



UML (1)

úvod, requirements management

© 2008 Radek Ošlejšek
FI MU Brno

oslejsek@fi.muni.cz
<http://www.fi.muni.cz/~oslejsek/PA103>



Tijuana “shantytown”

a vize ...



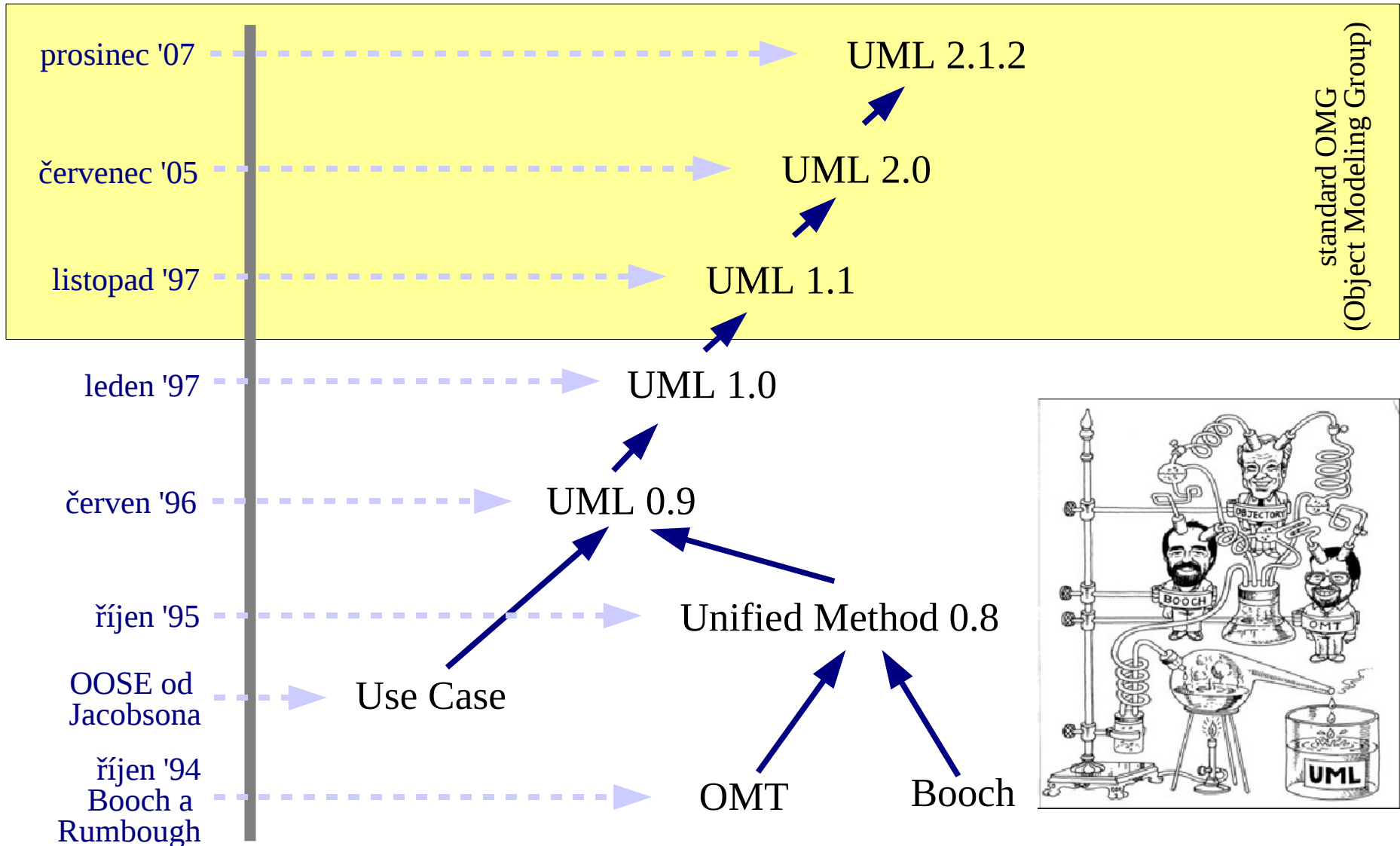
Fallingwater



- **Formální modelovací techniky**
 - např. Z, VDM
 - COOPN, Petri-Nets
 - Umožňují různé druhy kontrol, ověřování
 - Cílem je úplnost
 - Použity především pro modelování ústředních částí systému
- **Semiformální modelovací techniky**
 - např. Booch, UML, SysML
- **Neformální modelovací techniky**
 - RDD - Requirement Definition Document,
 - CRC analýza ,
 - Analýza textů,
 - Scénáře,
 - ...

Všechny mají své místo na slunci !

