První nefalšované neoficiální democvičení z předmětu IUS

ER diagramy, Usecase diagramy a další blbiny

Přednášející: (hafo titulů) Jakub „Pirožek“ Ludwig

- Ahoj všichni!
- Pokud budete mít dotaz, tak se ptejte, od toho tu jsme
- Nebojte se, že se vám budou ostatní smát, budou ☺
- Snažte se mě pokud možno nerušit hraním Cska, Doty a dalších supr her, to můžem dělat potom ☺
- Jdeme na to…

Nazdárek!

- Ještě drobnost
- Nejsem geniální
- Nejsem dokonalý
- Dělám chyby
- Pokud nějakou chybu uvidíte, tak mi o ní řekněte, případně začněte házet rajčata
- Snažil jsem se, aby tu žádná chyba nebyla, ale vždycky se nějaká najde ☺

Nazdárek!

Upozornění

Pořizování obrazových a zvukových záznamů přednášek k účelu dalšího šíření ~~není~~ je bez výslovného souhlasu přednášejícího dovoleno.

Zadání – Školící agentura

- Máte navrhnout systém pro školící agenturu, která pořádá kurzy a rekvalifikace. Samozřejmostí je výpis těchto kurzů a rekvalifikací. Systém musí umožnit registraci zákazníků, přidávání účastníků k objednávce a odesílání mailů s potvrzením objednávky. Z administrační části musí být možné vkládat nové kurzy/rekvalifikace. Každý kurz/rekvalifikace má svého lektora (nebo lektory), který může učit více kurzů/rekvalifikací najednou. Každý K/R má několik termínů, které se mohou konat na různých místech Brna. Místa se budou většinou opakovat (agentura má pár školících místností). Dále je potřeba mít kontrolu nad objednávkami, účastníky a mít možnost vystavit fakturu pro objednávku. Faktura bude odeslána mailem. Veškeré odesílání mailů v systému se řeší dávkovým zpracováním vždy o půlnoci daného dne.

- Usecase je diagram zachycující případy užití navrhovaného systému
- Usecase nám říká, kdo jsou účastníci, kde jsou hranice systému a co mohou účastníci dělat
- Usecase je definován svým identifikátorem, názvem a specifikací

Část první – Usecase - teorie

- Detail případu užití specifikuje jednotlivé „bubliny“ usecasu
- Většinou se používá tabulka, ale nemá to žádný standard
- Detail je definován vstupní podmínkou(ami), tokem událostí uvnitř a pak výstupní podmínkou(ami)
- Neboli **kdy** se to může stát, **co** se stane a **co bude pak**

Část první – Usecase - teorie

- Říkal jsem, že teorie bude na chvilku, jdeme na něco z praxe ☺

Část první – Usecase – praxe

- Usecase vzniká na základě toho, co by zákazník od systému požadoval a jak se systém bude chovat ke svému okolí
- Pokud v tom místě udělá analytik chybu, tak je to pak většinou ta poslední chyba, kterou udělá
- Problém je v získání požadavků od klienta, klient obvykle neví, co chce, ale nepřestane, dokud to nedostane

Část první – Usecase - praxe

- Větička ze slajdů IUS:
- Úlohou analytika je dát zákazníkovi včas a za určenou cenu ne to, co chce, ale to, o čem nikdy ani nesnil, že chce.
- Až když to dostane tak zjistí, že je to přesně to, co celou dobu chtěl ☺
- **Teď se asi smějete, ale tohle je přesně ono!**

Část první – Usecase - praxe

- Takže, jak se nezbláznit z bublin?
- **Je potřeba určit si hranice systému**
- Usecasem se dá teoreticky namodelovat celý okolní vesmír, ale to my nechceme (kdo by to pak opravoval…)
- Tudíž si jasně vymezíme, jakou věc už v systému neřešíme
- Uvažujme na moment náš příklad se školící agenturou
- Nebudeme modelovat, že objednatel musí zjistit údaje od účastníků, to je nám fuk

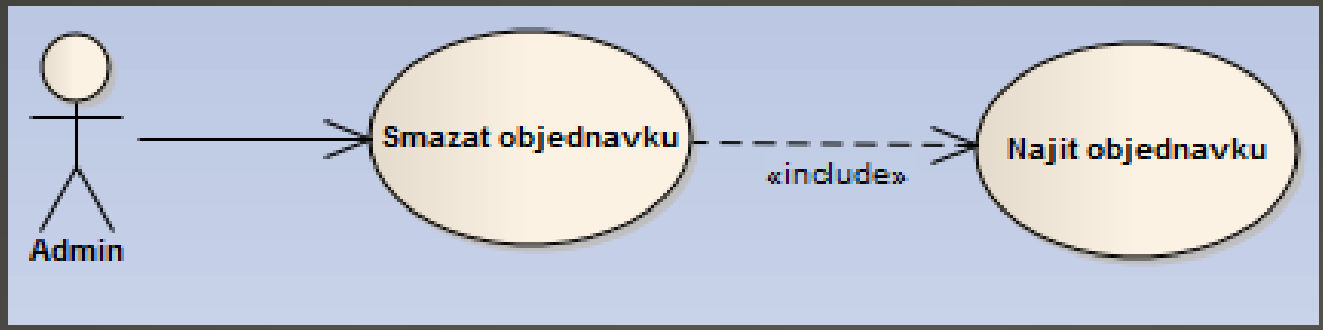
Část první – Usecase - praxe

- <<include>> a <<extend>> WTF?
- Jedná se o vazby mezi UC
- Jsou dobré k tomu, že můžeme nějakou „bublinu“ použít vícekrát, případně jí využít jinak, a nemusíme kvůli tomu modelovat „bublinu“ novou
- A teď k čemu jsou teda doopravdy dobré…

Část první – Usecase - praxe

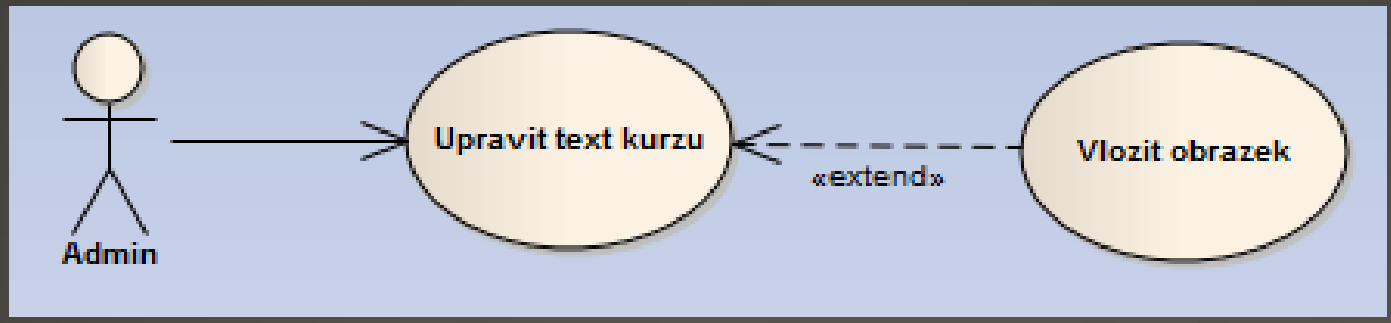
- Include – Do jedné „bubliny“ vložím funkčnost druhé „bubliny“
- Bývá povinná pro UC kam se vkládá
- Například pokud bude chtít administrátor smazat objednávku, je jasné, že jí musí první najít
- Šipka se maluje **od** UC kam chci vložit a ukazuje **na** UC, který vkládám

Část první – Usecase - praxe


Admin
Smazat objednavku
«include»
Najit objednavku

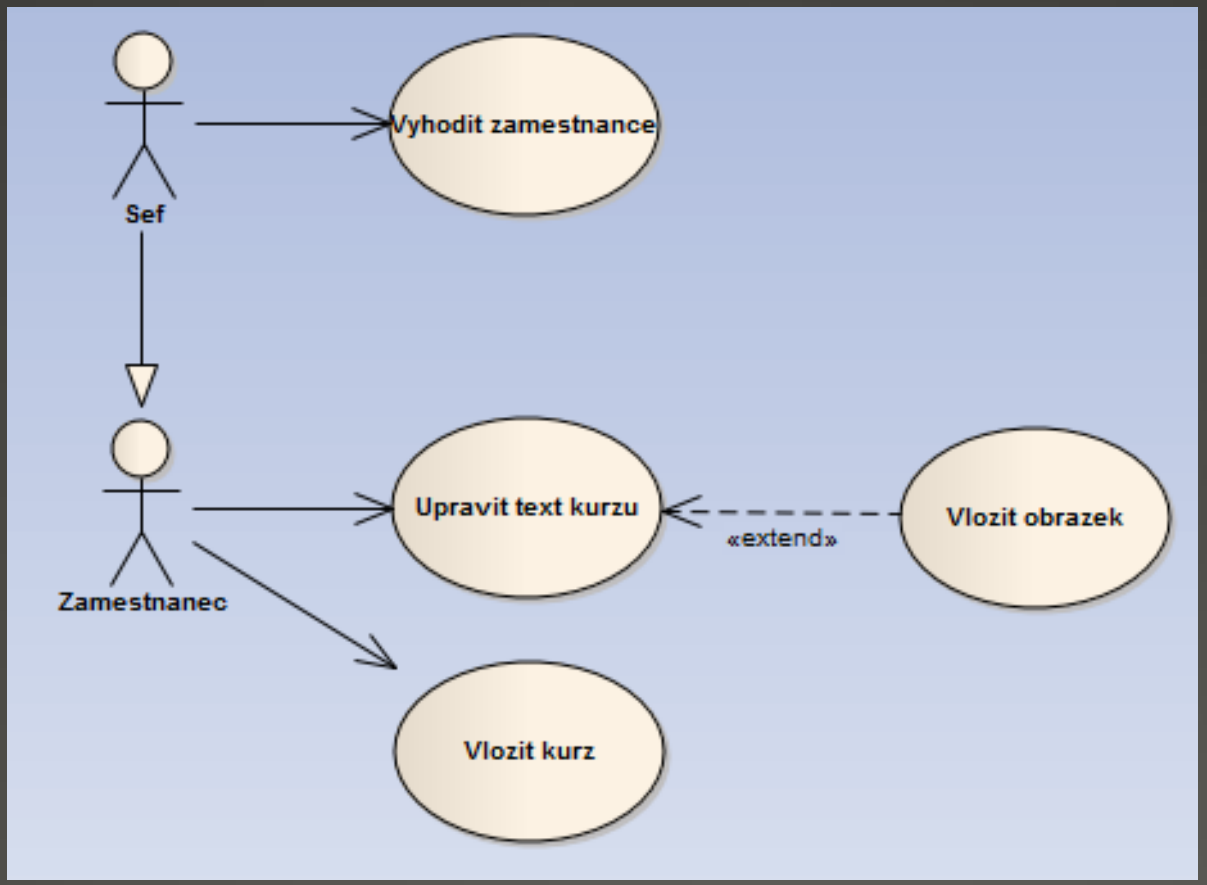
- Extend – Rozšíření funkčnosti jedné „bubliny“ jinou „bublinou“
- Není nutně povinná, může nastat jen za určitý podmínek, nebo pokud si to uživatel přeje
- Šipka se maluje **od** UC, který rozšiřuje funkčnost **do** UC, který tu funkčnost bude mít
- Jednoduše řečeno, je to opačně než u include ☺

Část první – Usecase - praxe


Admin
Upravit text kurzu
«extend»
Vlozit obrazek

- Zobecnění aktora
- Umožňuje nám říct, že všechno, co dělá jeden aktor, může dělat i druhý aktor
- Maluje se jako šipka (jak jinak) a má od aktora, který dědí vlastnosti jiného aktora
- Pro ty co umí OOP – tohle je prostě dědičnost
- A teď příklad, vezmeme opět naší školící agenturu…

