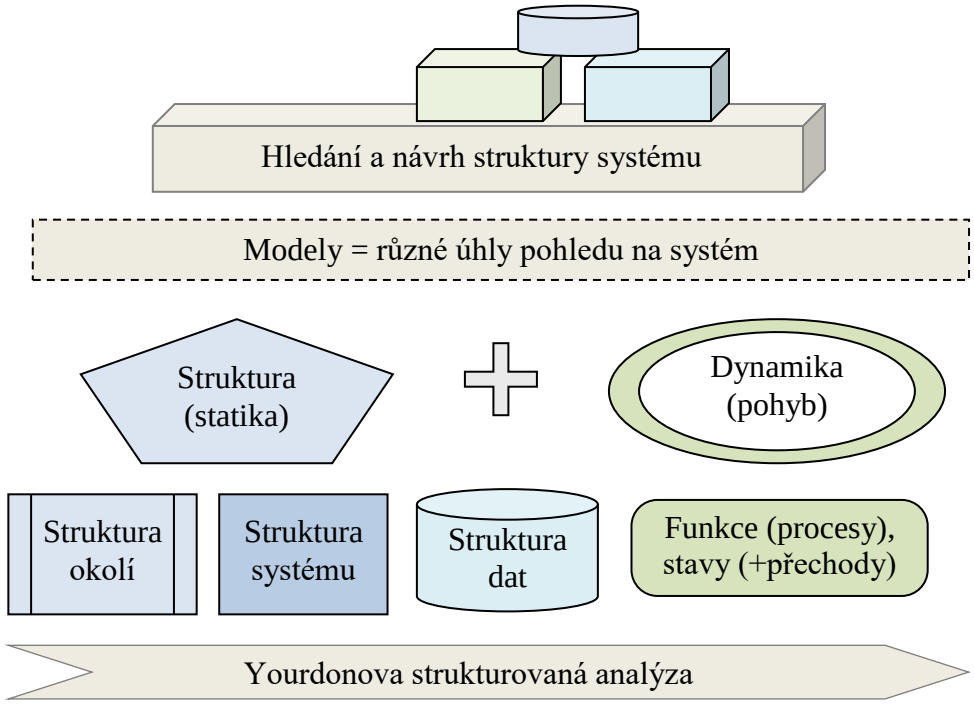
Předmět:	**Projektování IS**
Téma:	**Objektový přístup k vývoji IS (část A)**

Vyučující:	dr. Dušan Kajzar	Školní rok:	2020/2021

Obsah:

Opakování tématu „Strukturovaný přístup ... “



1. Úvod k objektově orientované analýze a návrhu IS

Objektově orientovaný (stručně „objektový“) přístup:

- principy „strukturování“ - zůstávají zachovány,
- ústřední pojem = objekt.

Objekt a systém:

- objekt = prvek, zapouzdřující vlastnosti + chování,
  - vlastnosti jsou dány – daty objektu, tj. hodnotami jeho atributů,
  - chování je dáno – funkcemi (operacemi, metodami), prací s daty,
  - příklad - objekt Student Josef Novák (ID=21, JM=Josef, PRI=Novák, DTN=2.2.2002, ...),
- systém = množina spolupracujících objektů,
  - objekty vznikají, pracují, reagují na události, zanikají,
  - příklad - systém IS SU (studenti, vyučující, předměty, rozvrhy, ...).

Poznámka k historii vzniku objekt. přístupu:

- historicky mladší - 2. pol. 80.let 20. stol.,
- objektové programování => objektová analýza a návrh.

Hlavní příčiny úspěchu objekt. přístupu:

- řízení systému --> událostmi (řešení okenních rozhraní),
- úlohy dávkové --> úlohy interaktivní,
- knihovny tříd --> znovu použitelné,
- porušení zásad --> syntaktické chyby hlášené překladači.

