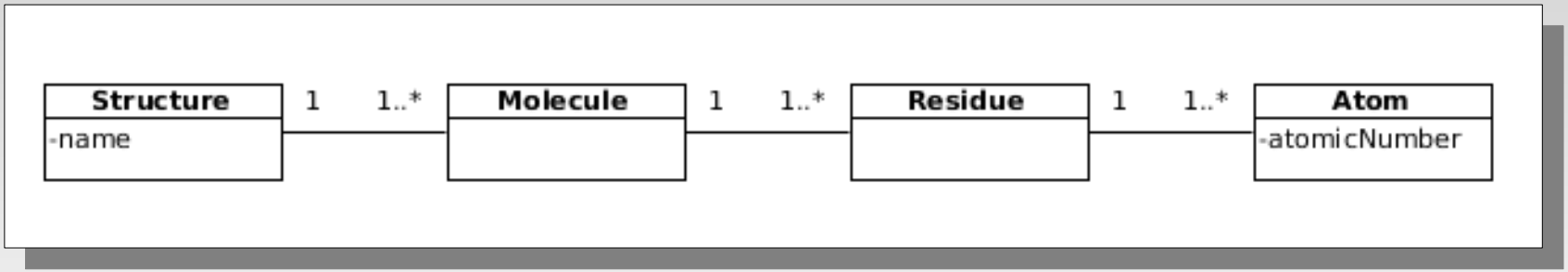
- Q: Jak snadné je změnit organizační strukturu?
- A: bez vzoru => změna modelu, se vzorem => přidání podtřídy a/nebo změna omezení

Příklad aplikace vzoru

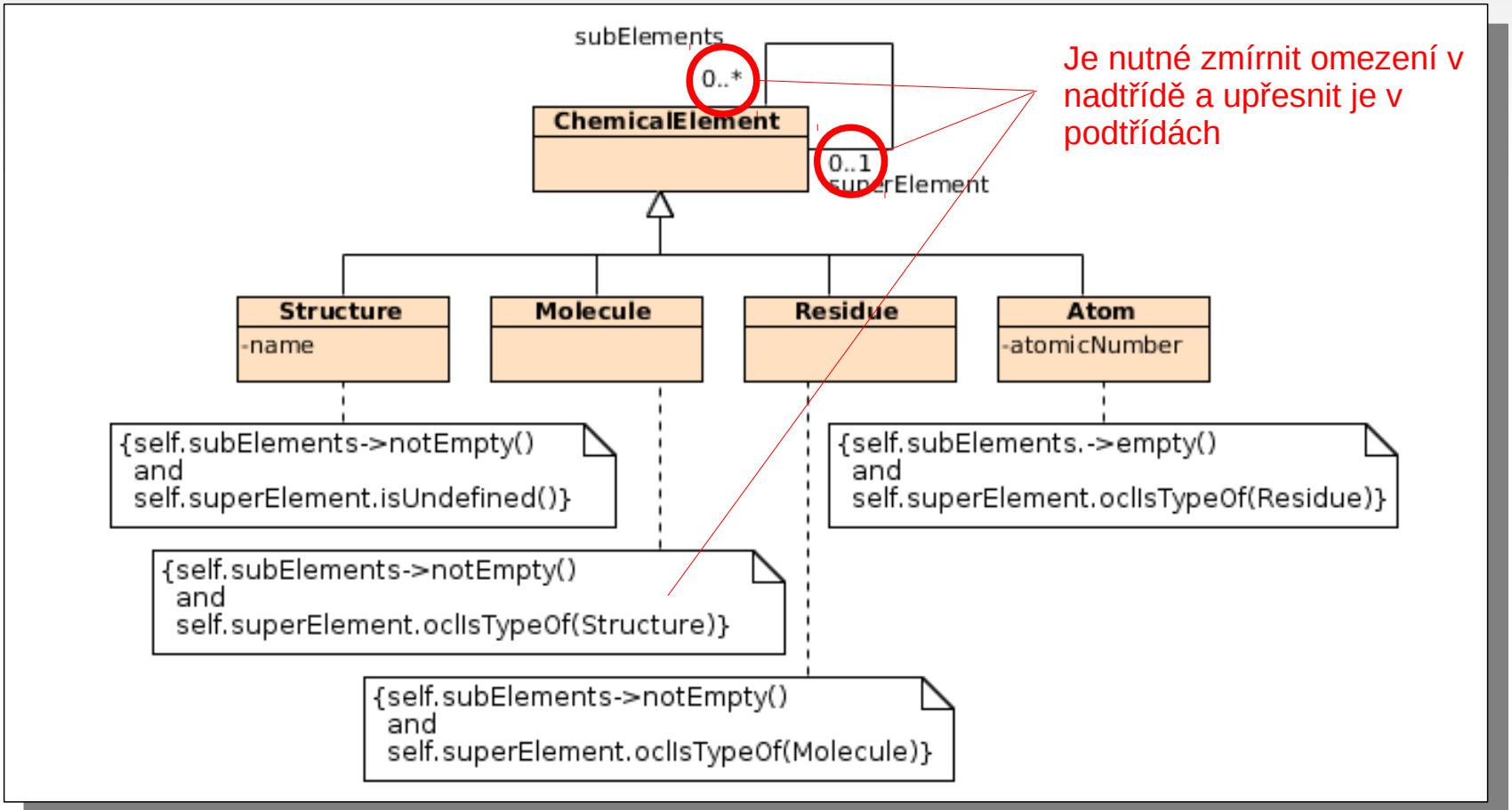


Kde už jsem podobný model viděl?

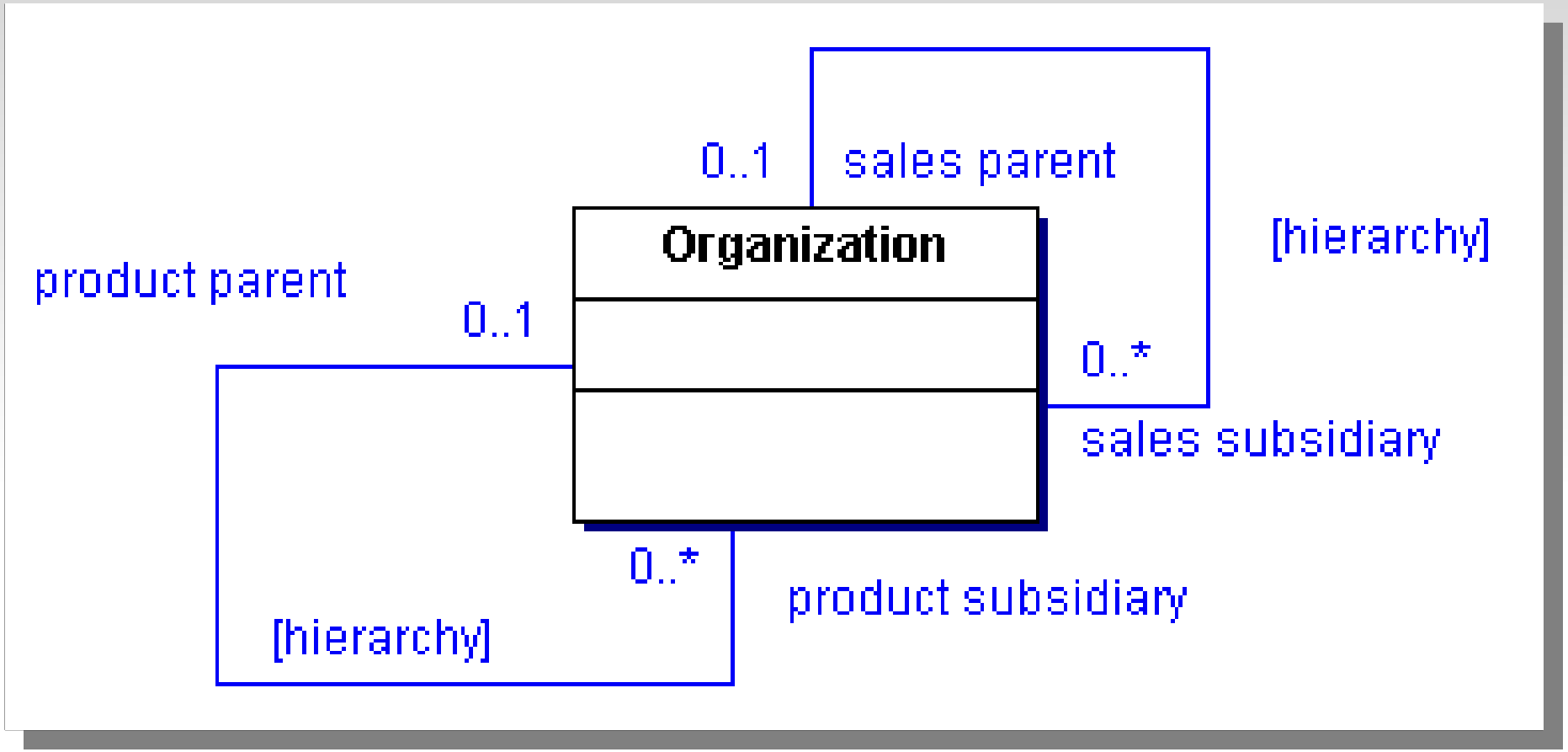
Příklad aplikace vzoru (pokr.)



Doménový model chemických struktur z minulé přednášky!