Historie UML



Co je to UML? (reklama)



- UML = **U**nified **M**odeling **L**anguage
- UML je **standard** OMG
- UML může být použit ve všech fázích životního cyklu vývoje a pro různé technologie implementace. Kombinuje vše nejlepší z
 - Modelování výrobních procesů (toky práce) – viz *Procesní řízení*
 - Konceptů datového modelování (entitně-relační diagramy) – viz *ANANAS*
 - Modelování objektů
 - Modelování komponent
- UML je převážně grafický jazyk
 - Grafické vyjádření napomáhá při pochopení, dokumentaci a vysvětlení problému
- UML je sada nástrojů, nedefinuje postup jak je využít (to je věc zvolené metodiky)
- Důležitým předpokladem pro úspěšný vývoj softwaru je:
 - Zvládnutí modelovacích technik (UML)
 - Zvládnutí jednoho nebo více procesů vývoje softwaru (metodiky)
 - Znalost nejdůležitějších návrhových vzorů

Proč potřebujeme jednotný standard?



| | OOSE | OOD | OMT | UML |
|----------------------------|--|-----|-----|-----|
| třída | <p>entitní objekt objekt rozhraní kontrolní objekt</p> | | | |
| objekt | <p>entitní objekt objekt rozhraní kontrolní objekt</p> | | | |
| asociace | | | | |
| generalizace/ dědičnost | | | | |
| use case vztahy | | | | |
| use case, aktor | | | | |
| use case | | | | |



UML lze použít pro:

- zobrazení hranice systému a jeho hlavních funkcí pomocí případů užití (use cases) a účastníků (actors)
- ilustrovat realizaci případů užití pomocí diagramů interakcí
- reprezentovat statickou strukturu systému pomocí diagramů tříd
- modelovat chování objektů pomocí stavově-přechodových digramů
- odhalit fyzickou implementační architekturu pomocí diagramů zapojení komponent
- rozšířit funkcionalitu pomocí stereotypů

UML – univerzální modelovací nástroj



Použití UML pro modelování podnikatelských procesů:

- při zahájení softwarového projektu pro vytvoření pevného základu
- pro zpětné inženýrství podnikatelského procesu

Použití UML pro modelování ontologií:

- sémantický web, expertní systémy, ...
- UML je „podtřídou“ OWL – Web Ontology Language

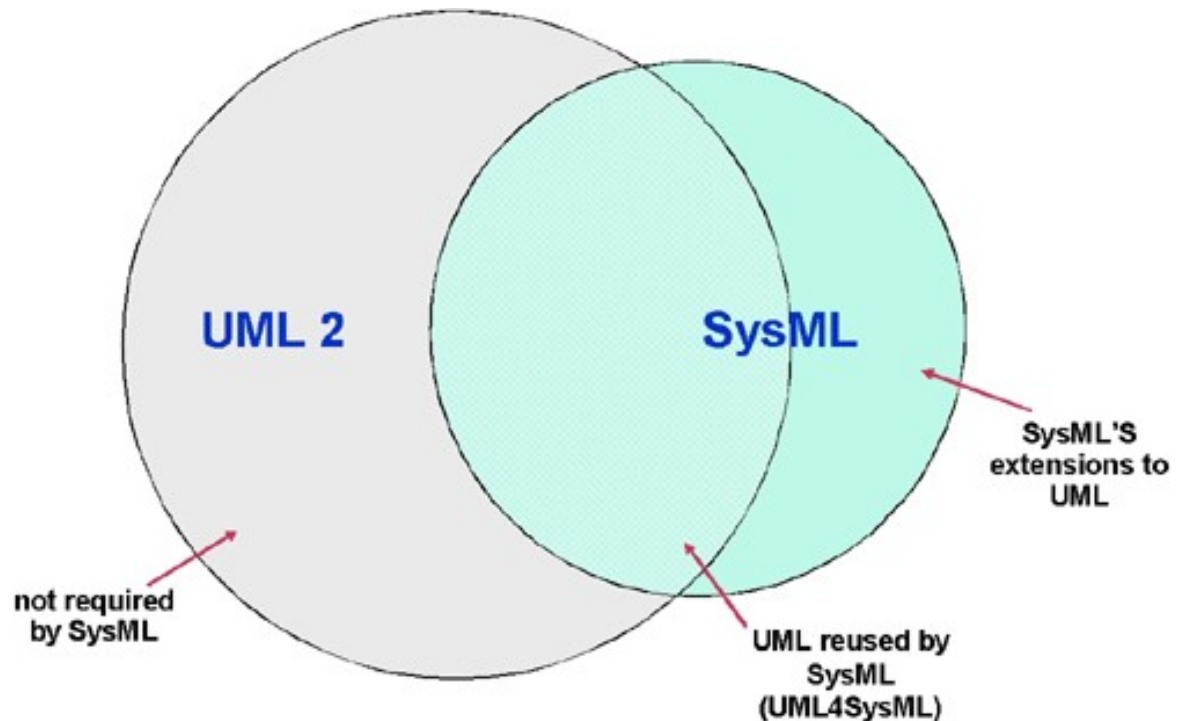
⇒ Stejné diagramy pro různé účely, různé úrovně abstrakce, různé předmětné oblasti.

UML není jediný modelovací jazyk – viz např. SysML

SysML - Systems Modeling Language



- domain-specific modeling language for systems engineering applications
- language to specify complex systems that include non-software components (e.g., hardware, information, processes, personnel, and facilities)
- based on UML
- OMG standard





Modely ukazující **statickou strukturu** systému:

- diagramy tříd, balíčků a objektové diagramy
- implementační diagramy: diagramy komponent, diagramy rozmístění

Modely ukazující **dynamické chování systému**:

- model případů užití \Rightarrow externí pohled na systém
- diagramy aktivit \Rightarrow externí/interní pohled na systém
- interakční diagramy: diagramy sekvencí, komunikace a časování
diagramy spolupráce \Rightarrow interní pohled na systém

Modely ukazující **dynamické chování jediné třídy**:

- stavové diagramy, diagramy aktivit

Vymezení problémové oblasti

Use Case modelování

Problém zachycení požadavků

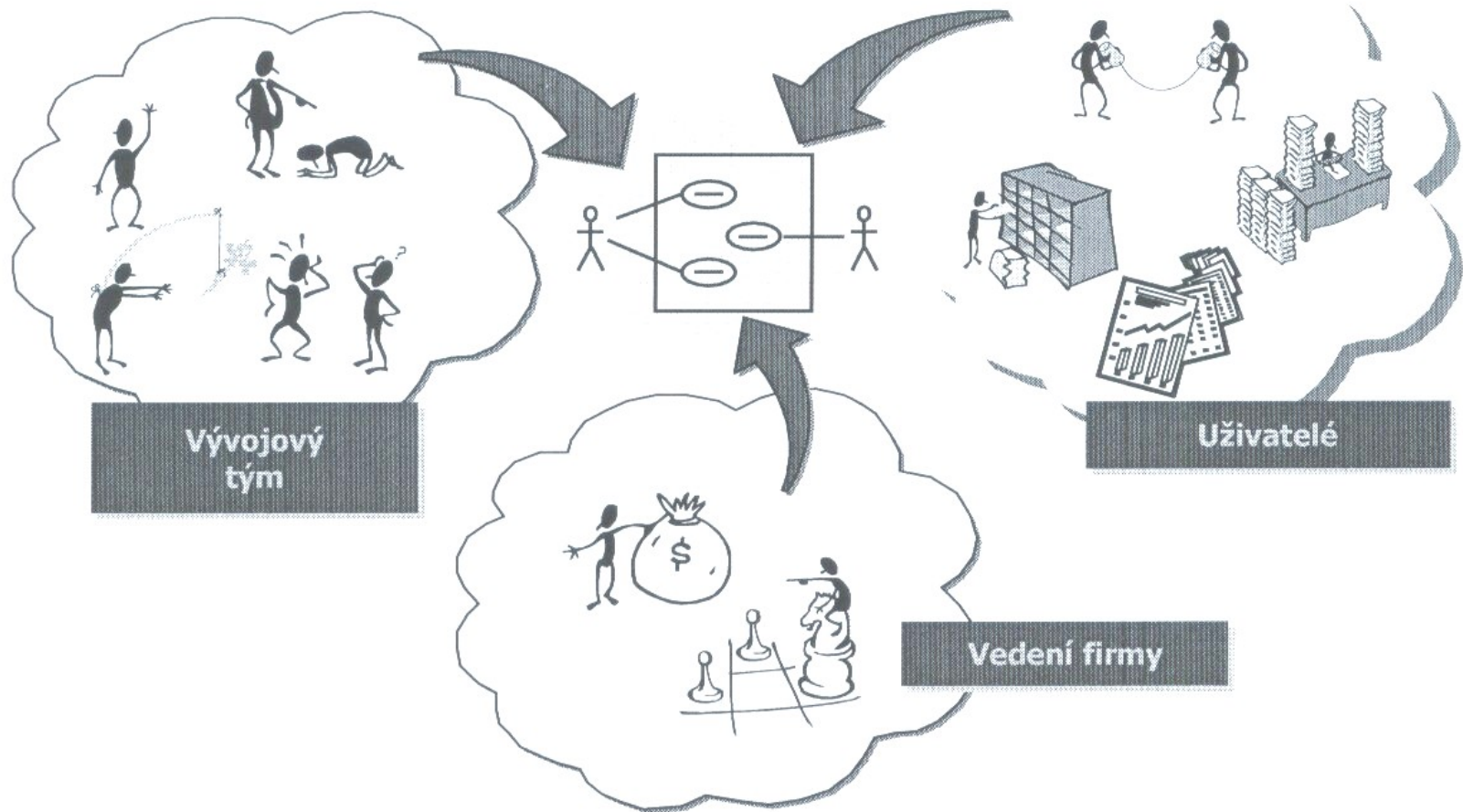


Diagram případů užití



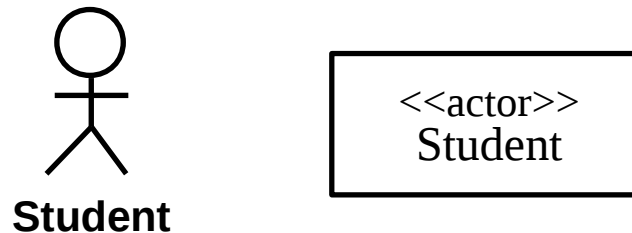
- Další názvy: *Use Case Diagram, UCD*
- Co potřebujeme:
 - Chceme zachytit potřeby a požadavky uživatelů
 - Potřebujeme efektivní mechanismus komunikace mezi uživateli a vývojáři
- Co UCD přináší:
 - Prostředek k vyjádření konkrétních cílů, které chce uživatel dosáhnout, prostřednictvím sekvence interakcí se systémem
 - Grafické znázornění je dostatečně jednoduché a čitelné i pro laiky
 - Dokumentace jednotlivých případů užití umožňuje detailně popsat konkrétní interakce