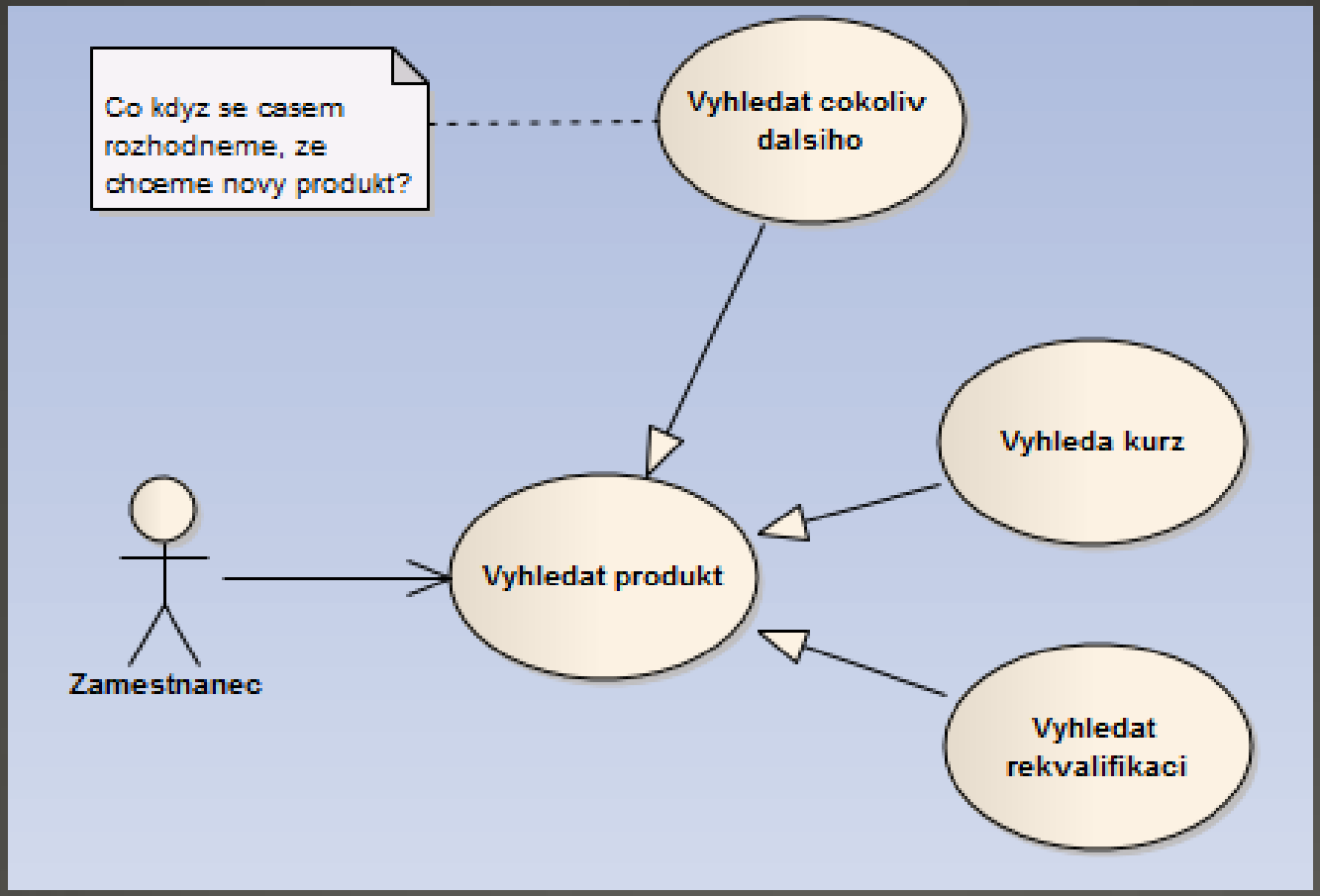
Část první – Usecase - praxe



Část první – Usecase - praxe

- Zobecnění případu užití
- To samé co zobecnění akorát pro „bubliny“
- Příklad: Řekněme, že bude potřeba vyhledávat v kurzech a rekvalifikacích
- Můžeme na to mít dvě bubliny, nebo to uděláme chytře
- Uvažujeme do budoucna, co když začneme nabízet i něco dalšího?

Část první – Usecase - praxe



Část první – Usecase - praxe

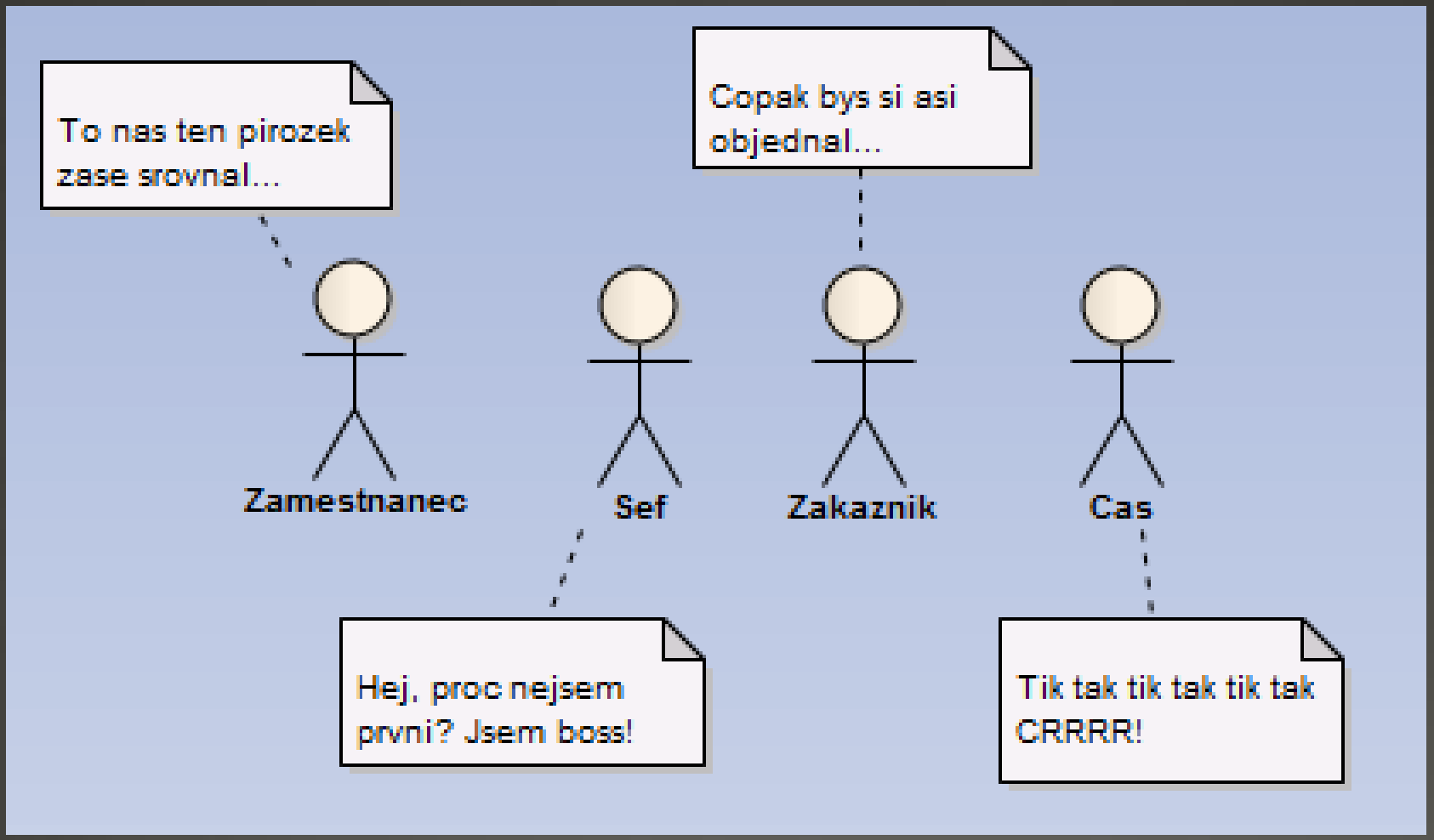
- Tak, praktické věci, které můžete potřebovat máme za sebou, jdeme se vrhnout na UC pro náš příklad

Část první – Usecase - praxe

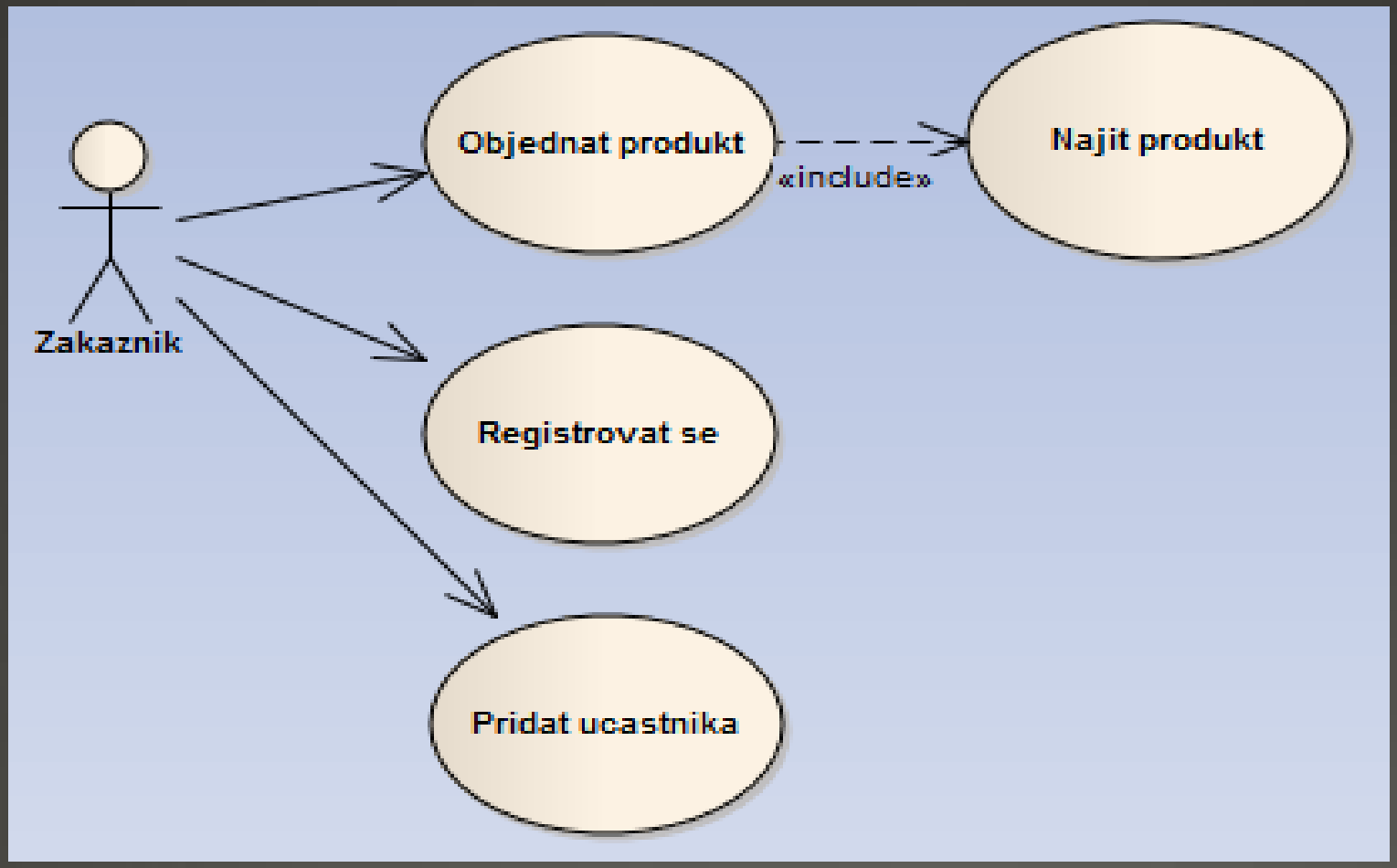
- DOTAZY?