Metody objektového přístupu:

- 90. léta 20. stol.
  - množství metod - z vývojové praxe i akademické půdy,
  - nejvýznamnější - Booch (OOD), OMT, OOSE,
- snahy po sjednocení metod v jednu univerzální
  - Unified Method (Booch, Rumbaugh),
  - nevedlo k úspěchu => transformace na UML.

UML (Unified Modeling Language):

- grafický jazyk pro modelování IS,
- Booch, Rumbaugh, Jacobson,
- notace UML – modely, grafické znaky.

Podpora UML:

- OMG (Object Management Group),
- www.omg.org, www.uml.org,
- trh CASE nástrojů - integrace UML.

V současnosti:

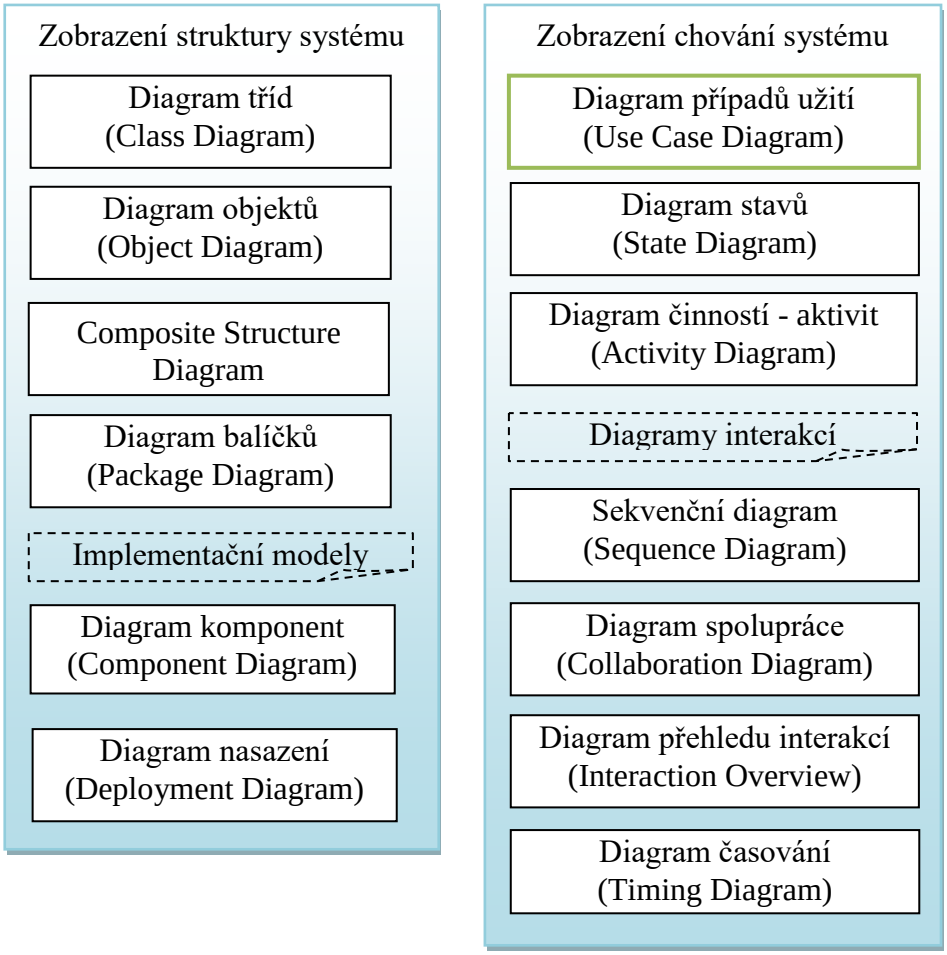
- UML
  - standard v oblasti notace modelování,
  - použití i mimo oblast vývoje IS (např. modelování procesů, workflow),
  - => znalost UML – požadavek mnoha pracovních míst,
- metodiky a metody vývoje IS
  - existují různé metodiky a metody,
  - odlišnosti metod - notace, zaměření na etapy vývoje, okruh úloh,
  - dnes nejvýznamnější – RUP, UP,
- metodiky a metody nabízí
  - „klasické“ prostředky pro boj se složitostí,
  - množinu modelů + postupy jejich tvorby.