Havarijní pojištění

- Zákazník (pojištěnec) si prostřednictvím webových stránek prohlíží nabídku havarijního pojištění. Prostřednictvím webu má možnost zadat údaje o sobě, autě a parametry pojištění a pak odeslat žádost o vytvoření pojistné smlouvy. Žádost je posouzena pracovníkem odd. smluv a v případě souhlasu s vytvořením smlouvy dostane zákazník poštou dvě kopie smlouvy a složenku. Jednu podepsanou kopii pošle zákazník zpět pojišťovně. Platby za pojištění jsou v systému evidovány opět pracovníky odd. smluv.
- Pojištěnec má dále možnost prostřednictvím webu zadávat pojistné události. Pojistná událost je vždy převzata likvidátorem škod (zaměstnanec pojišťovny), který se spojí s pojištěncem, vyřizuje škodní událost a výsledky zadává do systému.
- Pojišťovna spolupracuje s pojišťovacími agenty. Ti nejsou přímo zaměstnanci pojišťovny, ale spolupracují na základě smlouvy. Slouží pro podporu zákazníků, tj. pomáhají jim s vytvářením smluv, jejich změnami a s vyřizováním pojistných událostí. Agenti proto mohou zadávat žádosti namísto zákazníků. Jsou pak odměňováni na základě počtu uzavřených smluv a vyřízených pojistných událostí. Oproti zákazníkům mají navíc možnost zadávat žádosti o změnu smlouvy. Tyto žádosti o změnu jsou opět posouzeny pracovníkem oddělení smluv a vyřízeny podobně, jako žádosti o novou smlouvu.
- Management pojišťovny bude mít možnost spravovat ceníky pojištění, získávat statistiky o uzavřených smlouvách a statistiky pojistných událostí.
- Systém obsahuje heuristiky, které dokáží odhalit podezřelé události, např. Neobvykle vysoký počet škodných událostí zákazníka. V případě detekce takovýchto událostí je neprodleně informováno vedení prostřednictvím e-mailu.

- Další názvy: *účastník*, *actor*
- Entity vně hranic systému, které interagují se systémem.
- Dvě možnosti grafického znázornění („panáček“ je jednoznačně nejpoužívanější):

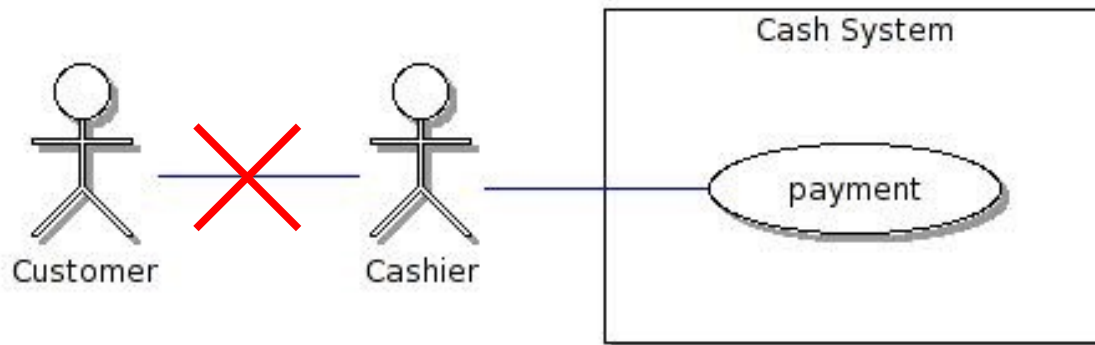


- Modelují **role**, ne konkrétní osoby!
 - v jedné roli může vystupovat více osob
 - např.: Jan Novák a Karel Zeman jsou oba *zákazníci*.
 - jedna osoba může hrát v systému více rolí
 - např.: Jan Novák je zároveň *vedoucí skladu a kontrolor*.
- Nejen uživatelé, ale i **externí systémy**, **části hardwaru** apod.
- Od UML 2 i **ostatní subjekty** => propojení UC modelů
- Aktérem může být i **čas**
 - např.: automatická záloha každý večer

UCD: Aktéři (II)



- Aktéři jsou externí, často se ale objeví i jako objekt uvnitř systému
 - např.: aktér *zákazník* komunikuje se systémem, objekt *zákazník* si uvnitř systému pamatuje detaily (jméno, adresa, ...)
- Modelujeme jen interakci mezi účastníky a systémem, nikoli interakce mezi účastníky



Otázka: Kterého aktéra ponechat?

Odpověď: Kdo je zodpovědný za správnost dat?

Kdo fyzicky interaguje (komunikuje) se systémem?

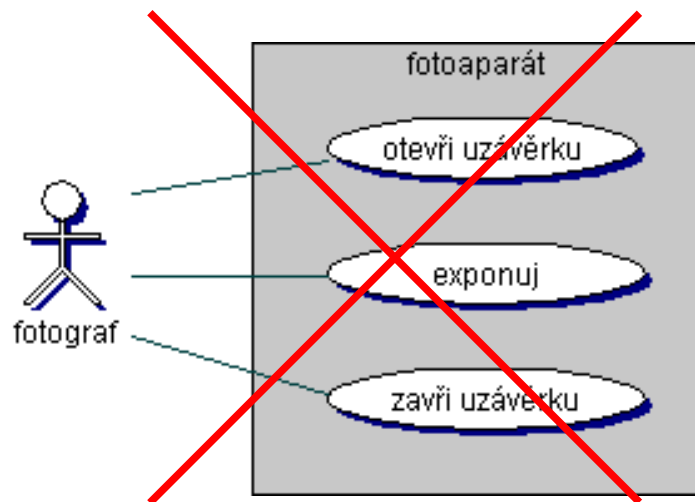


- Více aktérů pro jeden případ užití?
- Ano, ale:
 - Příklad užití je „přístupný“ více uživatelským rolím, pak je vhodné zamyslet se nad
 - vhodnějším rozdělením rolí,
 - dědičností mezi aktéry
 - Na řešení případu užití se podílí více rolí (uživatelů) současně
 - ve skutečnosti se za tím často skrývá několik nezávislých navazujících interakcí (=> dekompozice při následném upřesňování hranic systému)
- Jednosměrná komunikace mezi aktérem a případem užití
 - Nejčastěji ze systému směrem ven
 - Vhodné pro „pasivní“ zařízení/systémy a zajištěné spolehlivé komunikační protokoly
 - Používá se velmi zřídka, většinou se jedná o obousměrnou interakci