- Máte navrhnout systém pro školící agenturu, která pořádá kurzy a rekvalifikace. Samozřejmostí je výpis těchto kurzů a rekvalifikací. Systém musí umožnit registraci zákazníků, přidávání účastníků k objednávce a odesílání mailů s potvrzením objednávky. Z administrační části musí být možné vkládat nové kurzy/rekvalifikace. Každý kurz/rekvalifikace má svého lektora (nebo lektory), který může učit více kurzů/rekvalifikací najednou. Každý K/R má několik termínů, které se mohou konat na různých místech Brna. Místa se budou většinou opakovat (agentura má pár školících místností). Dále je potřeba mít kontrolu nad objednávkami, účastníky a mít možnost vystavit fakturu pro objednávku. Faktura bude odeslána mailem. Veškeré odesílání mailů v systému se řeší dávkovým zpracováním vždy o půlnoci daného dne.

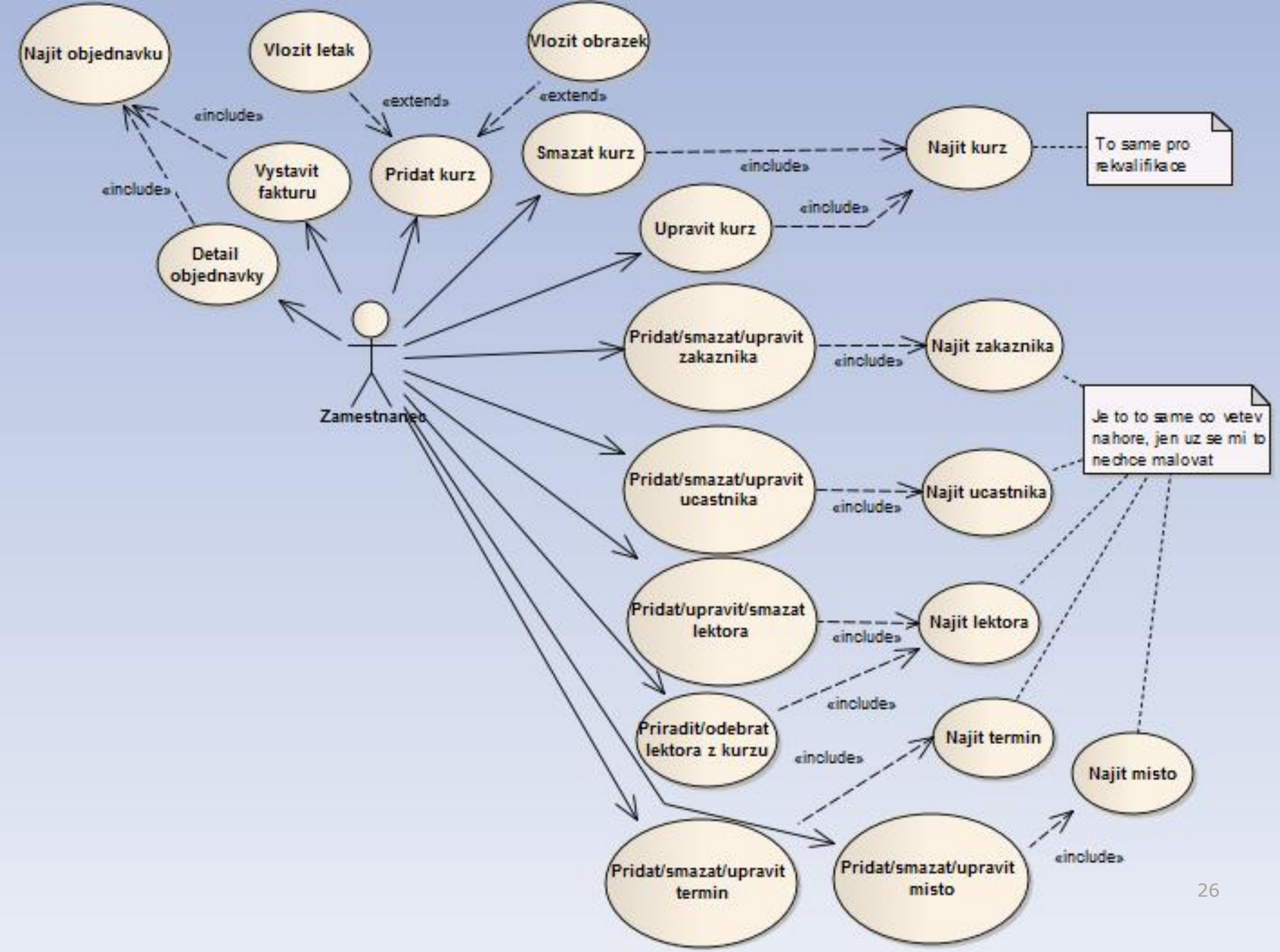
Zadání – Školící agentura

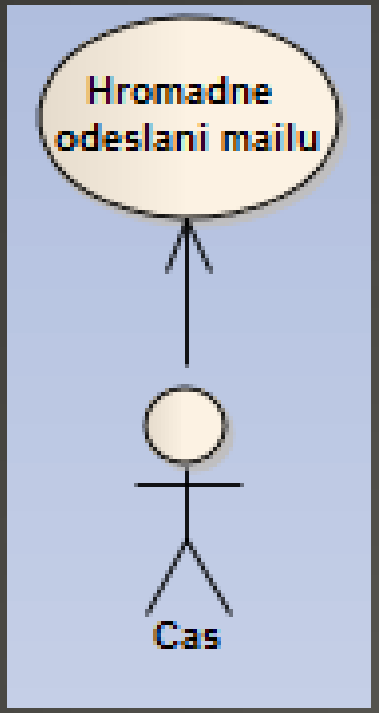


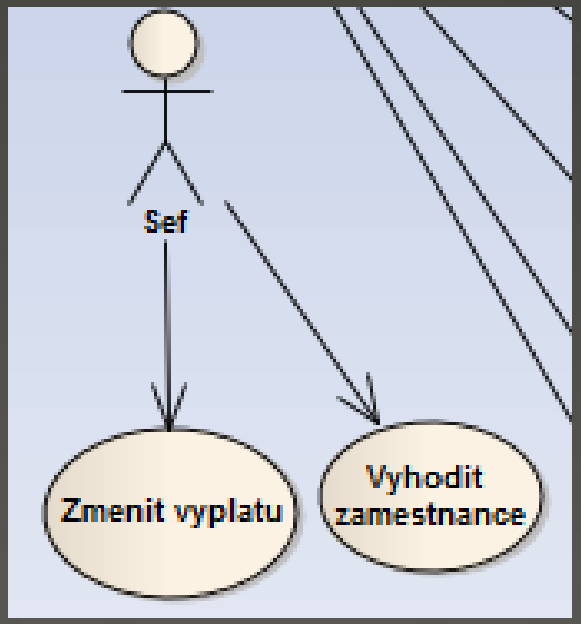
Část první – Usecase - návrh



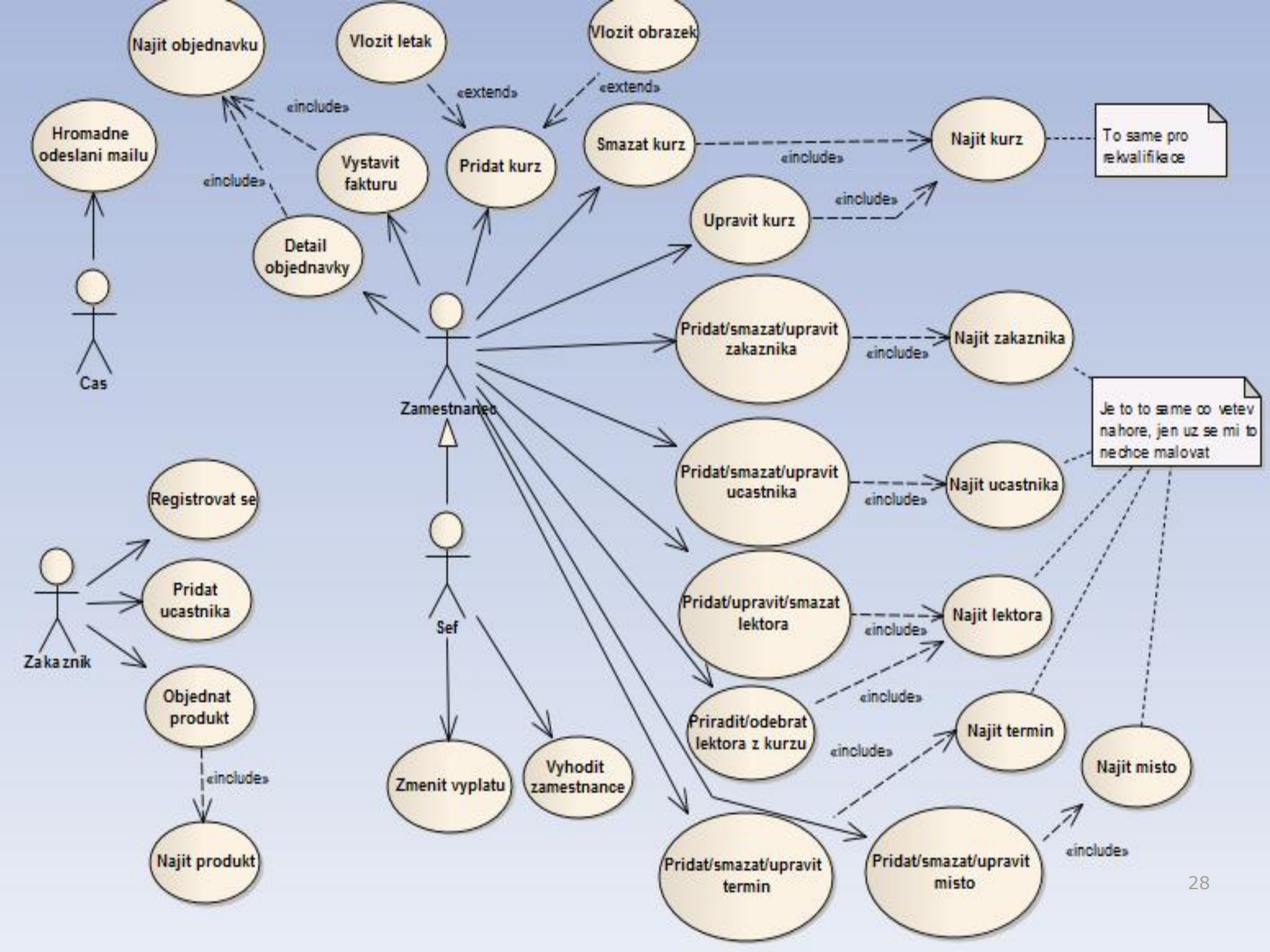
Část první – Usecase - návrh


Najit objednavku
Vlozit letak
Vlozit obrazek
«include»
«extend»
«extend»
Vystavit fakturu
Pridat kurz
Smazat kurz
Najit kurz
To same pro rekvalifikace
«include»
«include»
«include»
Detail objednavky
Upravit kurz
Pridat/smazat/upravit zakaznika
Najit zakaznika
Zamestnanec
Je to to same co vetev nahore, jen uz se mi to nechce malovat
Pridat/smazat/upravit ucastnika
Najit ucastnika
Pridat/upravit/smazat lektora
Najit lektora
Priradit/odebrat lektora z kurzu
Najit termin
Najit misto
Pridat/smazat/upravit termin
Pridat/smazat/upravit misto