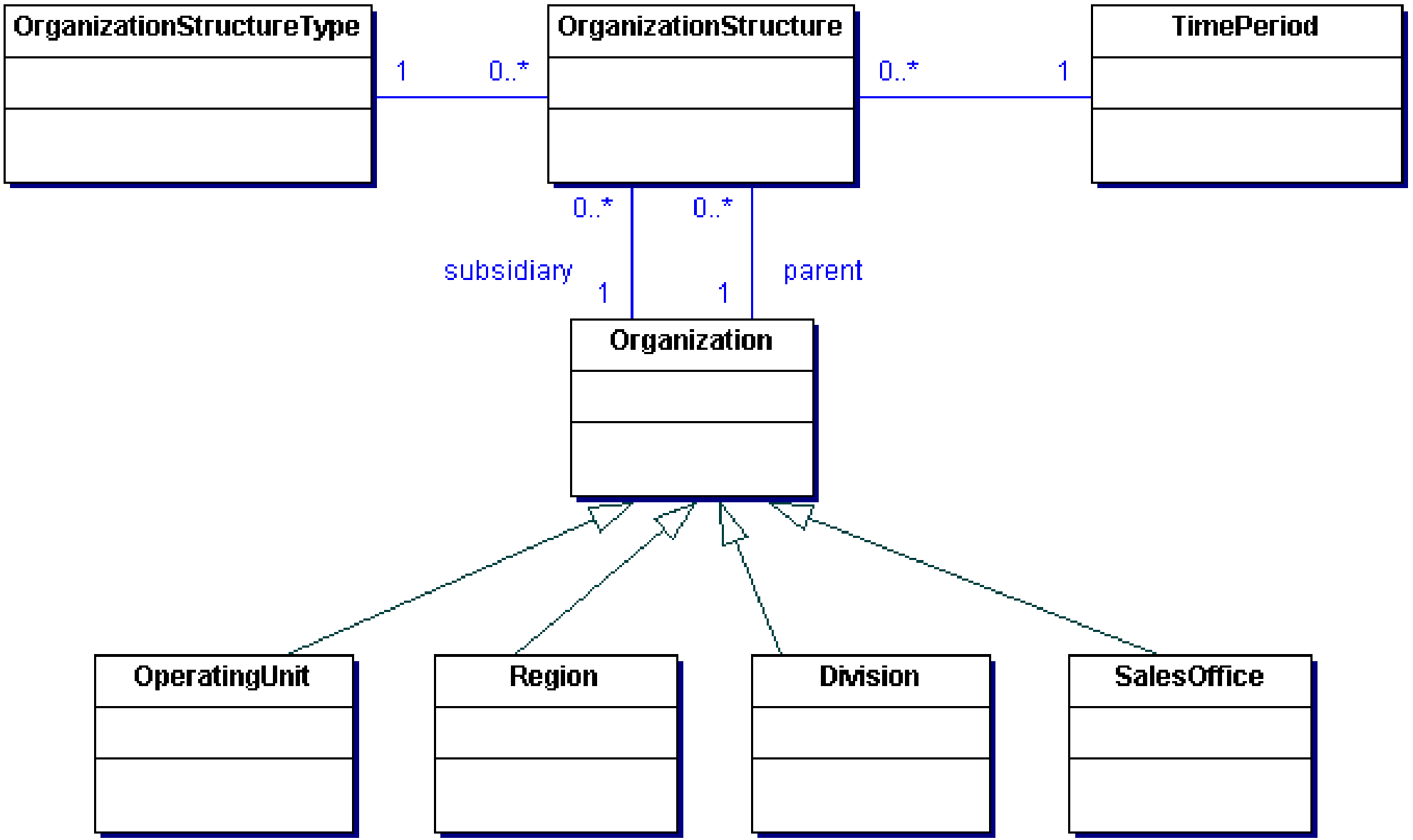
Organization Hierarchies – více hierarchií



příklad dvou nezávislých organizačních hierarchií (prodej a servis).

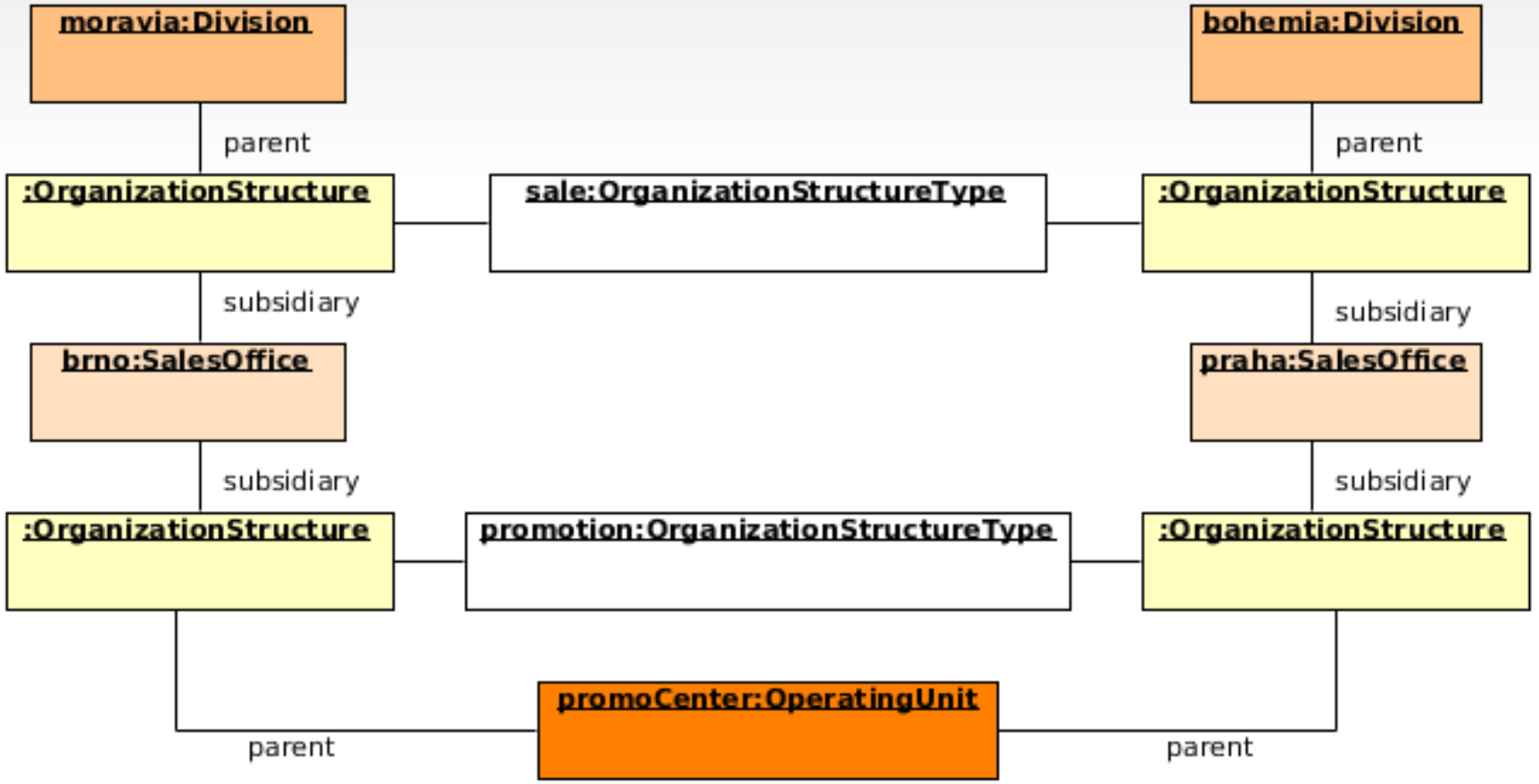
- Problém: pokud bude paralelních hierarchií více, snadno se model dostane mimo naši kontrolu (představte si podtřídy, které v příkladu chybí).

Organization Structure – vzor



Organization Structure – příklad

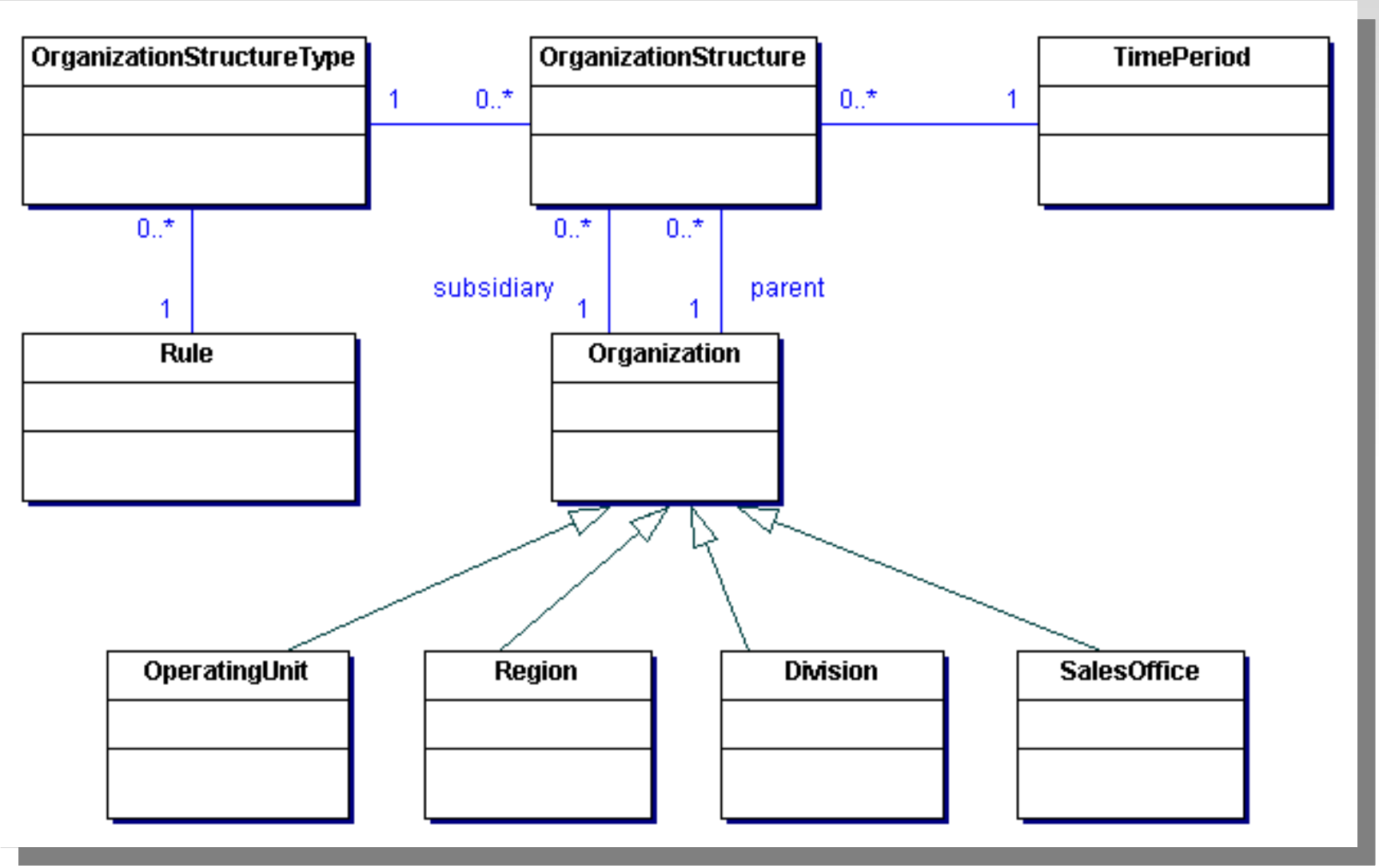
- Prodejce má prodejní místa v Brně a Praze.
- Z hlediska prodeje je brněnská pobočka podřízena moravské divizi, pražská pobočka české divizi
- Z hlediska propagace a marketingu jsou obě pobočky podřízeny přímo centrále