Důležité pro studium i praxi:

- osvojení tzv. „objektového“ způsobu myšlení,
- množina modelů UML = dílna nástrojů vývojáře,
- dávejte si do souvislosti UML + znalosti z OO programování!

2. Modely objektového přístupu k analýze a návrhu IS

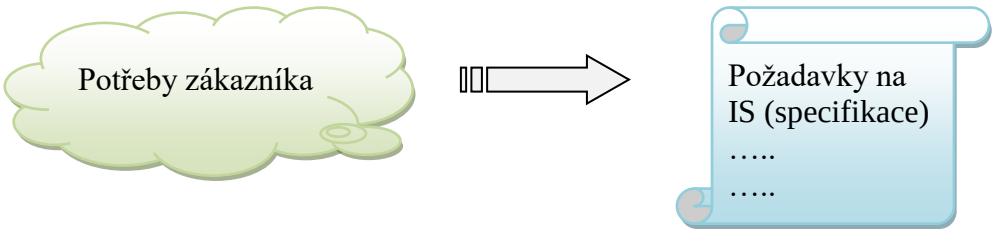
Přehled modelů objektově orientované analýzy a návrhu IS:



Použití modelů k zobrazení:

- stávajícího stavu IS – k porozumění současné situaci,
- cílového stavu IS - návrh nového (inovovaného) systému.

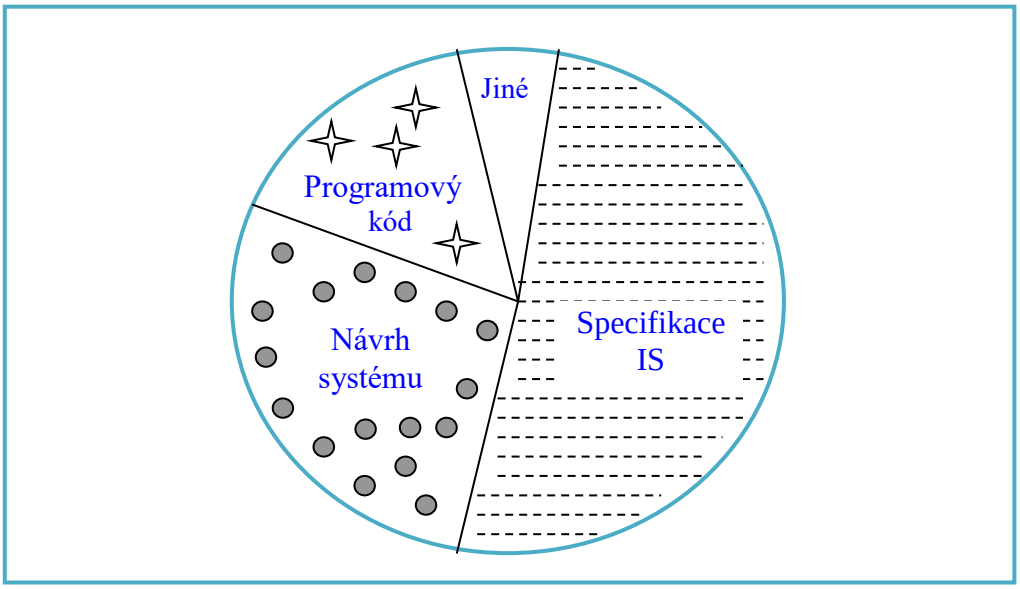
3. Potřeby zákazníka a požadavky na IS



Obrázek: Potřeby zákazníka a požadavky na systém

Inženýrství požadavků:

- proces získávání a dokumentování požadavků na systém,
- analýza zákazníkových potřeb => specifikace požadavků.