Část první – Usecase - návrh

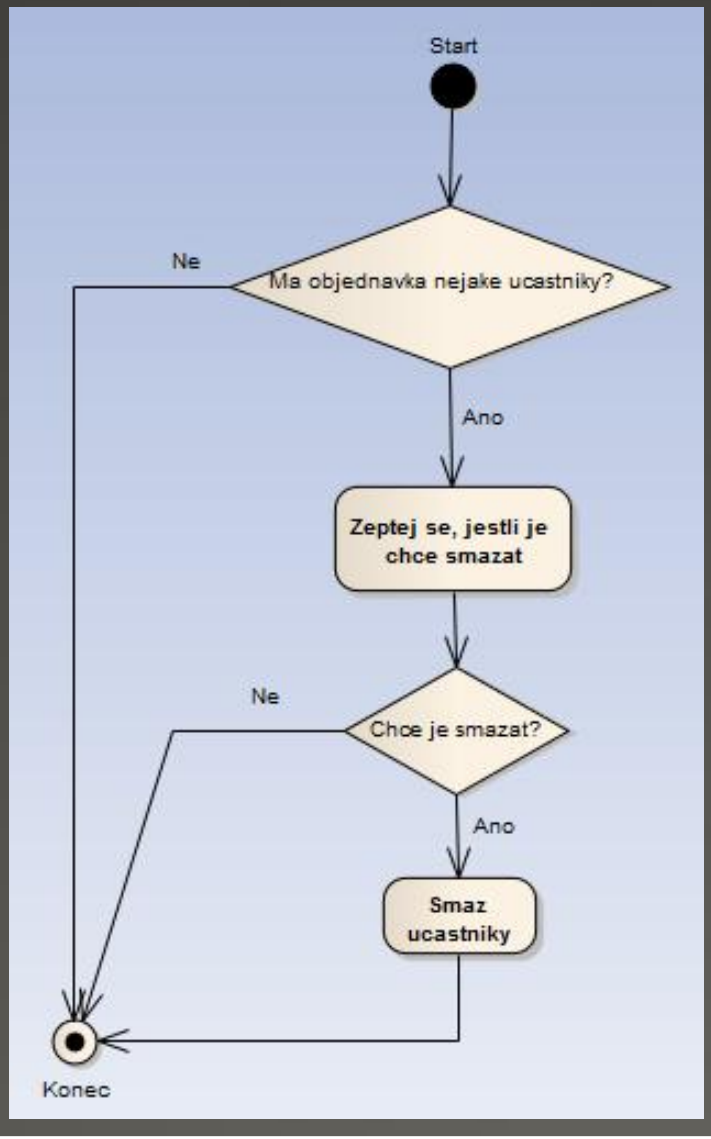

Najit objednavku
Vlozit letak
Vlozit obrazek
«extend»
«extend»
«include»
Hromadne odeslani mailu
Vystavit fakturu
Pridat kurz
Smazat kurz
Najit kurz
To same pro rekvalifikace
«include»
«include»
«include»
Detail objednavky
Upravit kurz
Cas
Zamestnanec
Pridat/smazat/upravit zakaznika
Najit zakaznika
«include»
Je to to same co vetev nahore, jen uz se mi to nechce malovat
Pridat/smazat/upravit ucastnika
Najit ucastnika
«include»
Registrovat se
Pridat ucastnika
Zakaznik
Sef
Pridat/upravit/smazat lektora
Najit lektora
«include»
Objednat produkt
«include»
Priradit/odebrat lektora z kurzu
Najit termin
«include»
«include»
Najit misto
Zmenit vyplatu
Vyhodit zamestnance
Najit produkt
Pridat/smazat/upravit termin
Pridat/smazat/upravit misto
«include»

- V praxi se to řeší tak, že samozřejmě nevytvoříme takovou zrůdu, ale každá „bublina“ může mít svůj detail
- A v detailu může být další detail a v něm další… většinou se dělá jen jeden level detailu
- Pro jednoduchost vám stačí pro projekt v IUS vyrobit něco takového
- Konvence: aktor do rohů a ke stranám, „bubliny“ doprostřed
- Jak vidíte, já jí tu porušuji kvůli tomu, aby se mi to vešlo na slajd ☺

Část první – Usecase - návrh

- Dalším krokem je vyrobit detaily případů užití, neboli, co se vlastně v těch „bublinách“ bude dít
- Může se použít tabulka, activity diagram, textový popis,…
- Důležité je, aby to pro danou situaci dostatečně a srozumitelně popsalo, co se vlastně má stát
- Příklad smazání účastníků u objednávky

Část první – Usecase - návrh


Start
Ma objednavka nejake ucastniky?
Ne
Ano
Zeptej se, jestli je chce smazat
Chce je smazat?
Ne
Ano
Smaz ucastniky
Konec

- Tímto můžeme kapitolu usecasu uzavřít a vrhneme se na návrh ER diagramu
- Ale předtím, krátká pauza ☺

Dotazy?

Část první – Usecase - návrh

- Diskuze

Část druhá – ERD - návrh

Část druhá – ERD - teorie

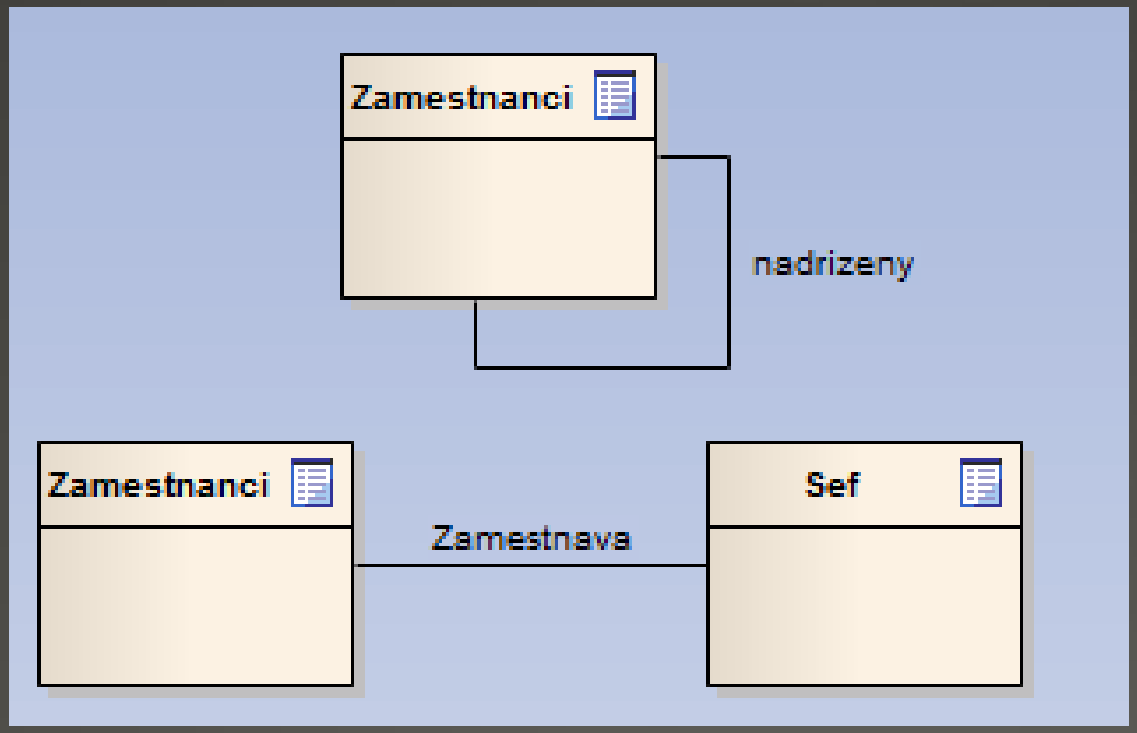
- **ERD – Entity relationship diagram**
- Zobrazuje vztahy mezi entitami
- Je klíčový pro pozdější návrh databáze
- Při jeho návrhu se snažíme:
  - Mít v systému všechna důležitá data
  - Nemít v systému redundantní data
  - Popsat, jaký je vztah mezi daty

- Slovníček pojmů pro práci s ERD
- „*Entita*“ – něco, co existuje v reálu a můžeme to jednoznačně určit
  - Kurz s ID 1
- „*Entitní množina*“ – soubor entit stejného typu, které sdílí atributy
  - Kurzy
- „*Atribut*“ – nějaká vlastnost entity
  - Název kurzu, jeho cena,…
- „*Relace*“ – vztah(asociace) mezi entitami
  - Kurz ID 1 má přiřazeného lektora ID 1

Část druhá – ERD - teorie

- **Stupeň vztahu**
- Pokud má entita vazbu pouze sama na sebe, je to unární
- Pokud mají mezi sebou dvě entity vazbu, je to binární
- Existuje ještě ternární, ale v praxi se setkáte nejspíš jen s binární nebo unární
- Příklad s naší agenturou

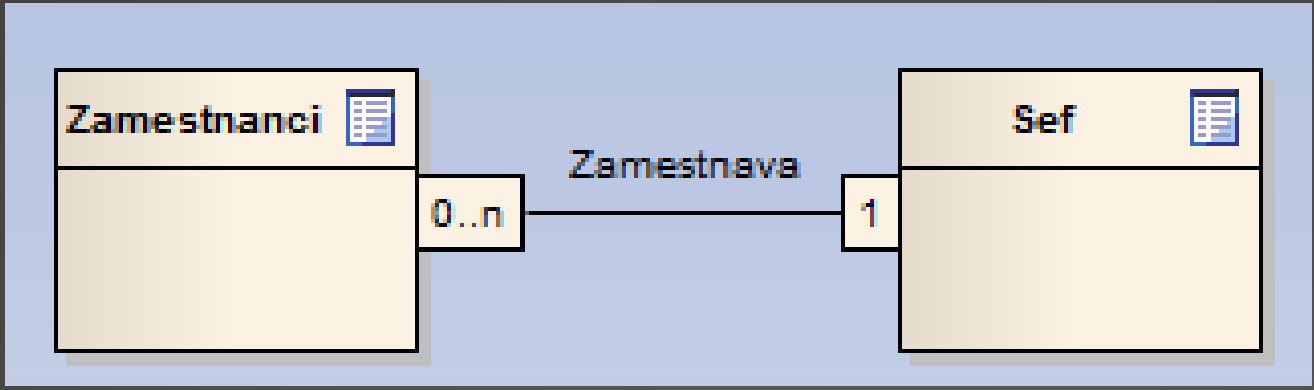
Část druhá – ERD - teorie



Část druhá – ERD - teorie

- **Kardinalita**
- Určuje množství vztahů kterých se může účastnit jedna entita
- Například pokud máme naší agenturu, která má jen jednoho šéfa, tak potom jeden šéf má 0..n zaměstnanců
- A zaměstnanci mají přesně 1 šéfa
- Lepší je to na obrázku