Organization Structure – vlastnosti

- vhodné pro modelování více organizačních hierarchií
- přidání nové hierarchie = nová instance třídy *OrganizationStructureType*
  - typ organizační struktury není modelován jako atribut třídy *OrganizationStructure*; důvod viz rozšíření o třídu *Rule*
- *TimePeriod* umožňuje zaznamenat změny v organizační struktuře v čase
- Omezení na organizační strukturu jsou definovány ve třídě *OrganizationStructure*
  - => přidání nové organizační struktury by mohlo vést ke změnám kódu
  - řešení: přidání třídy *Rule*

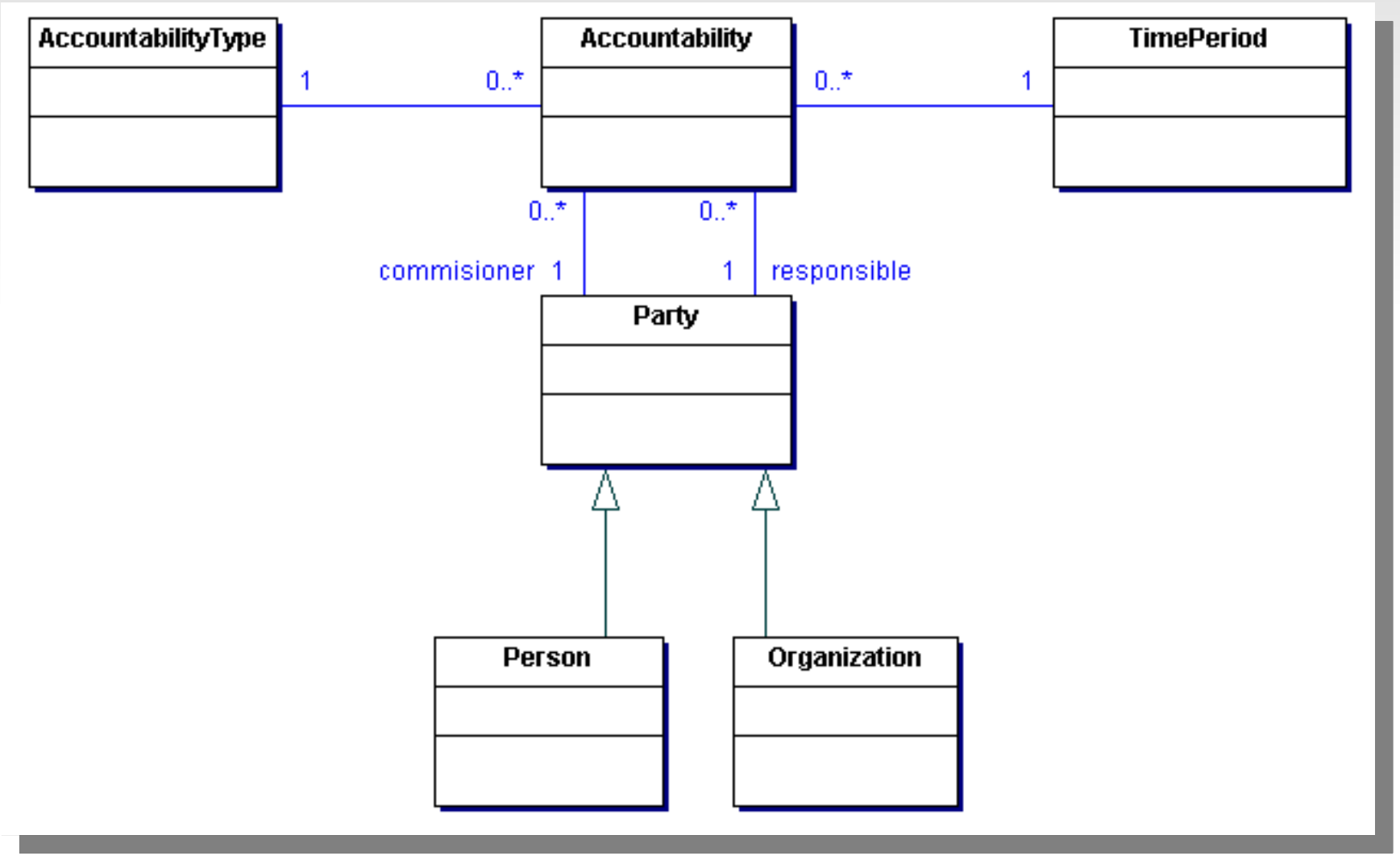
Organization Structure – rozšíření o Rule



- Pozn: pokud se málo mění typ org. struktury, ale často se přidávají podtypy organizace, pak to vede k častým změnám ve třídě *Rule.* V těchto případech proto bývá výhodnější asociovat třídu *Rule* přímo s podtřídami třídy *Organization.*

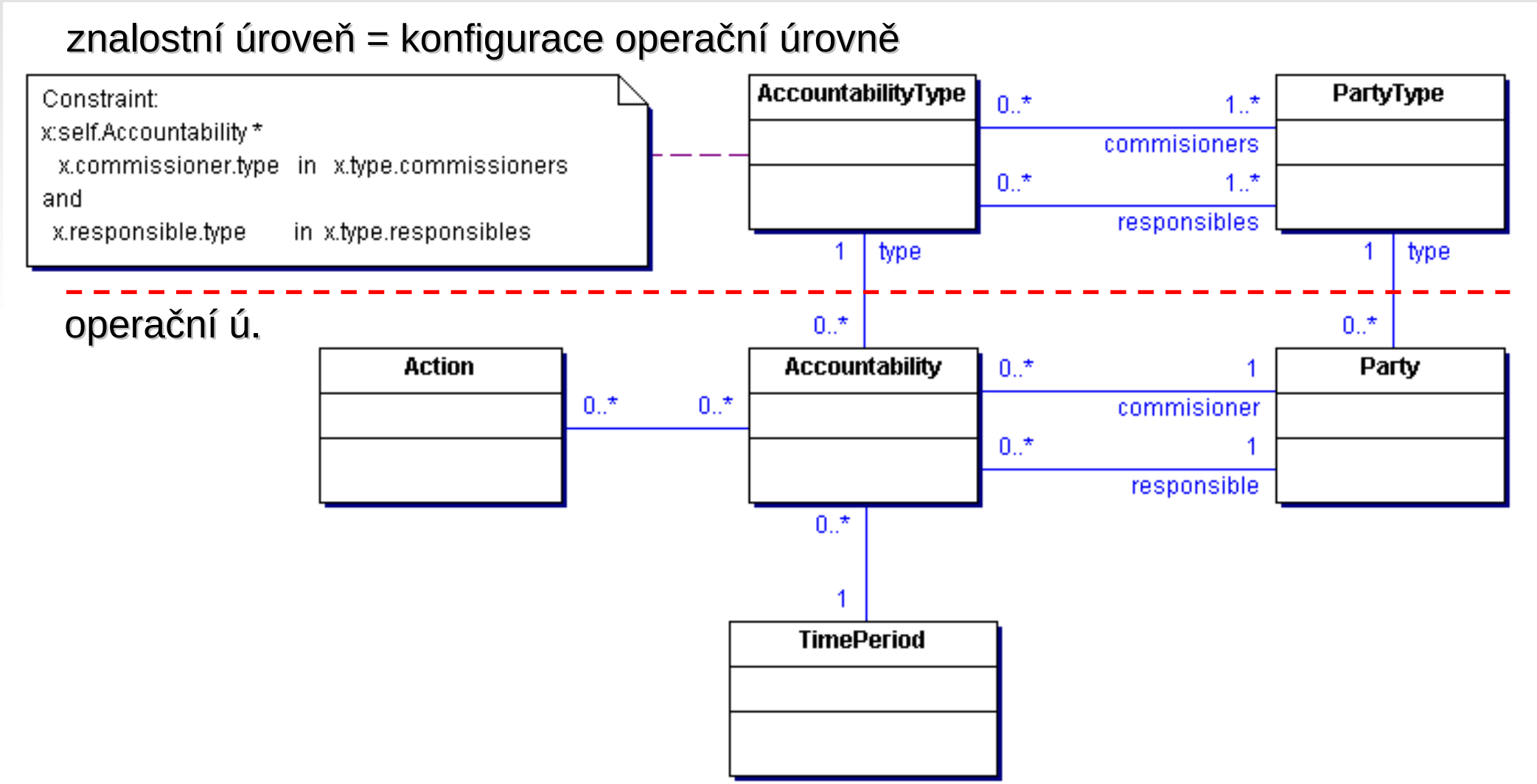
Accountibility – vzor

- Motivace: Vzor Organization Structure modeluje vztahy mezi organizacemi v čase v rámci definovaných pravidel. Ale i lidé mohou mít vztahy s organizacemi a ostatními lidmi v čase v rámci definovaných pravidel => Spojení vzorů Organization Structure a Party.