Obrázek: Zdroje chyb v SW (podle Patton, R.: Testování SW, Computer Press, 2002)

Požadavky jako základ tvorby systémů:

- vyjadřují **„CO“** by měl systém dělat,
- nemíchat dohromady s otázkou „JAK?“

Požadavky funkční a mimofunkční:

- funkční – týkají se uživatelských funkcí,
- nefunkční (mimofunkční), např.:
  - výkon, doba odezvy, vytíženost, dostupnost, spolehlivost,

- zabezpečení, důvěrnost dat,
- integrace s jinými IS, provozní standardy,
- technická omezení, např. nasazení na dané platformě,
- rozpočtová omezení, organizační požadavky, zákonné požadavky.

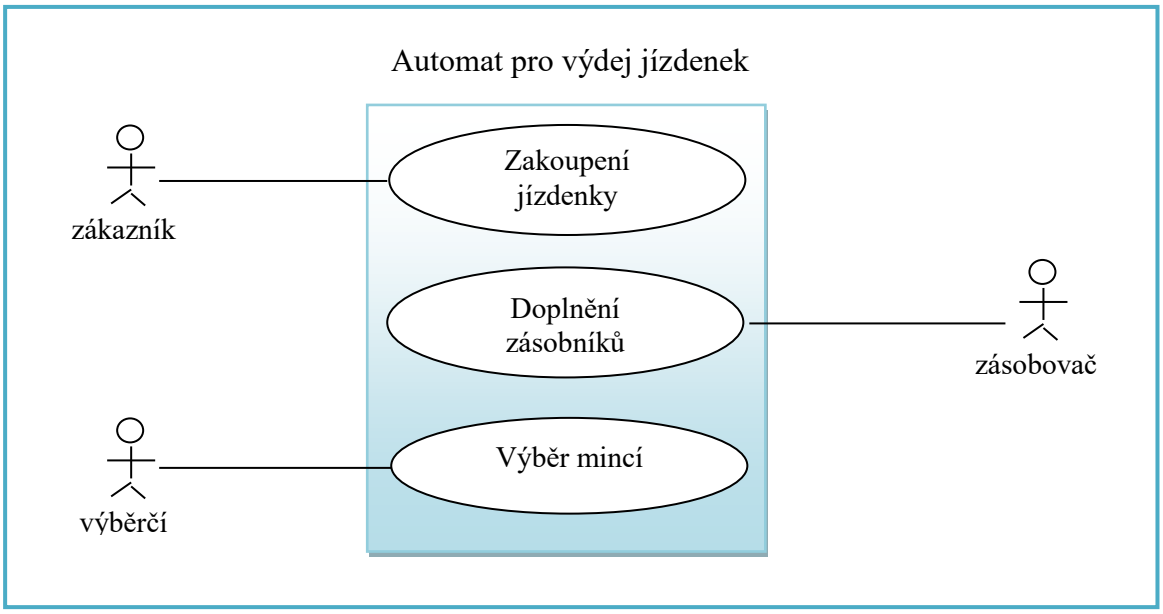
4. Diagram případů užití (Use Case Diagram)

Účel modelu „Use Case“:

- zachycení funkčních požadavků na systém,
- kdo a co od systému očekává,
- kdo a k čemu bude systém používat.

Základní prvky modelu:

- řešený systém,
- aktér (aktor, participant, účastník),
- případ užití (use case),
- scénáře případů užití.