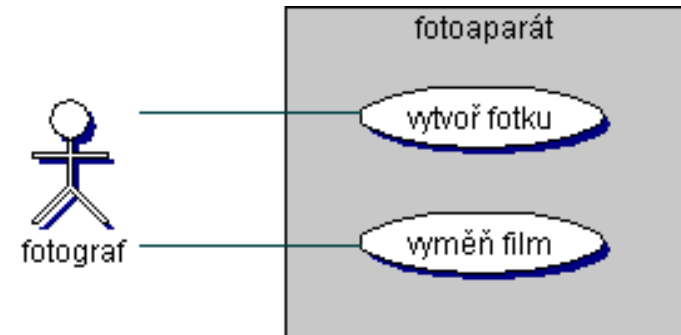
UCD: Případy užití



- Další názvy: *Use Case*, *typová úloha*, *uživatelská transakce*, ...
- Koherentní jednotky funkcionality poskytované systémem
 - Případ užití = konkrétní interakce, která vede ke splnění cíle
- Jsou vždy **zahájeny účastníkem**, vždy **napsány z pohledu účastníka!**
 - případy užití spuštěné časovými událostmi => účastník čas
- Jméno p.u. by mělo **vyjadřovat cíl**, kterého chce aktér dosáhnout



špatně – fotograf tyto činnosti nedělá izolovaně, ale jsou mu poskytnuty systémem jako jedna funkcionality “vytvoř fotku”



správně – fotograf vyvolává tyto činnosti a systém na ně reaguje

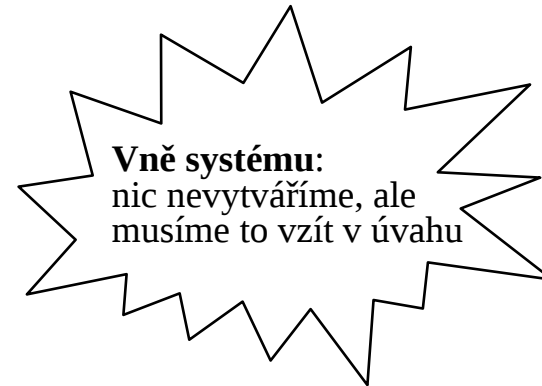
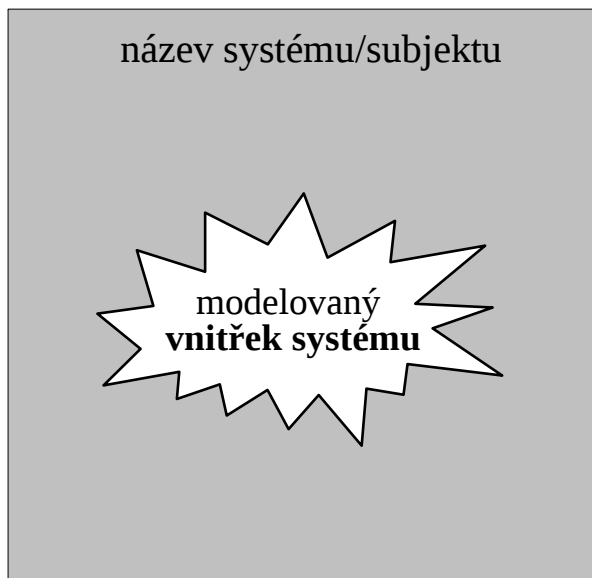


- Záleží na použité metodice, obecně můžeme použít tyto strategie:
 - Každému elementárnímu procesu z procesního modelování (viz přednáška *PV165: Procesní řízení*) odpovídá jeden případ užití.
 - Procházíme seznam účastníků a zvažujeme způsob, jakým bude každý z nich systém používat.
 - Jaké funkce jednotliví účastníci od systému očekávají?
 - Bude systém uchovávat a poskytovat informace? Pokud ano, jací účastníci budou tyto činnosti aktivovat?
 - Jací účastníci budou upozorněni na změnu stavu systému?
 - Existují nějaké vnější události, které ovlivňují systém? Co upozorní systém na tyto události?