Accountibility Knowledge Level – vzor

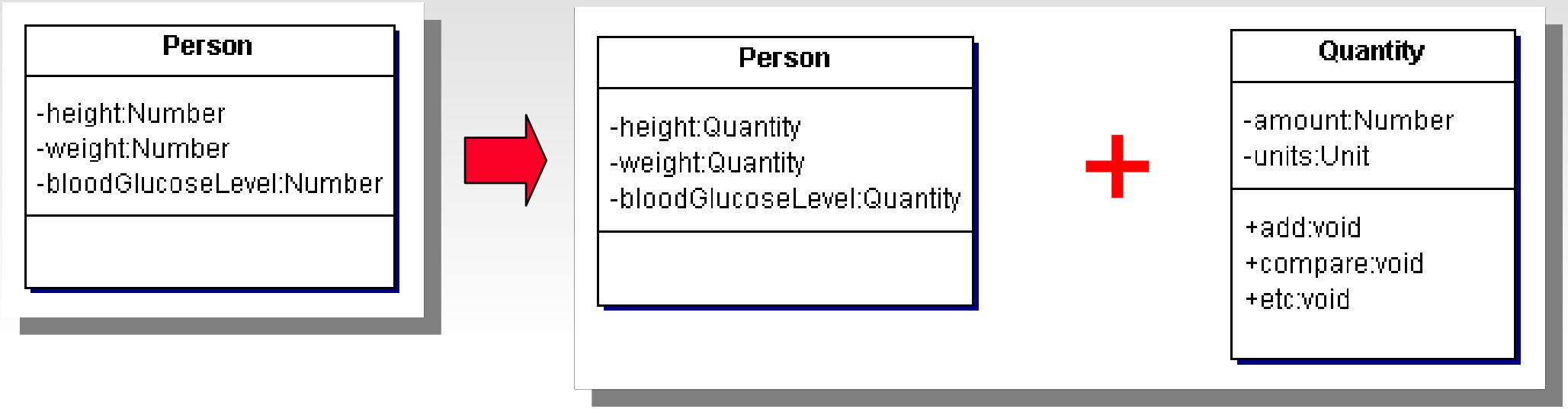
- řízení složitosti mezi typy zodpovědností a typy účastníků



Soubor vzorů Observations and Measurements

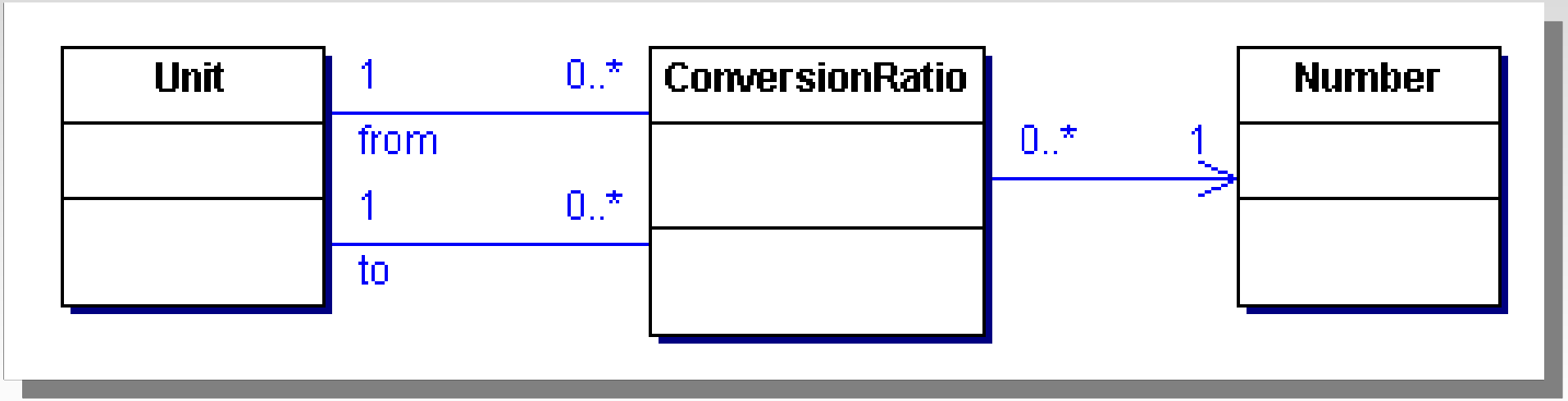
- Uchovávání a zpracování kvantitativních a kvalitativních informací
  - Př. pacient: výška, váha, kvalita zraku, tlak, ...
- Sofistikovanější alternativa k atributům
- Vzory:
  - Quantity – hodnoty a jejich jednotky
  - Conversion Ratio – převodní poměry
  - Compound Units – složené jednotky (např. km/h)
  - Measurement – měření a zaznamenání hodnot
  - Observation – obecná pozorování a jejich záznam
  - Subtyping Observation Concepts
  - Protocol – protokoly záznamů měření
  - ...

Quantity – vzor



Otázka: V jakých jednotkách je váha?
Princip: Pokud potřebujeme kombinovat atributy s různými jednotkami, tak je vhodné přesunout je do nového fundamentálního typu poskytujícího hodnotu i použité jednotky a navíc také operace mezi hodnotami ve stejných jednotkách.

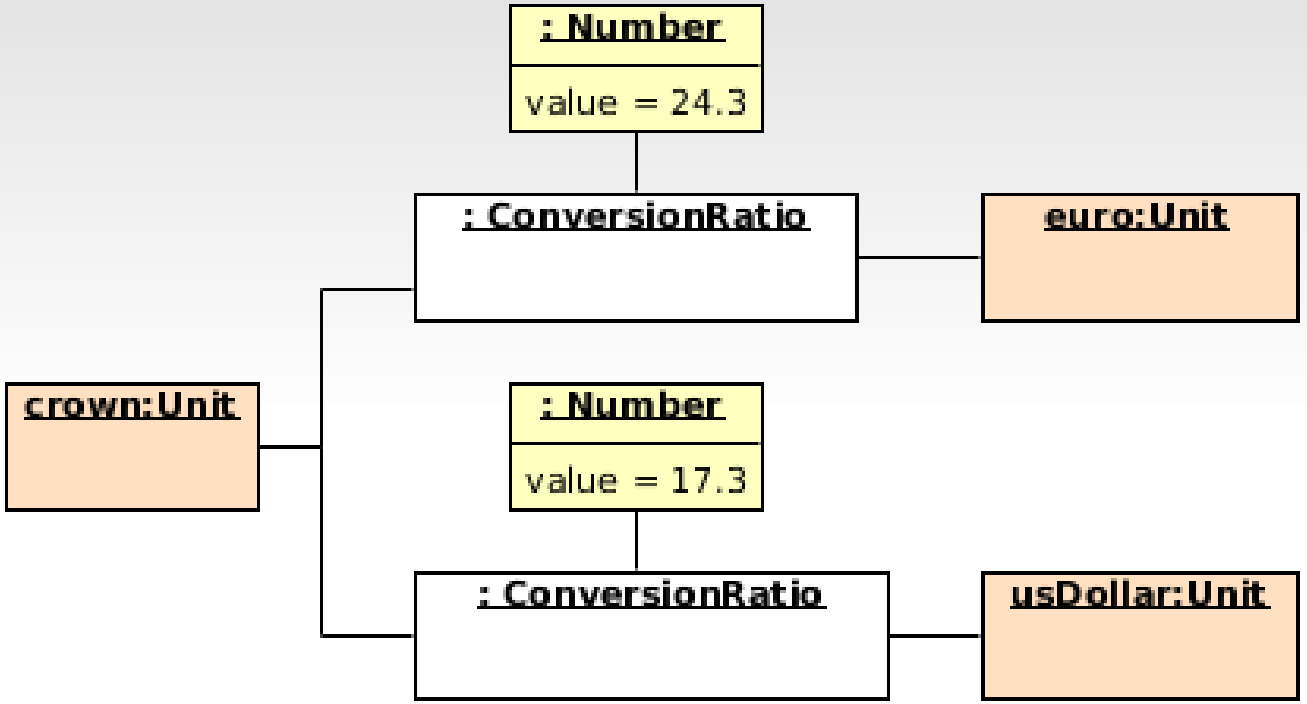
Convertion Ratio – vzor



- Vhodné pro jednoduchý převod jednotek
- Problém: Jak převést stupně Celsia na Fahrenheita nebo dny a měsíce? Musí se zapojit vzor *Individual Instance Method*.
- Problém: Jak převést složené jednotky? Viz následující vzor.