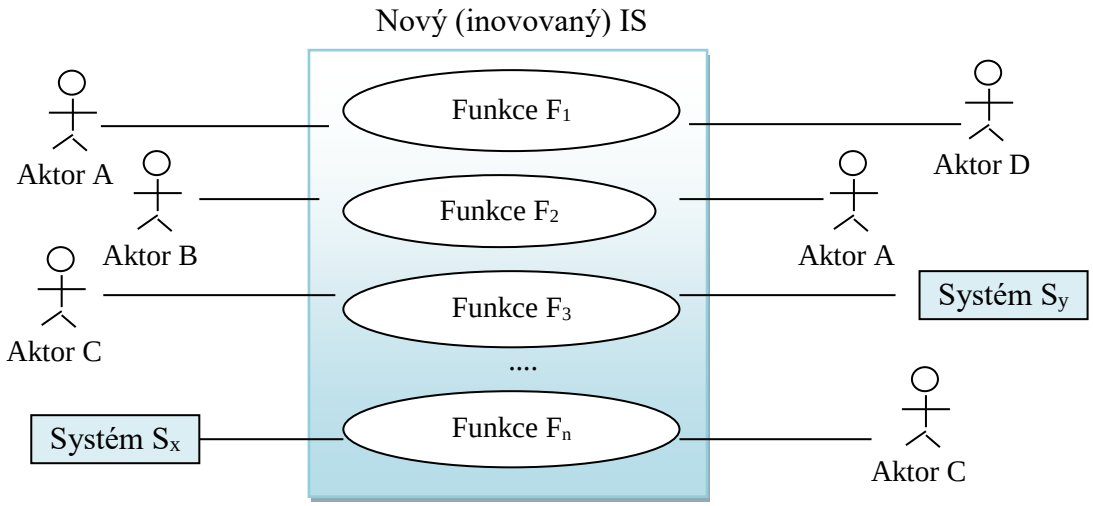


Obrázek: Příklad diagramu případů užití

Poznámka k pravé a levé straně:

- levá strana diagramu – aktivní aktéři (iniciátoři akce),
- pravá strana diagramu - příjemci výstupů,

- resp. nerozlišujeme význam levé a pravé strany diagramu.



Úkoly vývojáře při tvorbě Use Case:

- nalezení hranic systému (odhad hranic + přesvědčit se, že to tak je),
- vyhledání (určení) všech aktérů,
- zobrazení vazeb (relací) „aktéři <-> případy užití“,
- popis případů užití (scénáře a alternativní scénáře).

Struktura popisu případu užití (scénáře):

- aktér - iniciátor případu užití,
- vstupní podmínky,
- kroky scénáře (sekvence, větvení, cyklus),
- výstupní podmínky - ukončují případ užití,
- aktér - příjemce výsledku.

Scénář případu užití:

Případ užití: Zakoupení jízdenky
ID=AS-B-01
Stručný popis: Cílem scénáře je obsloužit zákazníka dopravního podniku. Po volbě typu a zaplacení jízdného zákazník obdrží jízdenku.
Hlavní aktéři: Zákazník
Vedlejší aktéři: Žádní.

Vstupní podmínky: Automat je ve stavu Ready.
Hlavní scénář: 1. Zákazník si zvolí typ jízdného dle nabídky na panelu. 2. Automat vypíše požadovanou výši Kč. 3. Zákazník vhodí mince povolené hodnoty do vstupního zařízení automatu. 4. Automat ověří vloženou sumu Kč. 5. Pokud .... 5.1 ... 5.2 ... 6. .... „n“ Zákazník odebere z výstupu jízdenku.
Výstupní podmínky: Zákazník obdržel jízdenku.
Alternativní scénáře: Zákazník požaduje storno akce. Zákazník vhodil neplatnou minci. ....

Větvení ve scénáři:

- hlavní tok událostí (ideální případ),
- vyjádření odchylek
  - rozvětvení scénáře „Když ...“,
  - pomocí alternativních scénářů.

Hledání alternativních scénářů:

- alternativy hlavního postupu,
- reakce na chyby, reakce na přerušení.

Přesné formulace ve scénářích:

- nikoliv jen „Jsou zadávány údaje o zákazníkovi.“,
- nýbrž „Kdo zadává?“, „Co? Kam? Jak ?“,
- schéma: <id> <systém> bude <mít funkci>.