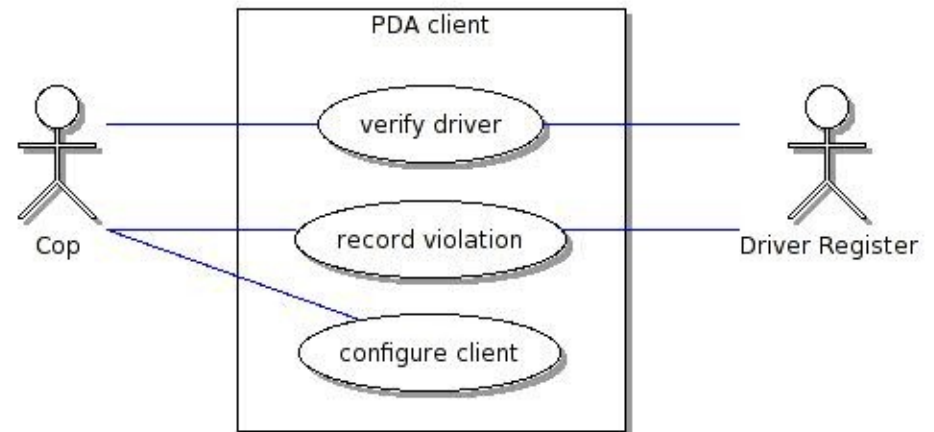
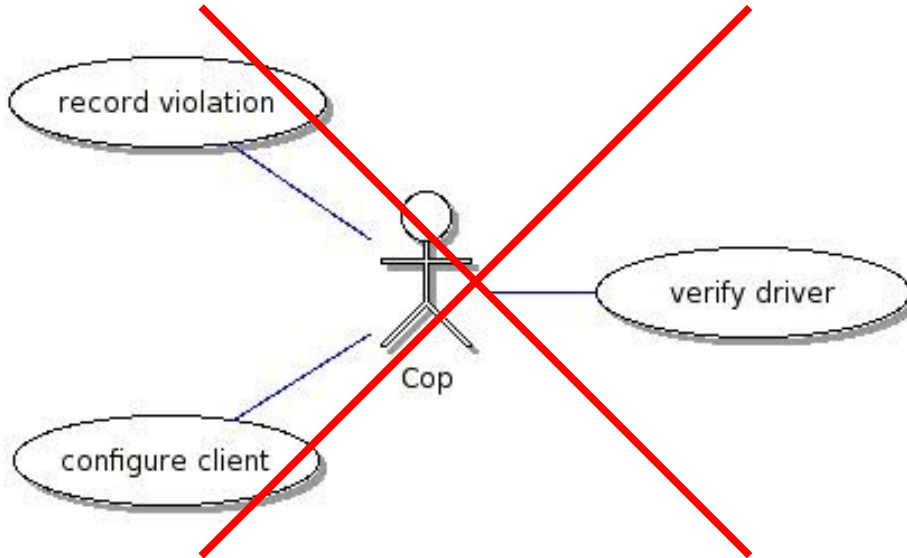
UCD: Subjekt



- Další názvy: *System Boundary* (používáno před UML 2.0)
- Je možné vytvářet více UCD pro různé úrovně abstrakce a různé části systému a propojovat je
- Často si ale vystačíme s jediným UCD (důležitá je přehlednost a čitelnost)



UCD: Subject (II)