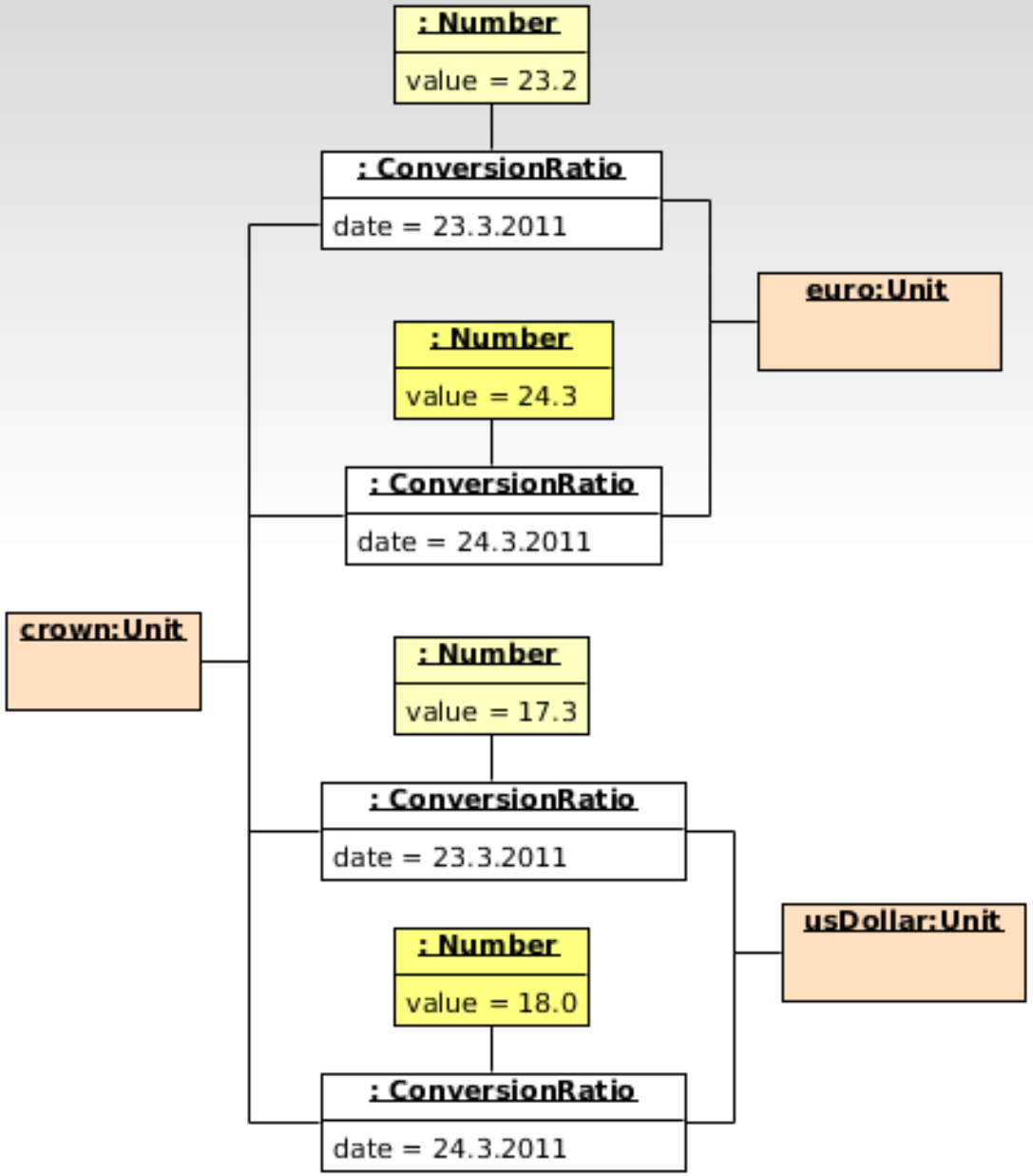
- **Úkol:** Sestavte kurzovní lístek.

Convertion Ratio – kurzovní lístek



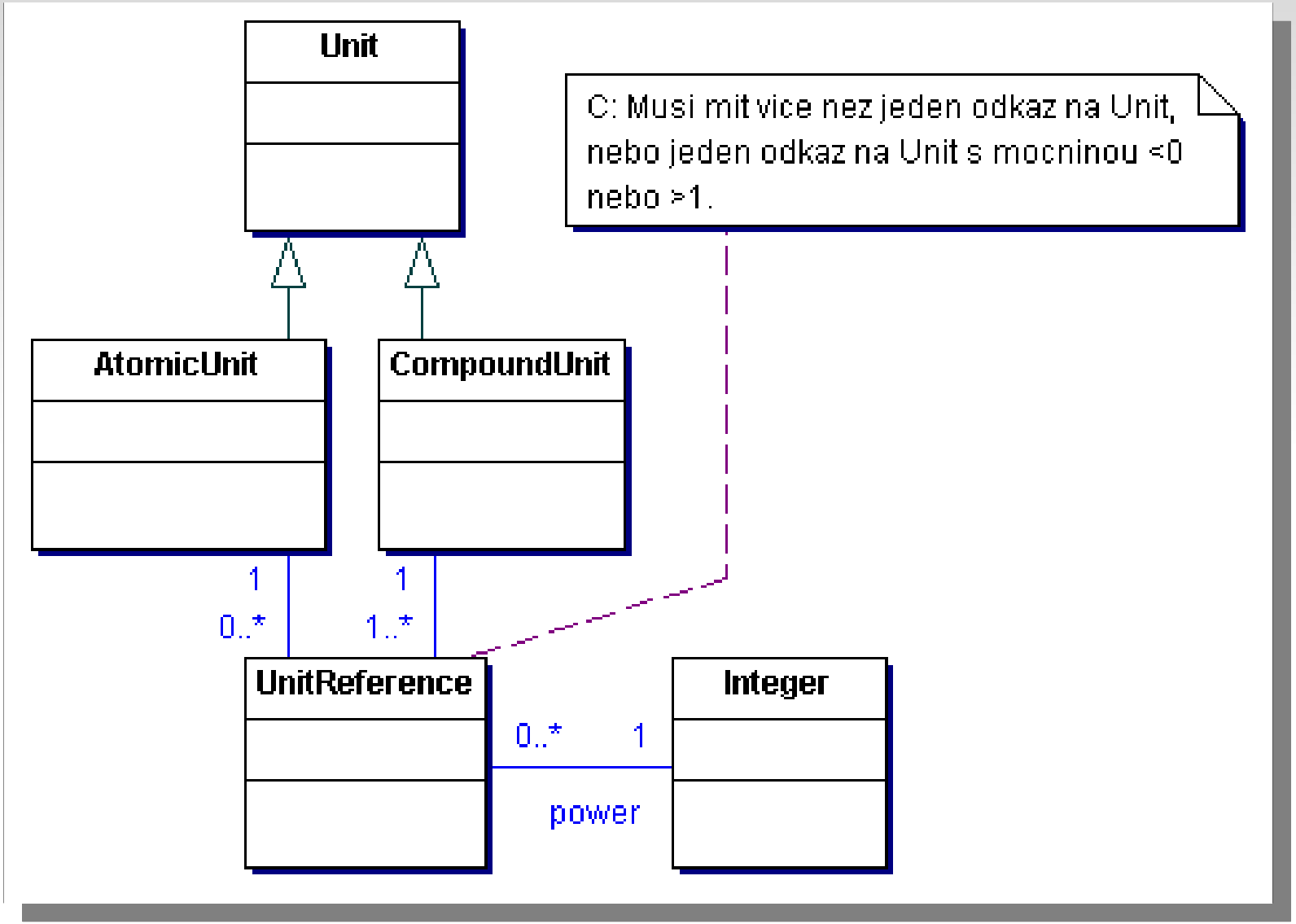
- **Úkol:** Sestavte kurzovní lístek s historií kurzů