Část druhá – ERD - teorie



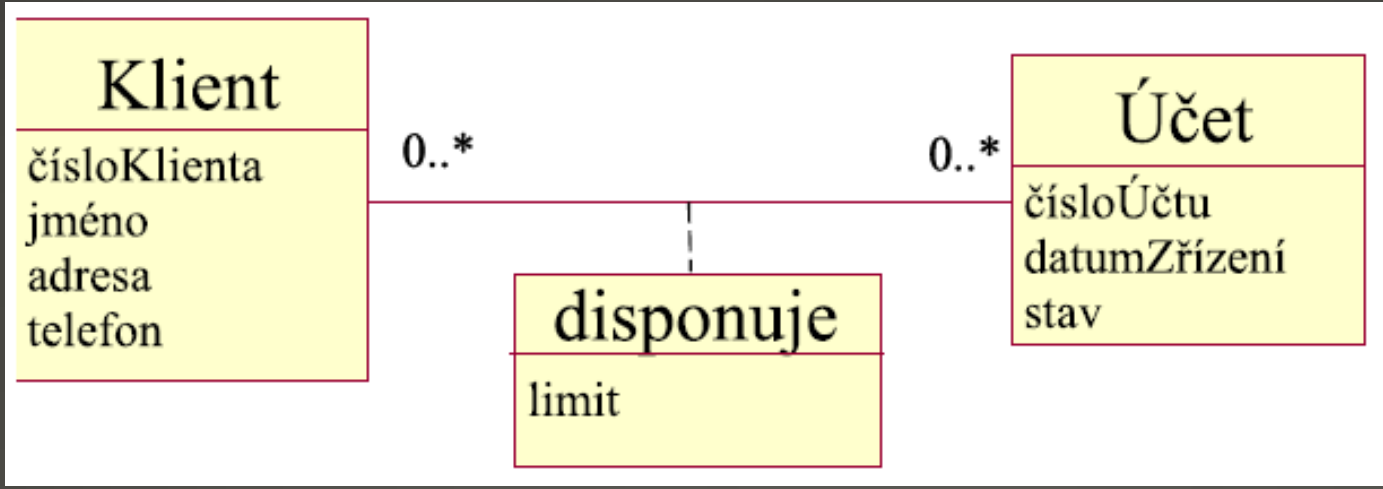
Část druhá – ERD - teorie

- **Členství**
- Na předchozím obrázku bylo krásně vidět i členství
- V podstatě to říká, kolik minimálně vztahů má jedna entita
- V našem případě šéf nemusí mít ani jednoho pracovníka (živnostník co pracuje sám)

Část druhá – ERD - teorie

- Atribut vztahu
- Nejen entita může mít atribut, i vztahy ho chtějí ☺
- A chtějí ho tehdy, když nemůžu přesně říct, které entitě by ten atribut měl náležet
- Využijeme slajdy IUS

Část druhá – ERD - teorie



Část druhá – ERD - teorie

- Patří limit klientovi, nebo účtu?
- Klient může mít limity pro různé účty
- Účet může mít limit pro různé klienty
- V praxi se to řeší jako M:N problém
- A co je to M:N problém se dozvíte…

- …na některém z příštích slajdů ☺

Část druhá – ERD - teorie

- **Slabá entitní množina**
- Je to entitní množina, která má svůj identifikátor složený z atributů jiné entitní množiny
- No jo, jenže co to znamená?
- Každá entitní množina musí mít možnost identifikovat jednotlivé entity, které obsahuje
- Takže každá entita musí mít unikátní identifikátor
- Unikátní v rámci systému, který modelujeme

Část druhá – ERD - teorie

- Například klient má v naší agentuře jasné číslo, které je mu přiděleno při registraci
- Říkejme mu třeba ID
- Takže entitní množina Klienti je silná, protože má identifikátor tvořený z atributů, které obsahuje
- V databázích se identifikátoru říká primární klíč a může ho tvořit jeden až N atributů
- Platí pravidlo, že primární klíč měl být co nejmenší, ale stále jednoznačný

Část druhá – ERD - teorie

- Slabá entitní množina se často používá pro řešení M:N problému, který bude na příštím slajdu
- Její identifikátor (primární klíč) je tvořen kombinací primárních klíčů entitních množin, mezi kterými realizuje vazbu
- Pokud jsme na pochybách jestli modelovat slabou nebo silnou množinu, vždy modelujte silnou (nedá se tím nic pokazit)

Část druhá – ERD - teorie

- Nechápete o čem jsem mluvil u posledních 3 slajdů?
- Nic se neděje, já to taky nechápal ☺
- Pochopíte to za moment, jakmile vysvětlíme M:N problém a bude obrázek
- Jdeme na to…

Část druhá – ERD - teorie

- **M:N problém**
- Vzniká v okamžiku, kdy je mezi entitami vztah M:N, tedy více entit z jedné množiny má více vazeb na entity z druhé množiny a opačně
- Kdo to nechápe? ☺
- Ukážeme si to na našem zadání se školící agenturou
- Mrkneme na kurzy a lektory

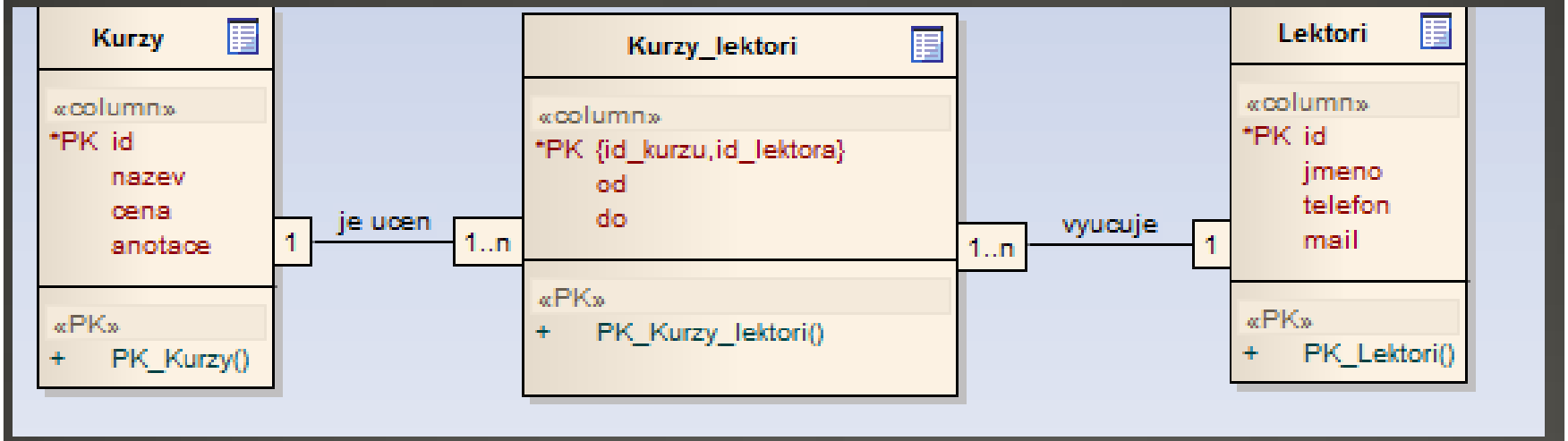
Část druhá – ERD - teorie

- „ Každý kurz/rekvalifikace má svého lektora (nebo lektory), který může učit více kurzů/rekvalifikací najednou.“
- Takže, musíme navrhnout ERD tak, abychom „někde“ mohli uchovat všechny kurzy, které lektor učí
- A zároveň u každého kurzu všechny jeho lektory

Část druhá – ERD - teorie

- Situaci nelze vyřešit přidáním atributů do entitních množin Kurzy a Lektori
- Pro každý nový kurz/lektora bychom museli mít nový atribut
- Což je blbost ☺
- Řešením je zavést novou entitní množinu, která bude uchovávat informace o vazbě
- A tuhle množinu můžeme vesele modelovat jako slabou entitní množinu
- Více na obrázku

Část druhá – ERD - teorie



Část druhá – ERD - teorie

- Protože tohle je poměrně důležitá věc a často to bývá jak u semestrálky, tak v projektu, tak dávám prostor pro

- Dotazy?

Část druhá – ERD - teorie

- Tentokrát té teorie bylo trošku víc, ale bylo to důležité.
- Pojďme se mrknout na trochu praxe ☺

Část druhá – ERD - teorie

- **Dobré zásady při modelování ERD**
- Entitním množinám dávám srozumitelná podstatná jména
- Vztahům naopak slovesa
- Nesnažím se modelovat celý systém jako entitní množinu (v našem případě nebudeme mít entitní množinu „Agentura“)
- Pokud máme atribut, jehož hodnota je důležitá, měli bychom ho modelovat jako entitu
- Držíme se toho, že nechceme v systému redundantní data

Část druhá – ERD - praxe

Část druhá – ERD - praxe

- Jak už jsem říkal na začátku, ERD je klíčový pro návrh databáze
- V praxi to znamená, že pokud už jste si zkoušeli databázi navrhnout, máte obrovský náskok
- Modelování ERD vám nebude dělat velký problém ☺
- Zkusíme si udělat ERD pro naše zadání

Zadání – Školící agentura

- Máte navrhnout systém pro školící agenturu, která pořádá kurzy a rekvalifikace. Samozřejmostí je výpis těchto kurzů a rekvalifikací. Systém musí umožnit registraci zákazníků, přidávání účastníků k objednávce a odesílání mailů s potvrzením objednávky. Z administrační části musí být možné vkládat nové kurzy/rekvalifikace. Každý kurz/rekvalifikace má svého lektora (nebo lektory), který může učit více kurzů/rekvalifikací najednou. Každý K/R má několik termínů, které se mohou konat na různých místech Brna. Místa se budou většinou opakovat (agentura má pár školících místností). Dále je potřeba mít kontrolu nad objednávkami, účastníky a mít možnost vystavit fakturu pro objednávku. Faktura bude odeslána mailem. Veškeré odesílání mailů v systému se řeší dávkovým zpracováním vždy o půlnoci daného dne.