Vyjádření pochybností:

- vyjádřete své pochybnosti u kategorických formulací zadavatele,
- u kvantifikátorů: všichni, každý, vždy, nikdy, nikdo, žádný, ...

- např.: „Každý, kdo si chce půjčit knihu, musí mít průkazku.“, „Opravdu každý? Neexistuje nějaký výjimečný případ?“

Omezení případů užití:

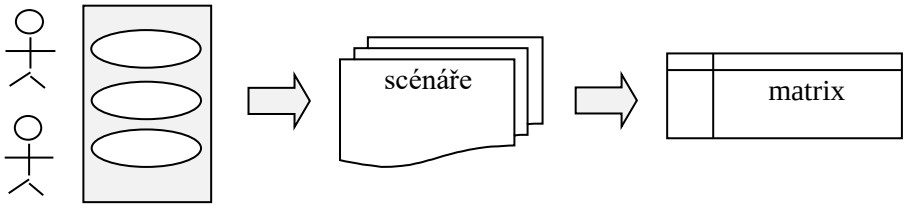
- zachytí funkční požadavky na systém,
- nezachytí nefunkční požadavky.

Mapování případů užití na požadavky:

- matice propojení (Requirements Tracebility Matrix),
- požadavky a jejich zahrnutí v případech užití.

Požadavky / Případy užití	P01	P02	P03	P04	P05	P06	....
UC01	+				+	+	
UC02		+	+				
UC03				+		+	
....							

Dílčí shrnutí základních částí Use Case diagramu:



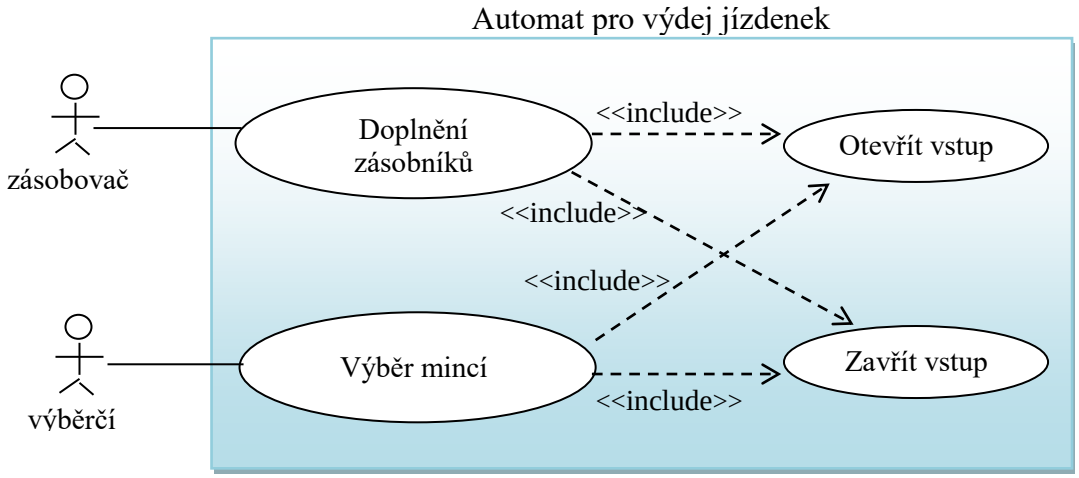
5. Pokročilé konstrukce v Use Case Diagramu

Pokročilé konstrukce s případy užití:

- vkládání <<include>>
- rozšíření <<extend>>
- zobecnění – aktérů, případů užití,
- seskupování.