Convertion Ratio – kurzovní lístek s hitorií



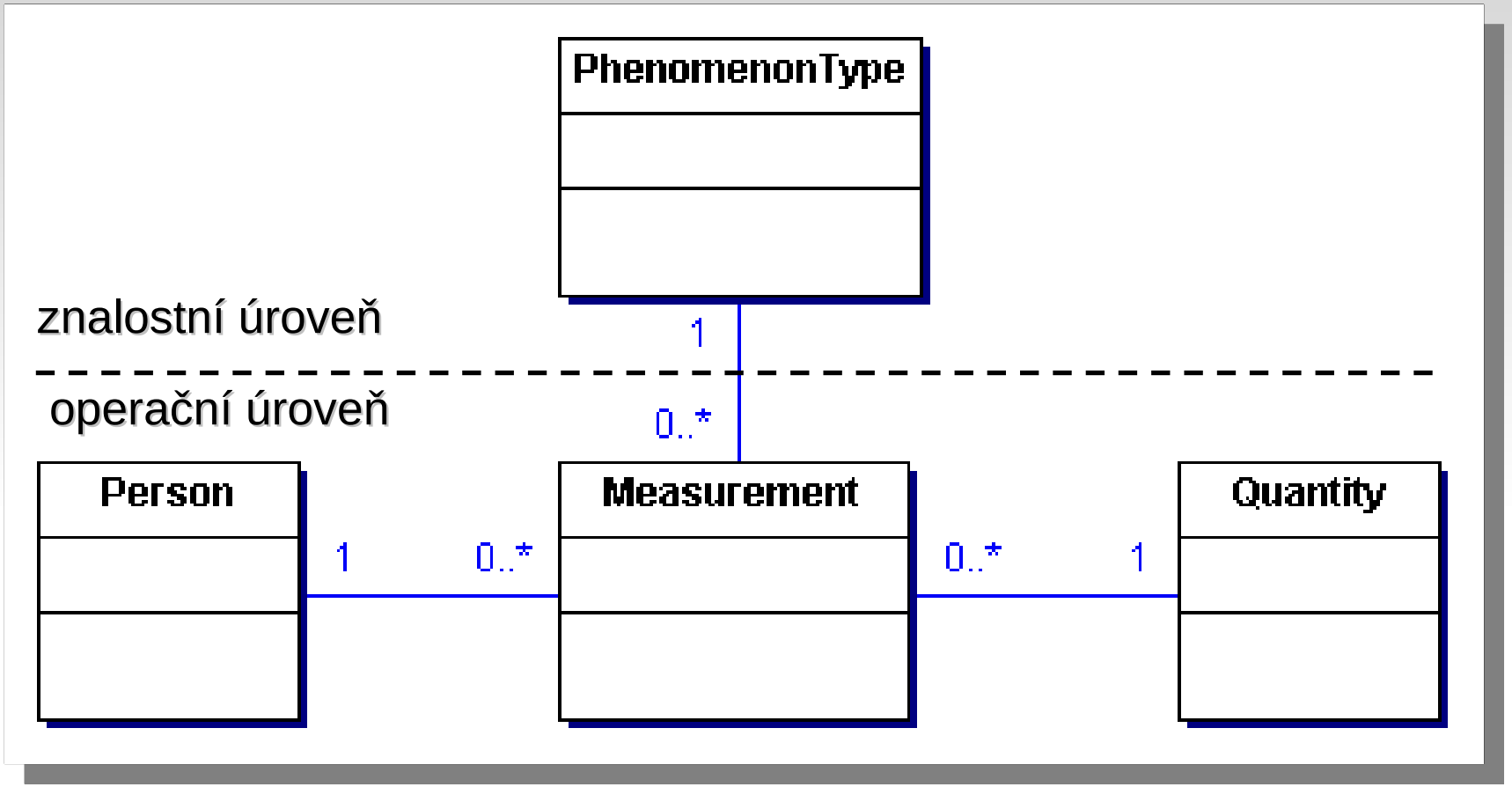
Compound Units – vzor

Použití:
zrychlení apod.
km hod^{-1}, Jcm^{-2}



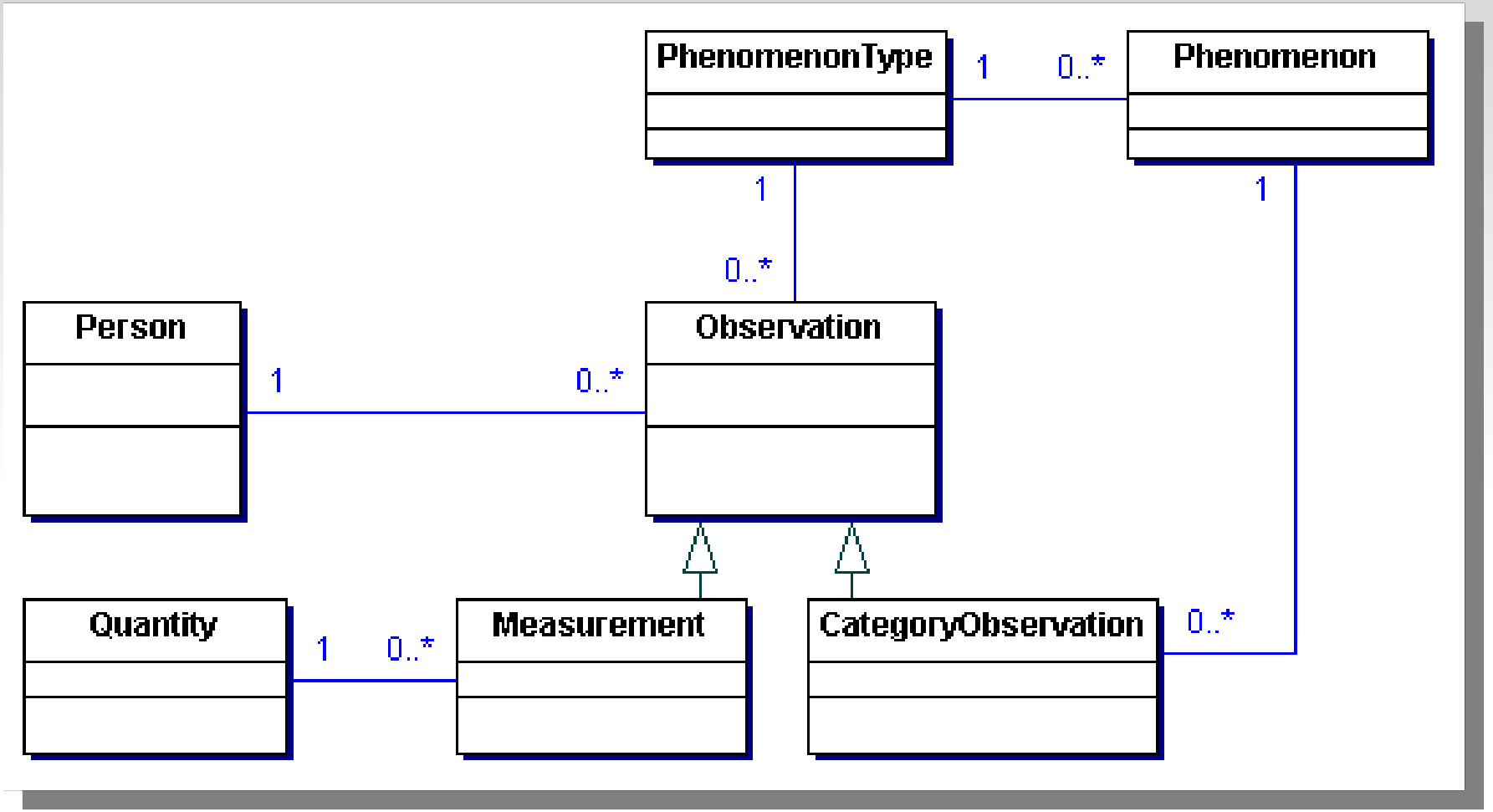
Úkol: rychlost v uzlech, míle/hod, km/hod, m/sec

Measurement – vzor



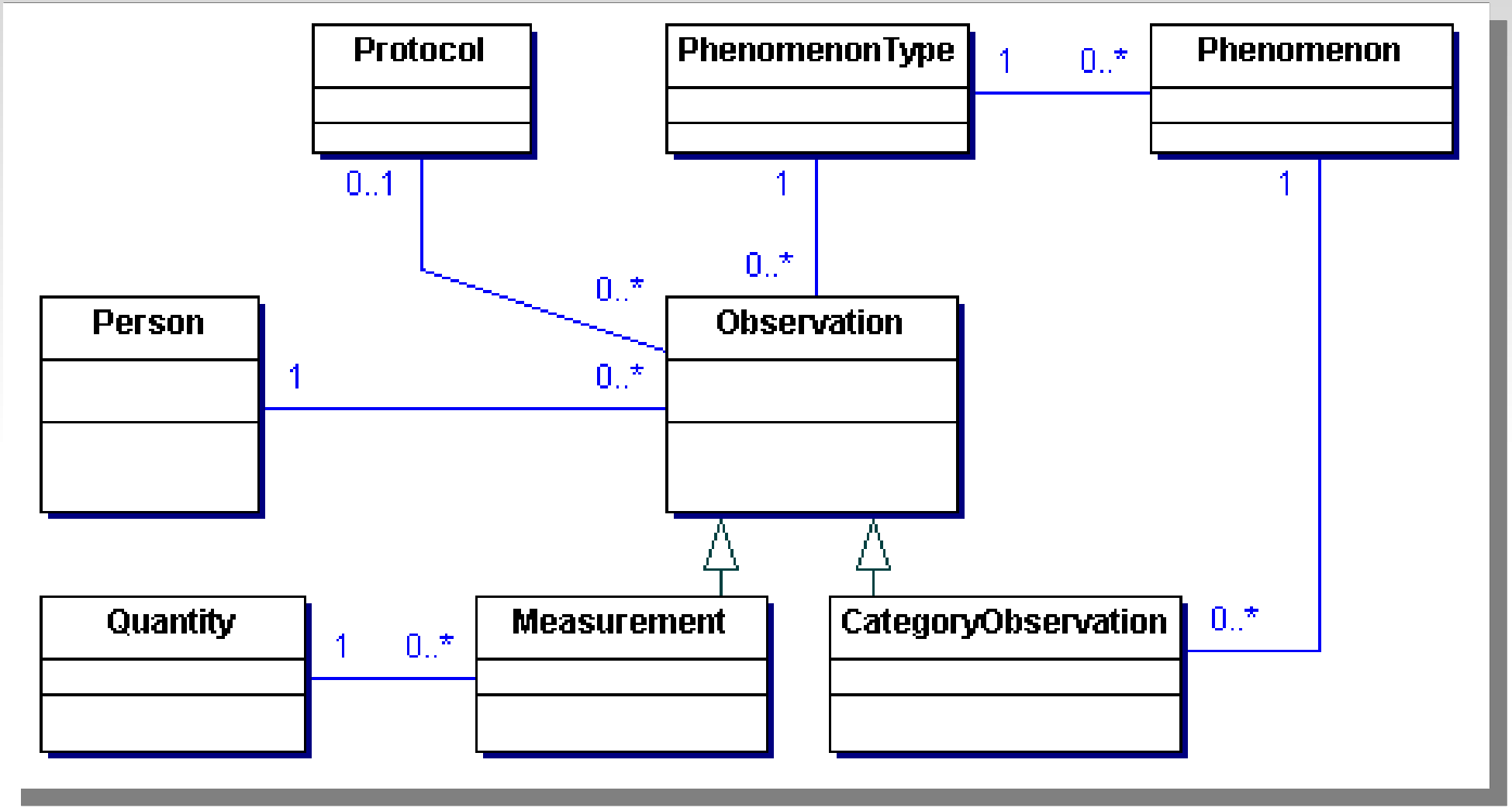
- PhenomenonType = jednotky, které měříme (výška, hmotnost, tlak, …)
- Operační úroveň obsahuje koncepty, které se mění každodenně. Jejich konfigurace je vymezena na znalostní úrovni, která se mění méně často