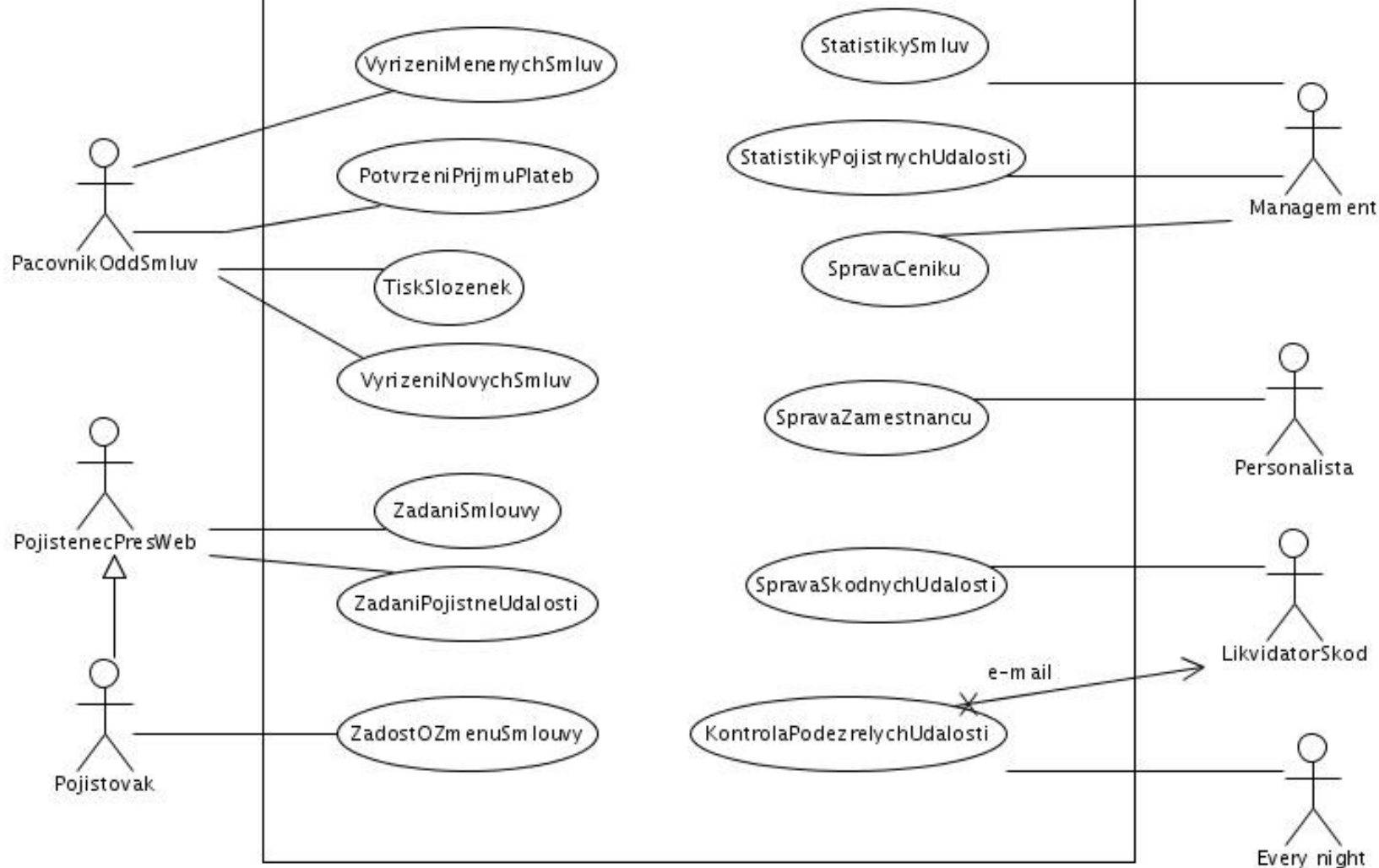


Pojišťovna: UCD



Visual Paradigm for UML Standard Edition (Masaryk University)

Pojišťovna

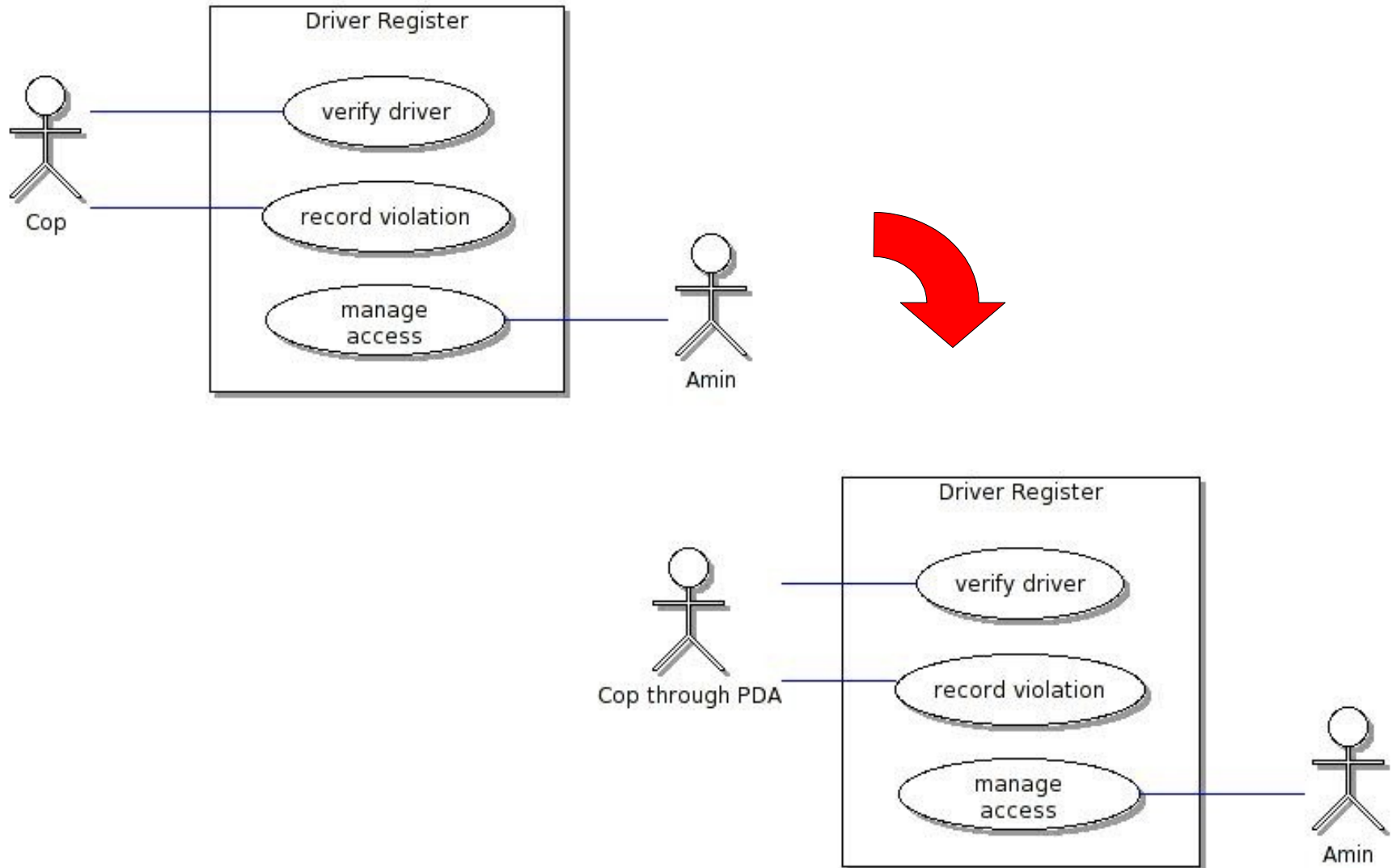


Vymezení hranic systému



- Definice hranic systému probíhá interaktivně.
- Hranice systému mohou být definovány na několika úrovních abstrakce, účastníci interakcí se mění podle úrovně abstrakce!
 - vysoká úroveň bez žádných reprezentantů
 - střední úroveň, která bere v úvahu kdo/co skutečně vkládá za informaci
 - nižší úroveň, která bere v úvahu, jaká data opravdu vstupují
- **Př. 1: platby za provedené služby**
 - Na začátku víme pouze to, že náš systém musí řešit platby za provedené služby.
 - Později si vyjasníme, že platby jsou evidovány v externím systému, který zadavatel používá. Náš systém tedy bude platby řešit přes tento externí systém.
 - Nakonec dospějeme k tomu, že komunikace bude probíhat prostřednictvím XML zpráv. Náš systém bude mít modul, který se o komunikaci bude starat.
- **Př. 2: zálohování dat**
 - Zákazník požaduje zálohování všech dat.
 - Později zjistíme, že data jsou uložena na externím databázovém serveru, který je automaticky zálohovaný a proto se o tuto činnost náš systém vůbec nemusí starat.