Část druhá – ERD - návrh

- **Jak navrhnout ERD a zůstat normální**

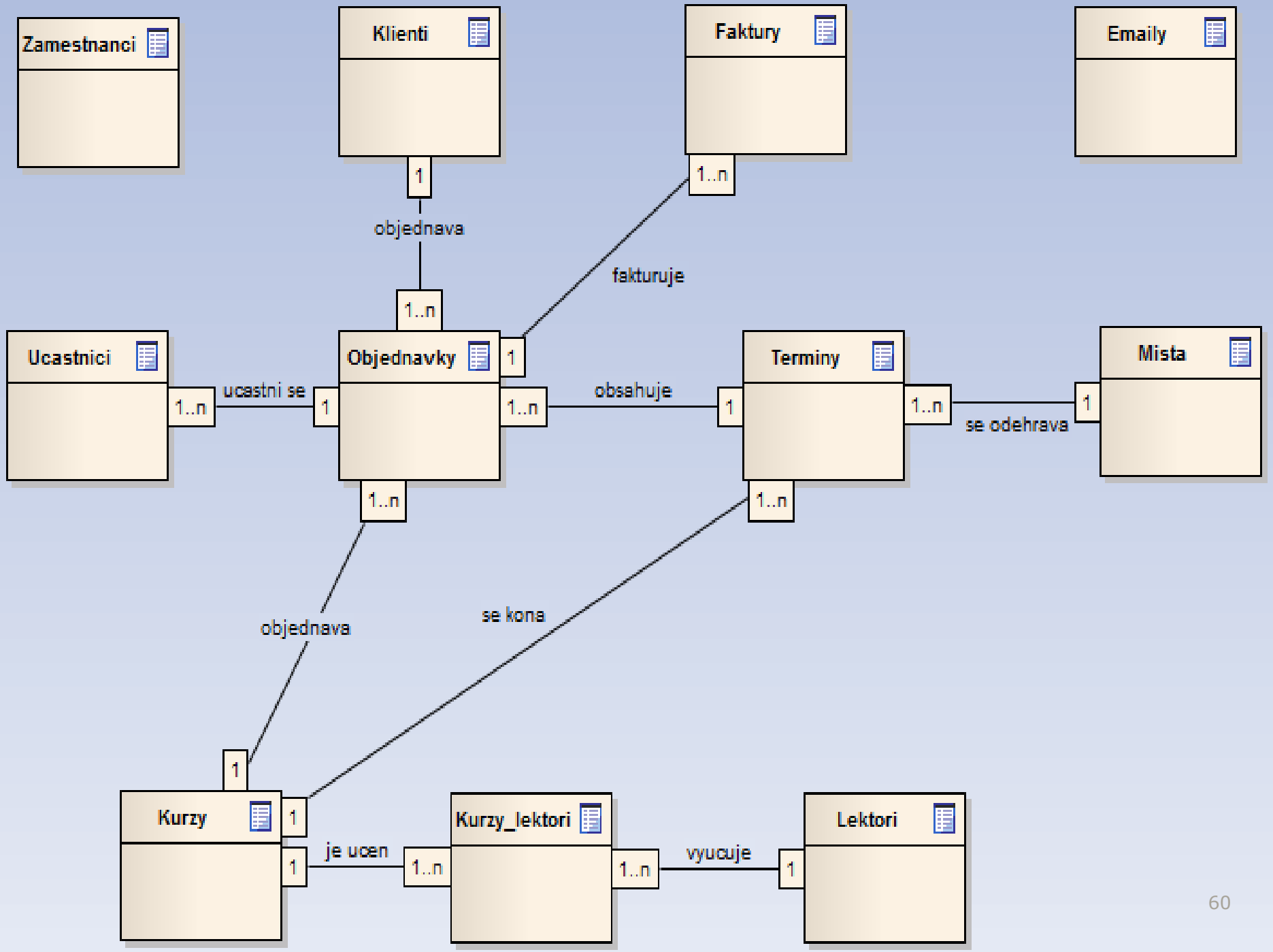
1. Zamyslím se, jaké všechny entity budu modelovat
2. Jednu si vyberu a určím její atributy
3. Ověřím, jestli je potřeba nějaký její atribut modelovat jako entitu
4. Šup na bod 2. a jedeme s další entitou
5. Vytvoříme vztahy (dáváme bacha na M:N)
6. Pojmenujeme vztahy a odstraníme zbytečné

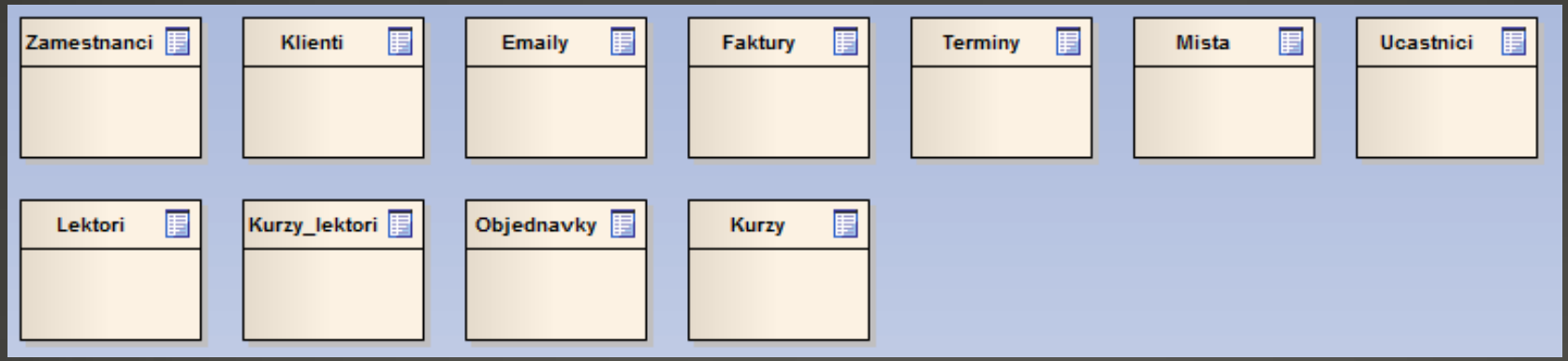
- Tak jedem, nejdřív se zamyslíme, jaké entity vidíme v zadání
  - Klienti
  - Účastníci
  - Objednávky
  - Faktury
  - Termíny
  - Zaměstnanci
  - Emaily
  - Místa
  - Kurzy
  - Lektoři

Část druhá – ERD - návrh

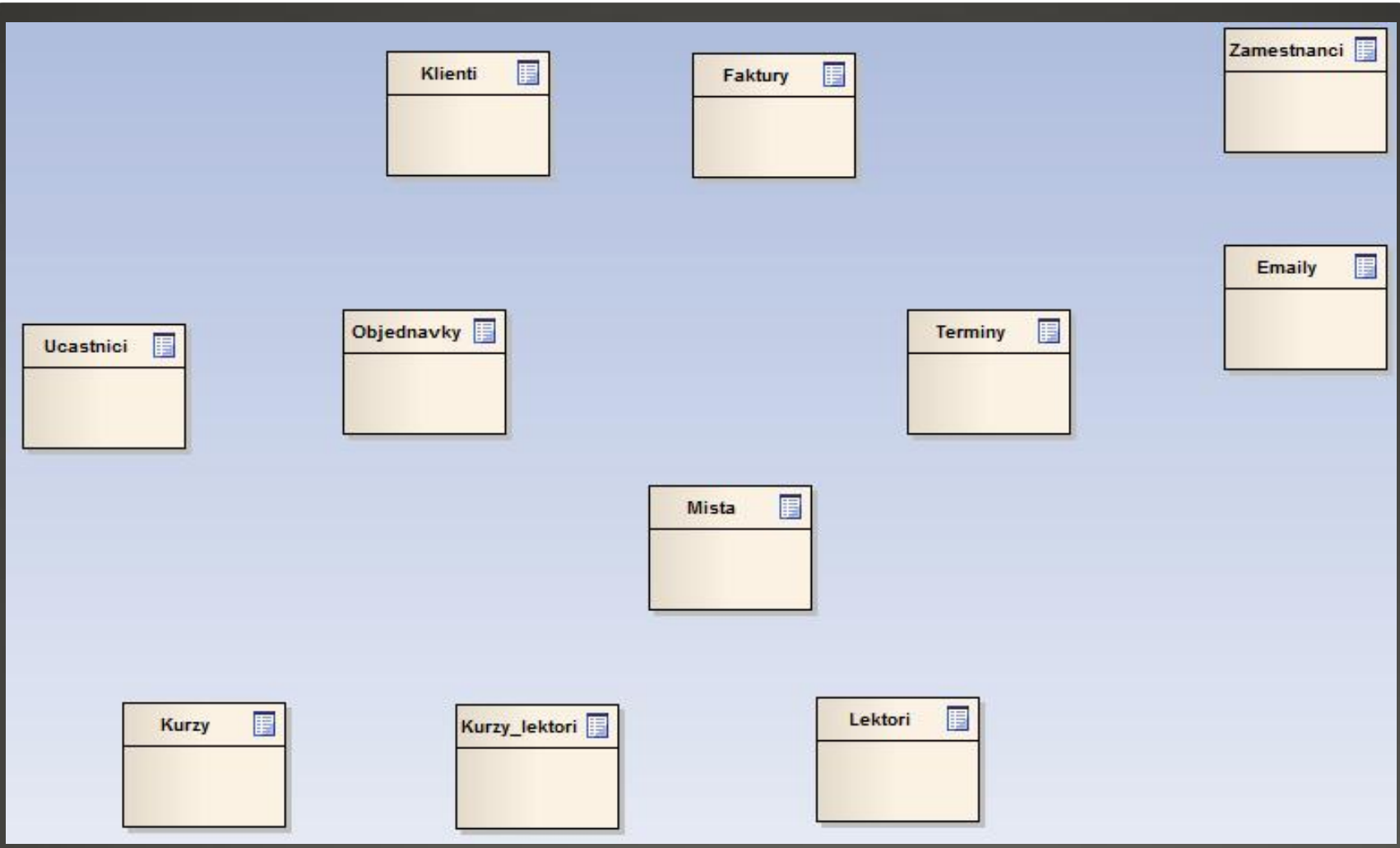
- Teď použijeme jednoduchý algoritmus o 6ti bodech, který byl na jednom z předchozích slajdů a vyjde nám krásný obrázek
- Nejprve vám ho ukážu a pak si zkusíme projít krok za krokem jeho tvorbu

Část druhá – ERD - návrh


Zamestnanci
Klienti
Faktury
Emaily
1
objednava
1..n
1..n
fakturuje
Ucastnici
Objednavky
Terminy
Mista
1..n
ucastni se
1
1..n
obsahuje
1
1..n
se odehrava
1
1..n
1..n
objednava
se kona
1
Kurzy
1
1
je ucen
1..n
Kurzy_lektori
1..n
vyucuje
1
Lektori



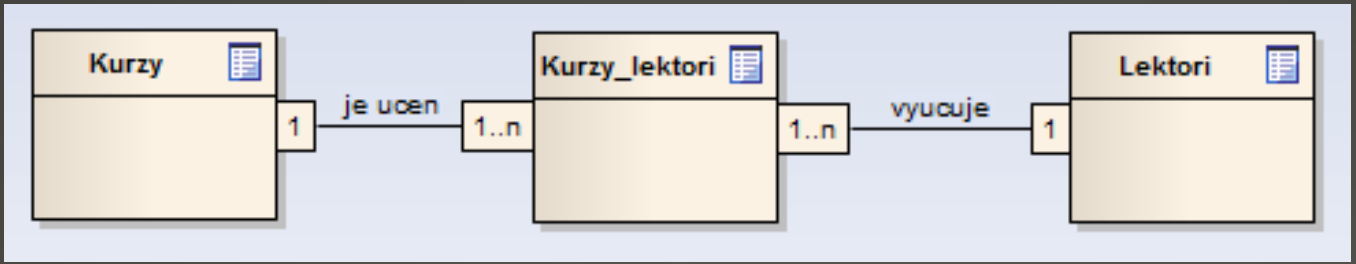
Část druhá – ERD - návrh



Část druhá – ERD - návrh

- Začneme naším M:N problémem, který už jsem vyřešil na jednom z předchozích slajdů
- Z důvodu úspory místa zatím nebudou mít naše entitní množiny atributy
- Z obrázku je jasné, že jeden kurz může mít nyní více lektorů a jeden lektor může učit více kurzů
- Konvence říká, že vazební entitní množina má jméno vzniklé kombinací entitních množin, které spojuje