Observation – vzor



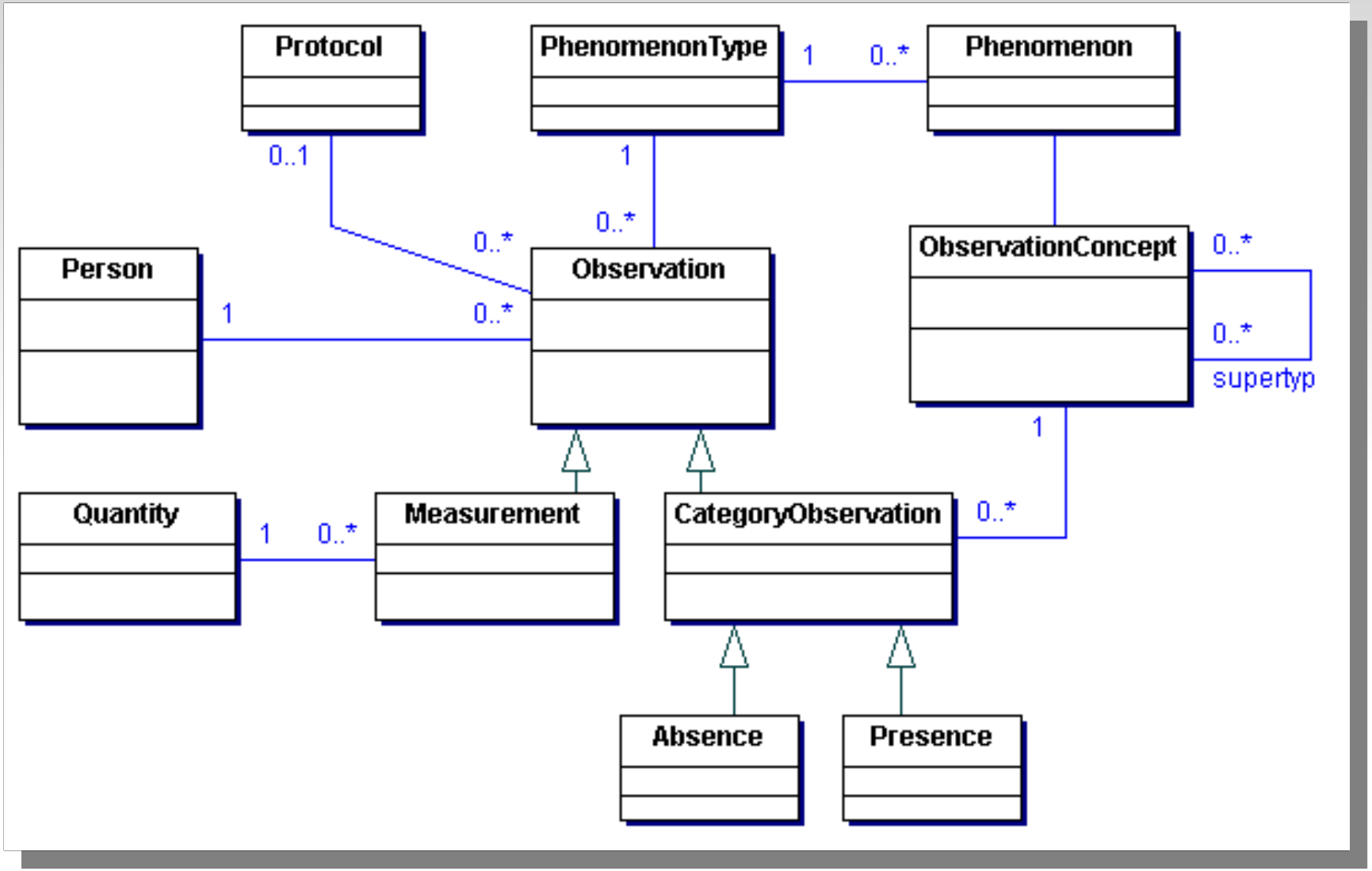
- Záznamy nenumerických dat (pohlaví, krevní skupina, …)
- Úkol: Jak namodelovat nízkou hladinu oleje v autě?
- Řešení: *PhenomenonType* = *„hladina oleje“*, *Phenomenon* = *„příliš velká“*, *„OK“*, *„nízká“*. *Observation* propojí auto (instance „Person“) a phenomenon *„nízká“.*

Protocol – vzor



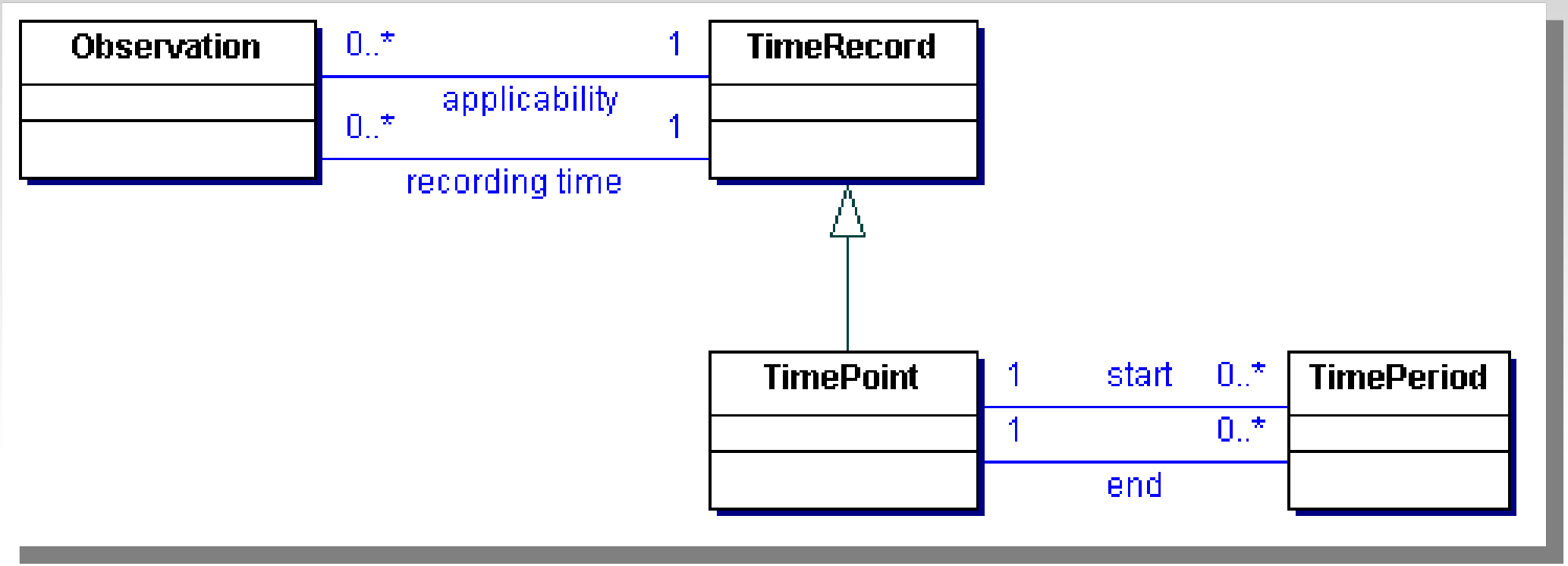
Způsob, jakým bylo pozorování provedeno.

Subtyping Observation Concepts – vzor



Pozorování přítomnosti subtypu znamená pozorování všech nadtypů.
Pozorování nepřítomnosti nadtypu znamená nepřítomnost všech subtypů.
Př.: Krevní skupina A a podskupiny A1 a A2

Dual Time Record – vzor

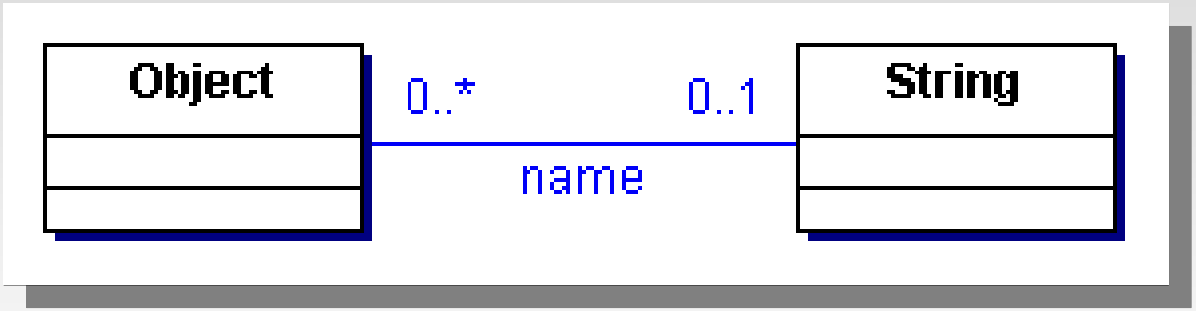


- Během pozorování je možné zaznamenat čas výskytu a čas pozorování.
- Lze zaznamenat jednotlivé časy nebo dobu (časovou periodu).
- Úkol: Pacient sdělil na prohlídce dne 1.4.2010 doktorovi, že před půl rokem mu začal bolet hrudník a trvalo to týden. Jak to namodelovat?
- Řešení: *Observation* = „bolest hrudníku“,
  recording time = 1.4.2010, *applicability* = 1.12.2009 až 8.12.2009

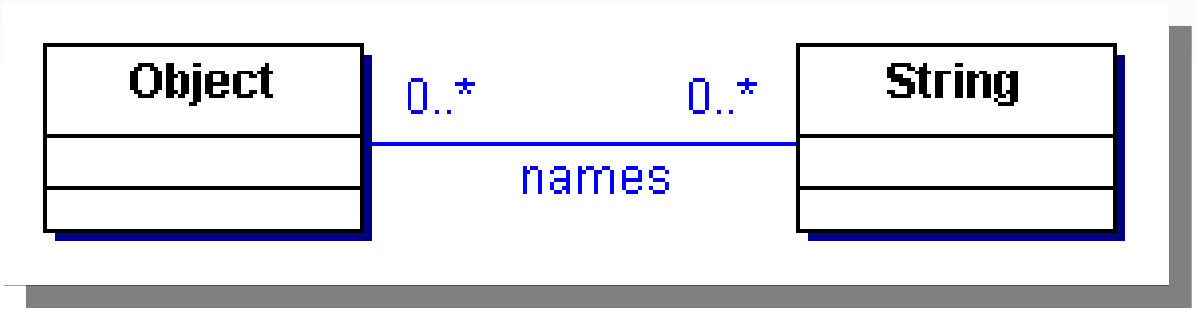
Soubor vzorů Reffering to Objects

- Pojmenování a identifikace objektů z pohledu uživatele systému
- V reálném světě se na objekty odkazujeme nejčastěji pomocí jména, přestože teto přístup nezaručuje jednoznačnou identifikaci.
  - Př.: Masarykovo náměstí
- Vzory:
  - Name
  - Identification Scheme
  - Object Merge
  - Object Equivalence