Vkládání <<include>> :

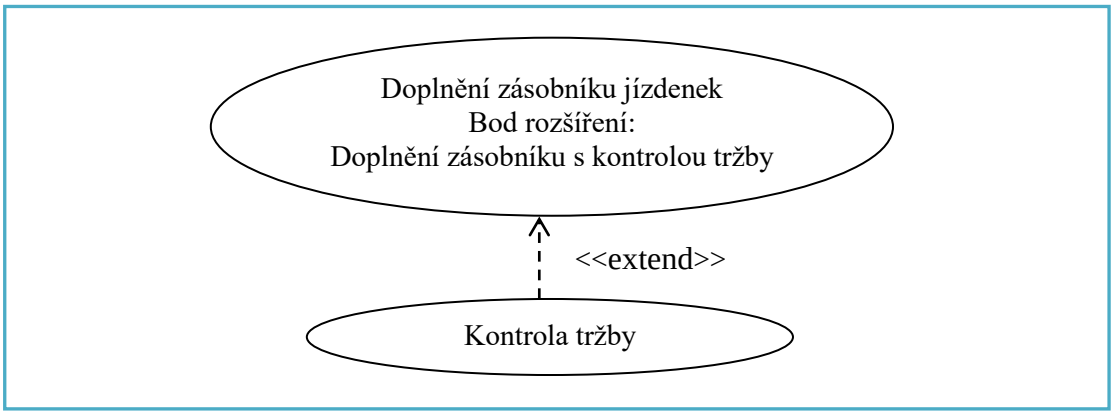
- bázový use case (klient) - není úplný,
- dodavatel funkcionality - úplný, neúplný,



Obrázek: Vkládání <<include>> případů užití

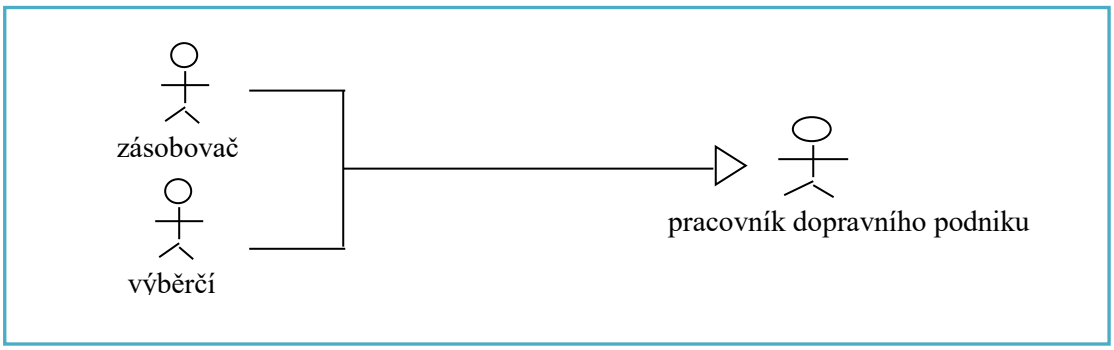
Rozšíření <<extend>> :