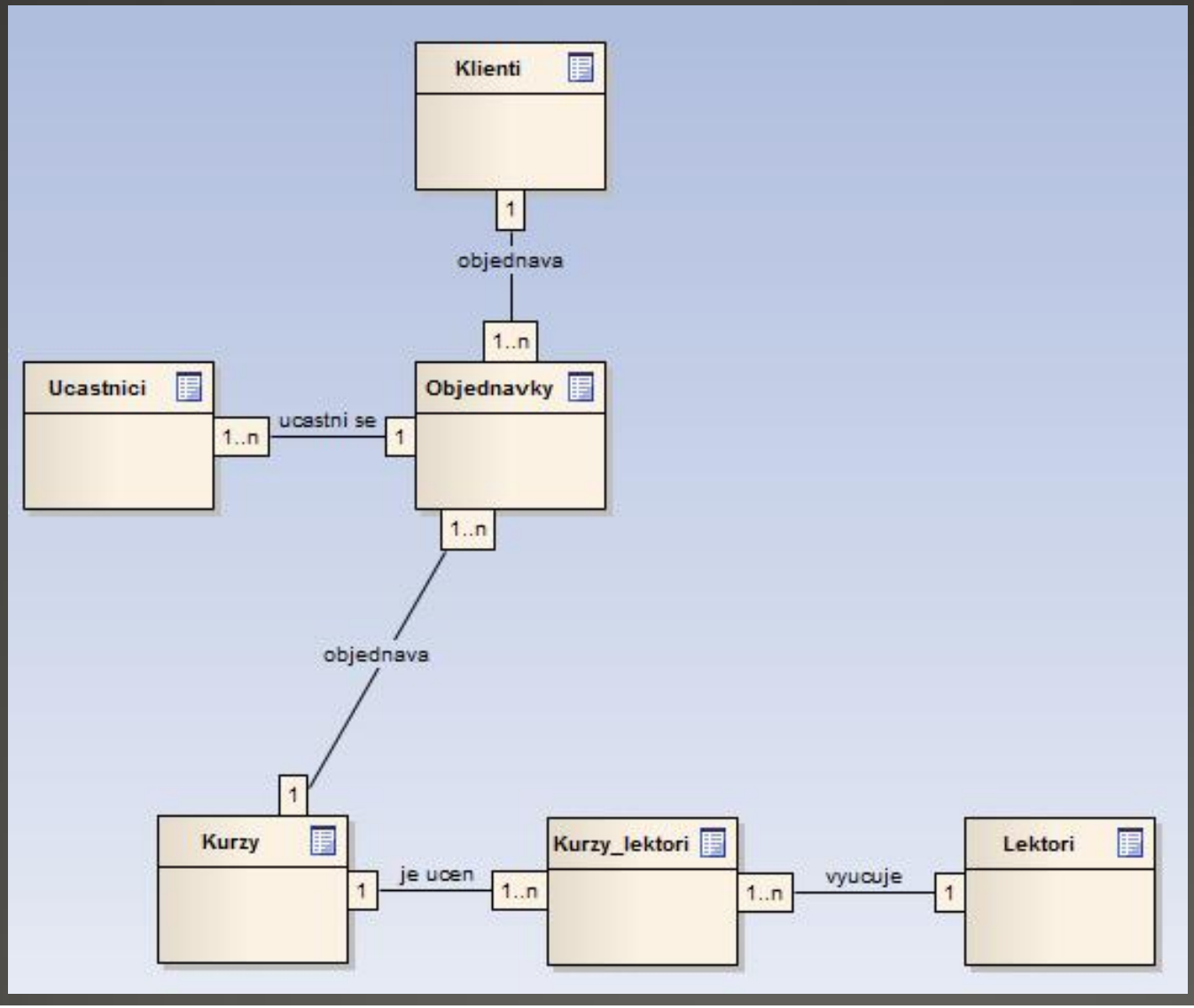
Část druhá – ERD - návrh



Část druhá – ERD - návrh

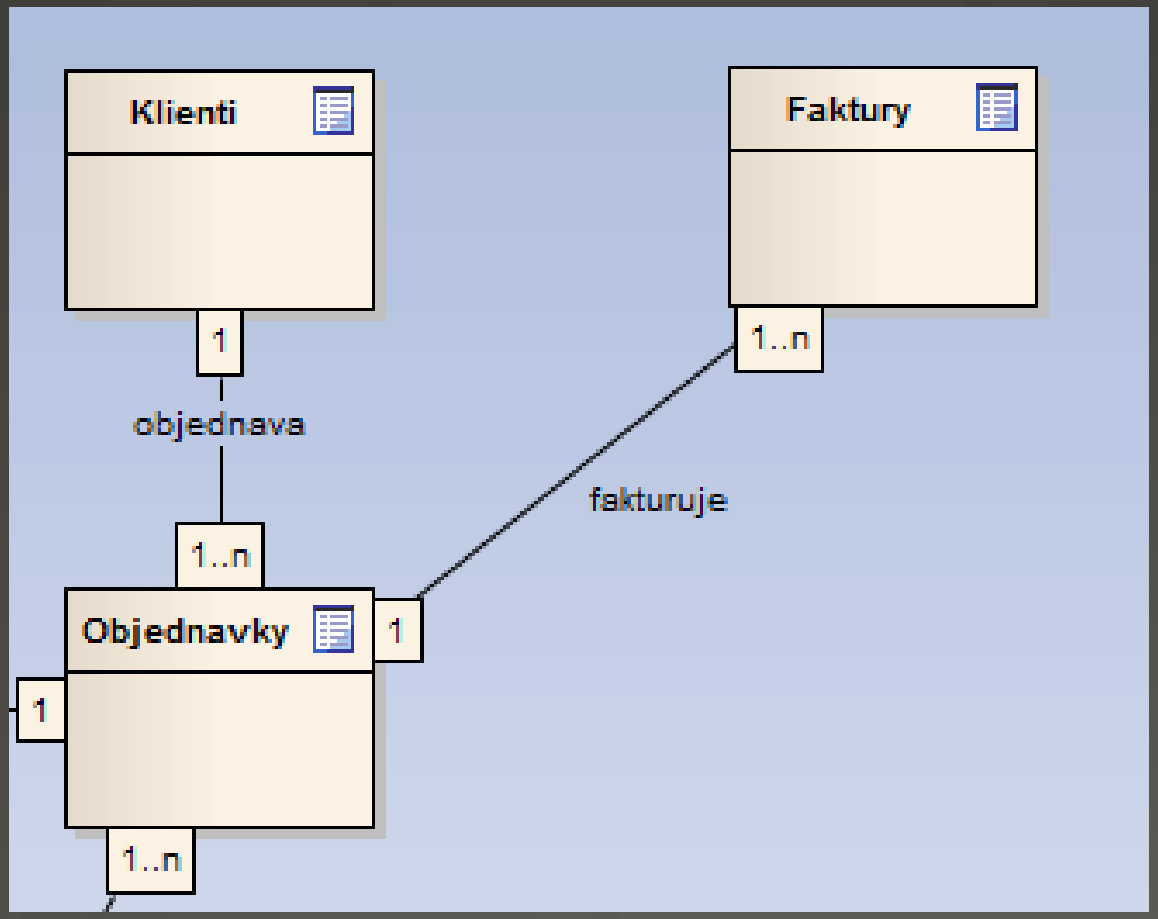
Část druhá – ERD - návrh

- Teď se podíváme na Klienty, Objednávky a Účastníky
- Jeden klient může mít v systému více objednávek – vazba 1:N
- Jedna objednávka může mít více účastníků – vazba 1:N
- K objednávce se také váže jeden kurz, ale jeden kurz může být ve více objednávkách – vazba 1:N ale 1 bude u kurzů


Klienti
1
objednava
1..n
Ucastnici
1..n
ucastni se
1
Objednavky
1..n
objednava
1
Kurzy
1
je uceni
1..n
Kurzy_lektori
1..n
vyucuje
1
Lektori

- Teď už to bude lehké, mrkneme na Faktury
- Faktury se váží k objednávce a jedna objednávka může mít více faktur
- Například faktura a následný dobropis
- Vzniká nám tedy další vazba 1:N

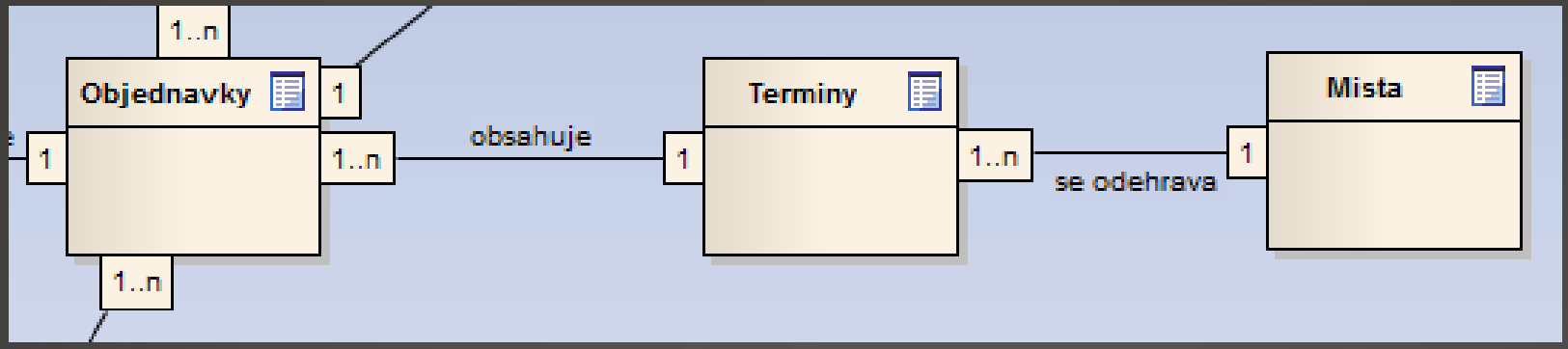
Část druhá – ERD - návrh



Část druhá – ERD - návrh

Část druhá – ERD - návrh

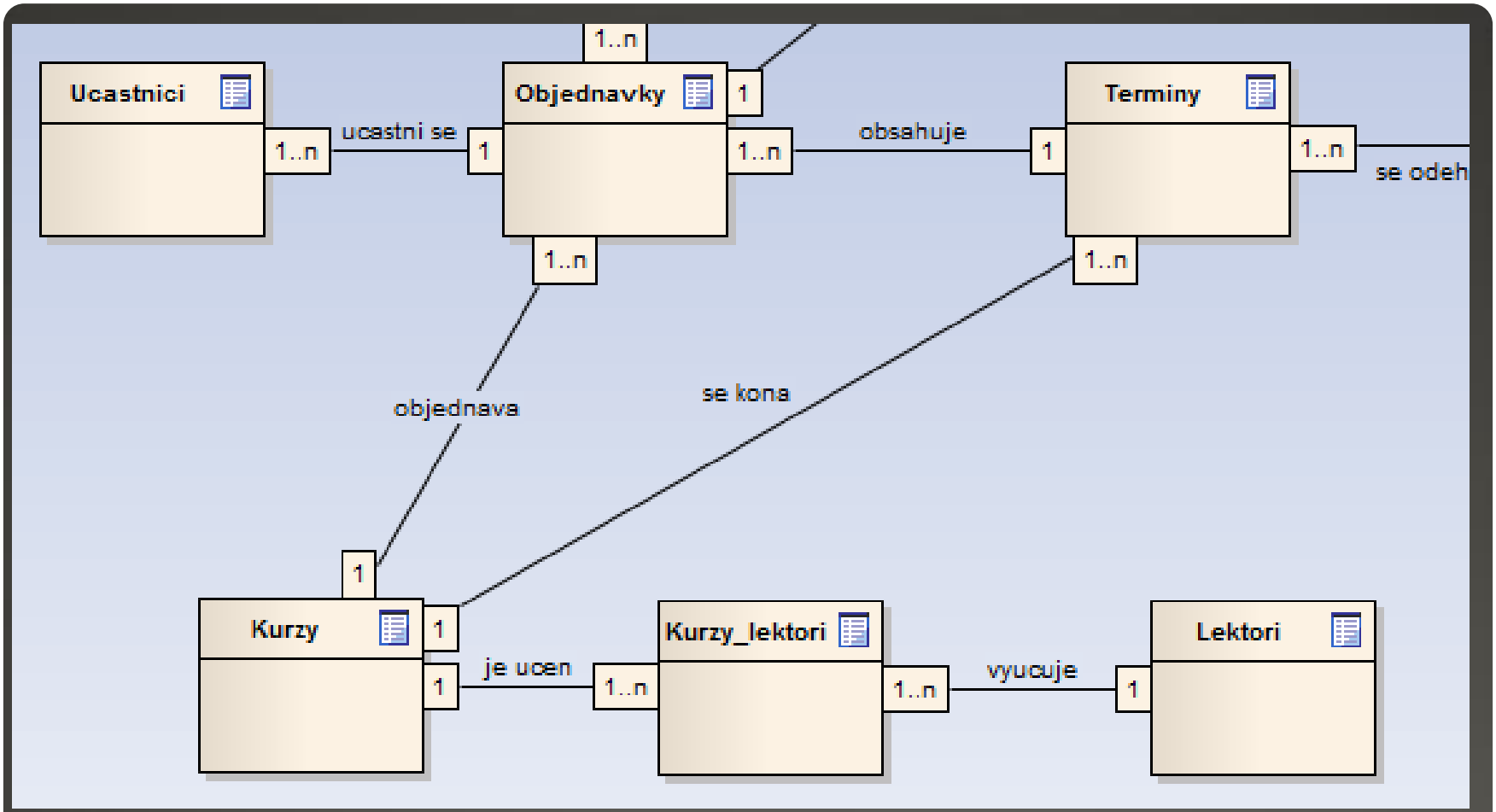
- Další budou Termíny a Místa
- Je jasné, že na jeden termín může být více objednávek – vazba 1:N
- A na jednom místě se může konat více termínů – opět vazba 1:N