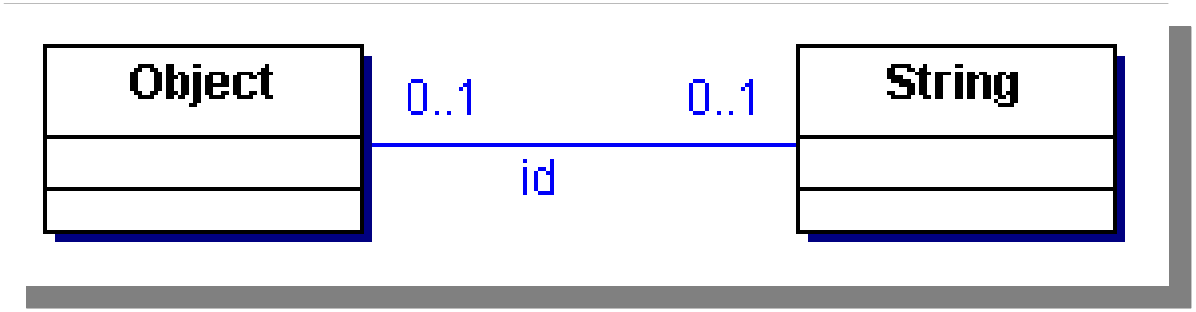
Name – vzor



objekt nemusí mít jméno, stejné jméno může mít více objektů

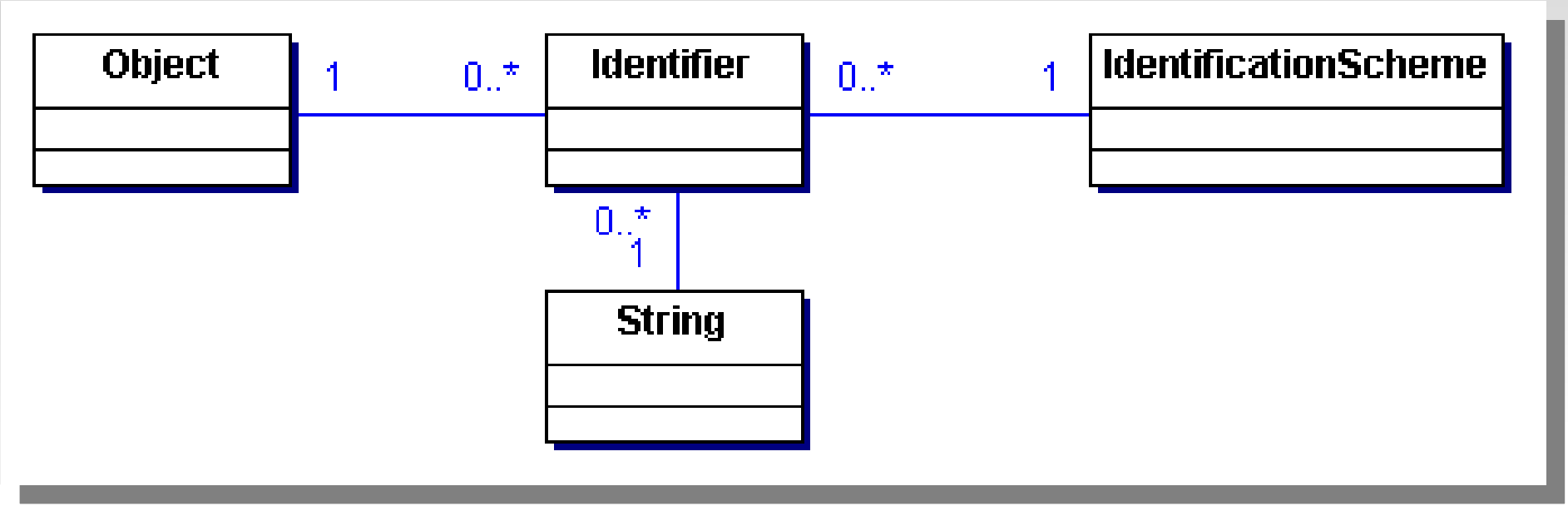


objekt se jménem a aliasy, viz také *Identification Scheme*



jednoznačný identifikátor objektu

Identification Scheme – vzor

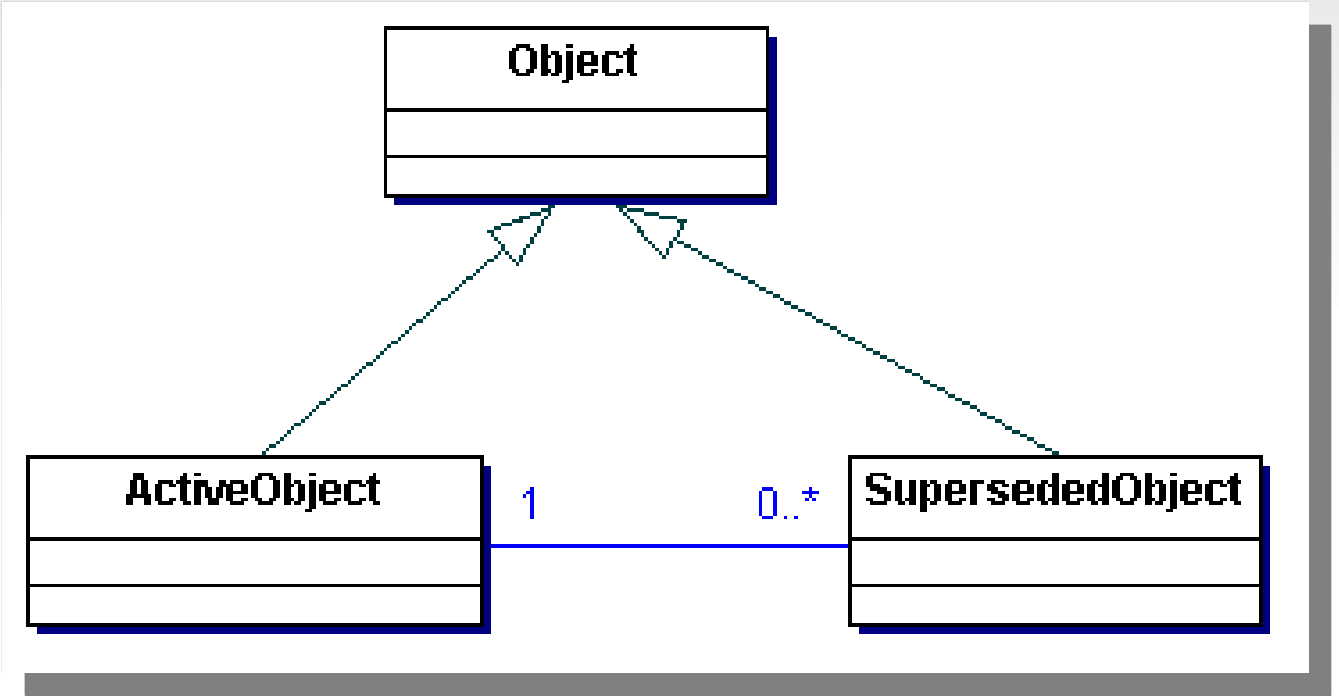


Princip: Na objekt se odkazujeme podle několika identifikačních schémat.
Př.: identifikace bank pomocí kódů SWIFT, CHAPS, ...
Otázka: Kolika různými způsoby evidujeme člověka?
Odpověď: RČ, jméno, přezdívka, „Petr z práce“, atd.

IdenticationScheme = ”rodným číslem”, ”popisem”, …
Object = konkrétní osoba
String = hodnota identifikátoru

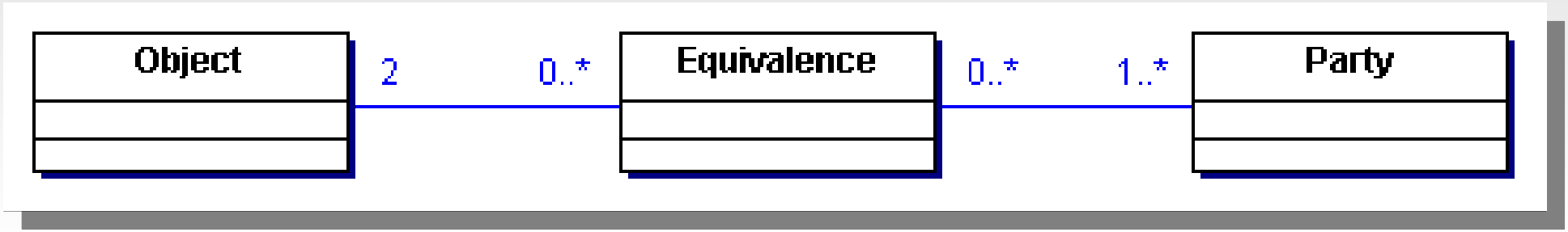
Object Merge – vzor

- V systému vznikají (částečně) duplicitní objekty (o nově přijatém pacientovi se zjistí, že má již záznam na jiném oddělení)
- Strategie „Copy“ a „Replace“ ?



- Aktivní objekt nahradí „nahrazovaný“ objekt.
- Nahrazovaný objekt předává všechny zprávy aktivnímu objektu.
- To, co bylo dříve považováno za odlišné, je nyní nahrazeno jedním řešením.

Object Equivalence – vzor



Ekvivalence podmíněná tím, kdo ji používá (tvorba synonym pro danou skupinu uživatelů)