- bázový use case - vždy úplný,
- rozšiřující use case - úplný, neúplný,

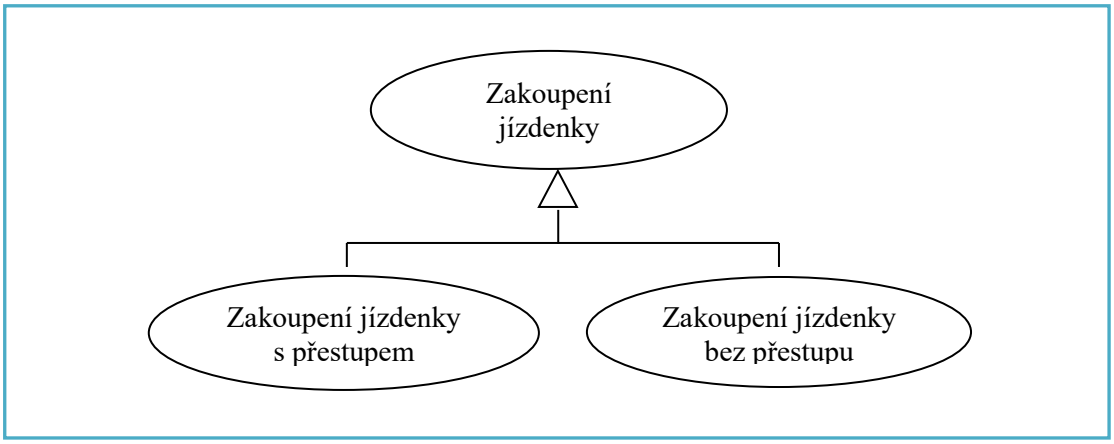


Obrázek: Rozšíření <<extend>> případů užití

Zobecnění – aktérů, případů užití:

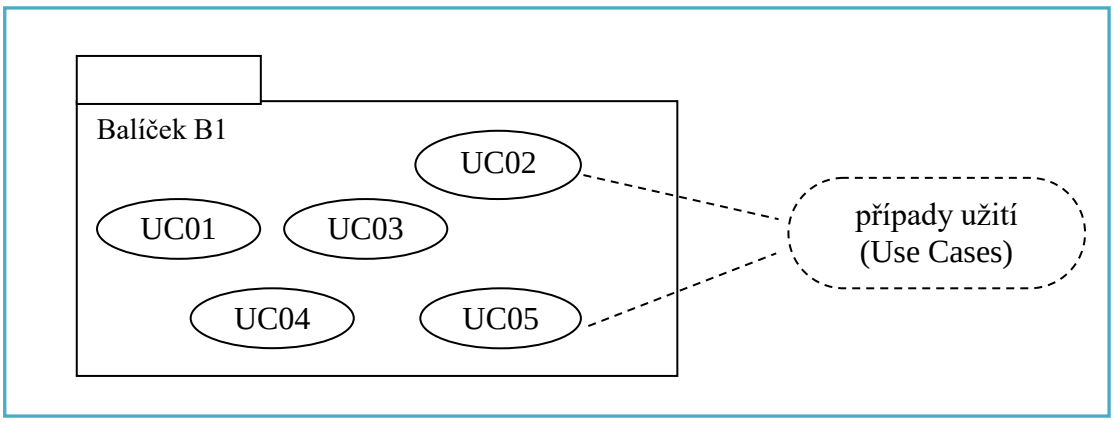


Obrázek: Zobecnění aktérů



Obrázek: Zobecnění případů užití

Seskupování:



Obrázek: Seskupování - Balíček

6. Shrnutí k Use Case Diagramu

Shrnutí k Use Case Diagramu:

- diagram -> scénáře -> matrix,
- zobrazuje dynamický pohled na systém zvnějšku,
- k popisu hranice a komunikace mezi systémem a jeho okolím,
  - externí entity (aktéři) – životní i neživotní,
  - rozčlenění aktérů do skupin (podle používání systému),
- k pochopení chování systému vzhledem k jeho okolí,
  - základní členění systému na případy užití,
  - popis slovními scénáři případů užití.

Use Case Diagram NENÍ určen pro:

- znázornění vzájemné komunikace aktérů,
- zachycení „vnitřku“ systému
  - implementace (realizace) případů užití („Jak?“),
  - realizace případů užití interními procesy (funkcemi) systému,
  - zobrazení vnitřních stavů systému a změn stavů,
- podrobný funkční rozklad systému na subsystémy.

Poznámky a doporučení:

- cílem je vytvořit srozumitelný model (uživatelům i vývojářům),
- vyhýbat se pokročilým konstrukcím, nepřispívají-li srozumitelnosti,
- scénáře popisují
  - „Co“ od systému očekávají aktéři,
  - nikoliv „Jak“ to systém realizuje,
- nejde o zpracování funkční dekompozice (!)

Use Case Diagram je také podkladem pro:

- návrh uživatelského rozhraní IS,
- plán a návrh testů.

Proč je Use Case vhodný jako podklad ke zde uvedeným činnostem?