Část druhá – ERD - návrh

- Ještě drobnost, chybí nám vazba mezi Kurzy a Termíny, protože termín se váže na kurz
- Což je krásná vazba 1:N ☺

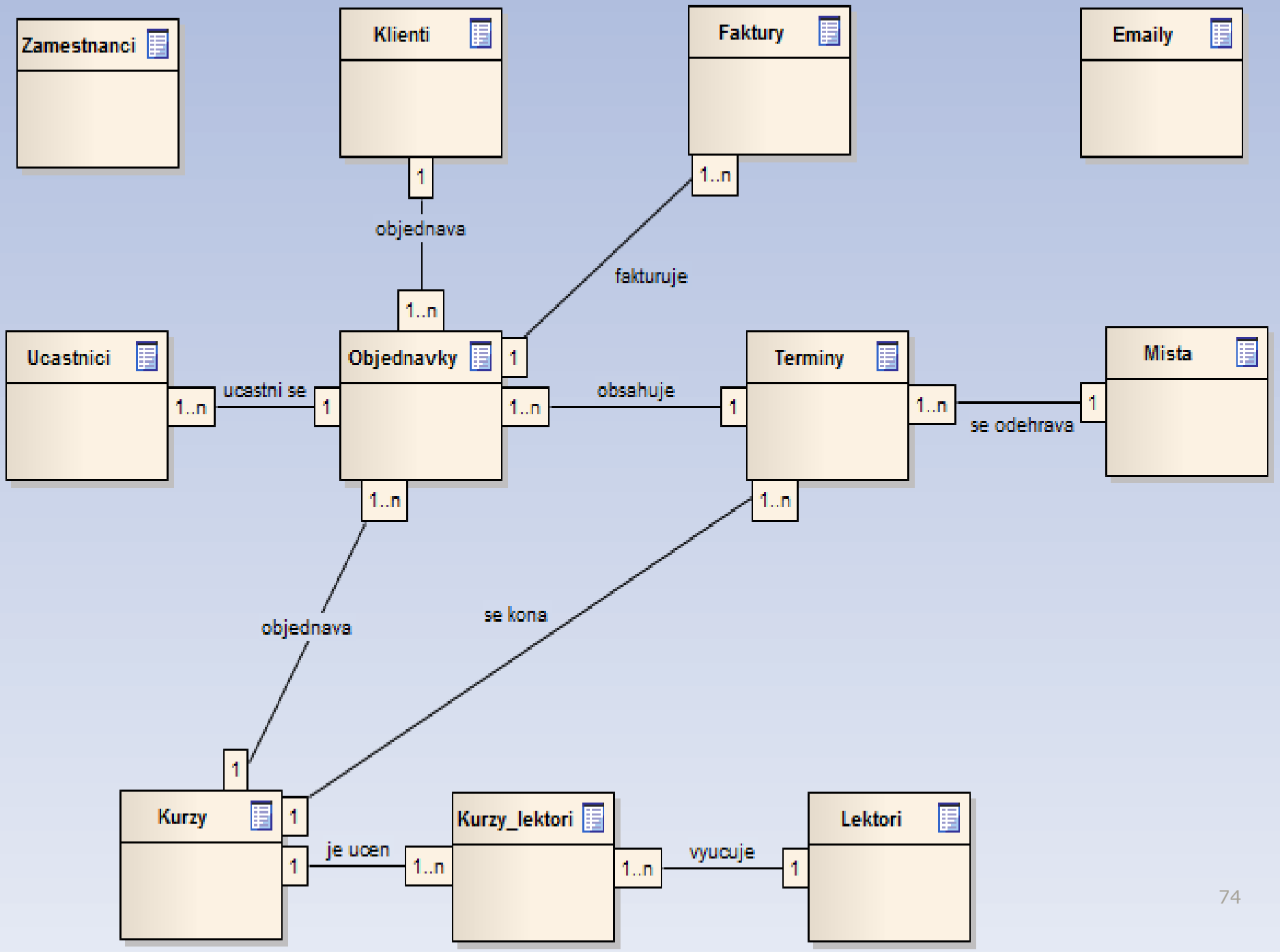
Část druhá – ERD - návrh



Část druhá – ERD - návrh

Část druhá – ERD - návrh

- Už jen kousek a máme hotovo
- Do diagramu přidáme dvě entitní množiny
- Zaměstnance a Emaily
- Nebudou mít žádné vazby, ale být tam musí
- Když nebudou v ERD nebudou v návrhu databáze a to by bylo špatně


Zamestnanci
Klienti
Faktury
Emaily
1
objednava
1..n
1..n
fakturuje
Ucastnici
Objednavky
1
Terminy
Mista
1..n
ucastni se
1
1..n
obsahuje
1
1..n
se odehrava
1
1..n
1..n
objednava
se kona
1
Kurzy
1
Kurzy_lektori
Lektori
1
je ucen
1..n
1..n
vyucuje
1

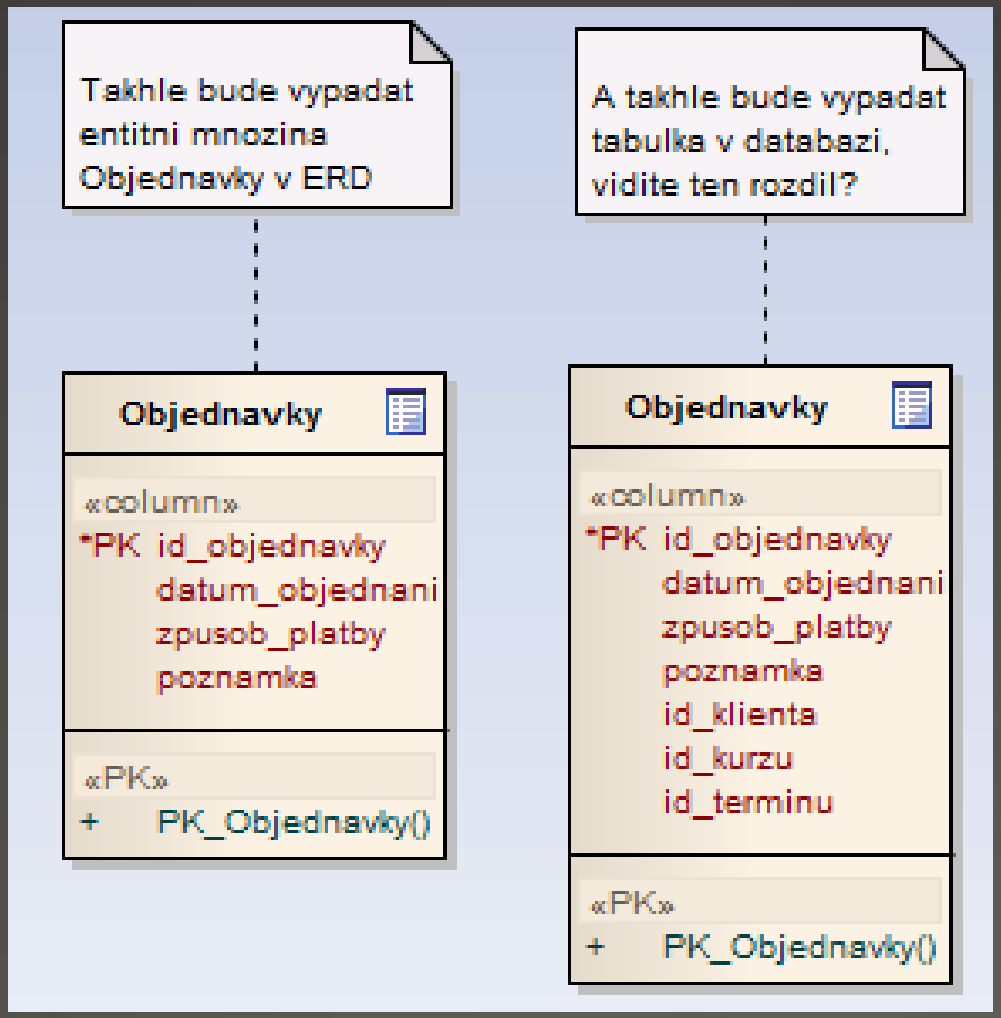
- Teď doplníme do entitních množin atributy
- Tahle část tvorby ERD je poměrně jednoduchá, proto si ukážeme jen pár tabulek
- Ale předtím…

- Dotazy?

Část druhá – ERD - návrh

- Začneme entitní množinou Objednávky
- U objednávky by bylo dobré vědět:
  - Kdy jsem objednával
  - Jak chci platit
  - Případnou poznámku pro agenturu
  - Vše ostatní vyjadřují vazby (jaký kurz, jaký termín,…)
- Atributy, které vyjadřuje vazba do entitní množiny nemodelujeme!
- Ale v databázi je tam samozřejmě uvést musíme, jako cizí klíče

Část druhá – ERD - návrh



Část druhá – ERD - návrh

- Obdobně vytvoříme atributy pro zbytek entitních množin
- Držíme se zásady, že nemodelujeme atributy, které vyjadřují vazby
- Pro příklad ukázka Klientů, Termínů a Míst

Část druhá – ERD - návrh

Část druhá – ERD - návrh

- **FAQ**
- U faktur je 1..n protože se může vystavit faktura a pak dobropis
- Zaměstnanci a maily sice nemají žádnou vazbu, ale obojí v systému potřebuji
- Emaily – seznam mailů, které se mají odeslat
- Zaměstnanci – to je snad jasné, ne?
- Vazby čteme od místa, kde je 1

Část druhá – ERD - návrh

- Dotazy?

Část druhá – ERD - návrh

- Diskuze

Část druhá – ERD - návrh

- Při tvorbě těchto slajdů jsem čerpal z:
  - Slajdy k předmětu IUS
  - Zápiskům z přednášek předmětu IUS a IDS
  - Knížky o UML která už nevím, jak se jmenuje
  - Konzultací s odborníky
  - Webů:
    - http://www.agilemodeling.com

Část druhá – ERD - návrh

- Za pomoc při prezentaci děkuji:
  - Ing. Bohuslavu Křenovi, Ph.D. za možnost uskutečnit tohle cvičení
  - Paní Duránikové za rezervaci místnosti a pánům od AV techniky za podporu
  - Mé rodině, že mě podporovala
  - Křečkovi, kterého nemám
  - Přítelkyni, kterou mám ☺
  - Autorům neoficiálního, leč oblíbeného studentského fóra, bez kterého bychom se nesešli

Část druhá – ERD - návrh

- Díky za pozornost ☺

Mějte se hezky