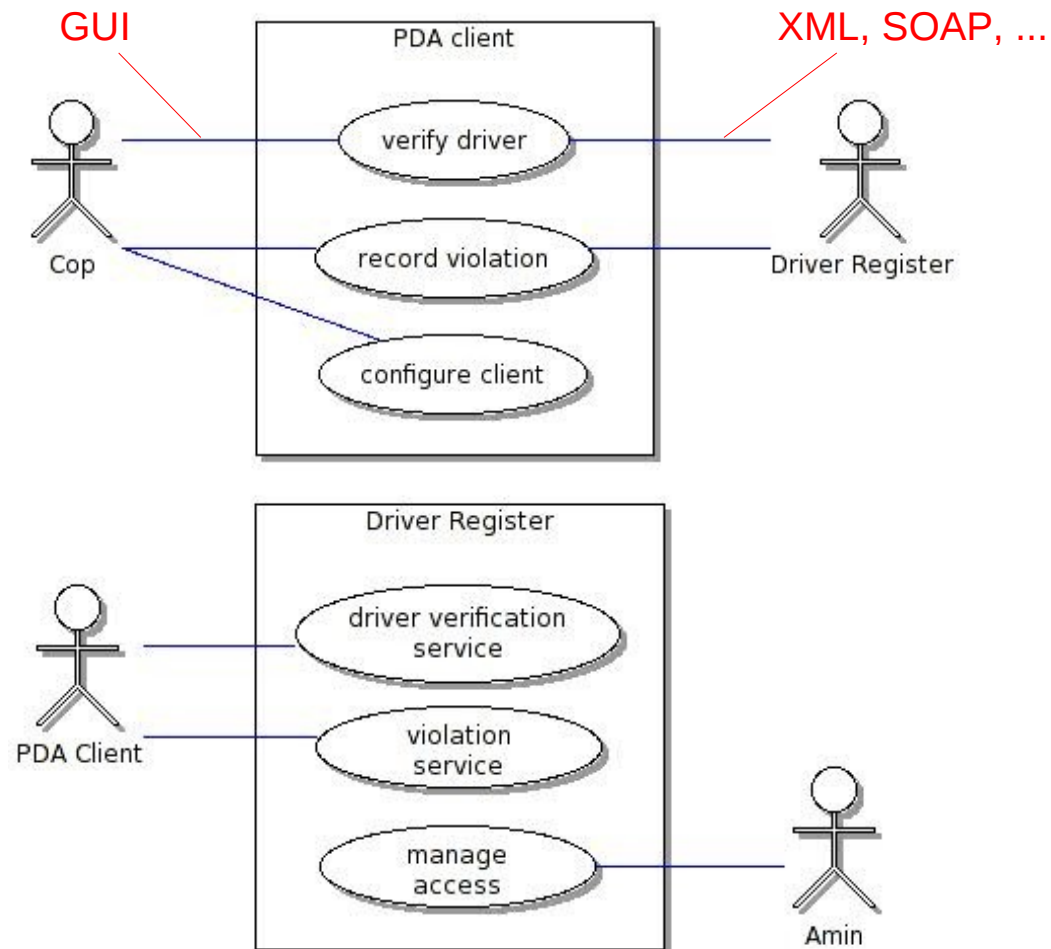
Př: Vymezení hranic systému (II)



Př: Vymezení hranic systému (III)



Na PDA běží speciální klientská aplikace, kterou musíme také vyvinout



Specifikace případů užití



- Každý případ užití musí být dokumentovaný
- Dokumentace je psána z pohledu aktéra
- Obsahuje detaily, které musí systém poskytnout aktérovi při provádění případu užití
- Neexistuje žádný standard UML pro psaní specifikace p.u. ...
- ..., ale typicky obsahuje:
 - **název** případu užití
 - **aktéři** účastníci se případu užití
 - **popis interakce** pomocí **toků událostí** a/nebo **scénářů**
 - speciální (nefunkční) požadavky, např. časová omezení
 - priority a stav vývoje ...



Další názvy: *Flow of Events*.

Dokumentace pomocí toků nejčastěji obsahuje:

- **vstupní podmínky** (preconditions)
 - kritéria, která musí být splněna před spuštěním p.u.
- **výstupní podmínky** (postconditions)
 - kritéria, která musí být splněna na konci p.u.
- **normální tok** událostí nebo interakcí
 - jednotlivé kroky v případě užití
- **alternativní a výjimečné toky** událostí

Toky událostí (II)



Jak popsat tok událostí:

- Volný text
- Číslované kroky
- Pseudo kód (obvykle příliš detailní)
- Diagramy aktivit
- Diagramy interakcí s pseudo kódem nebo textem na levé straně

Číslované kroky:

- nejčastější forma
- první krok popisuje okolnosti zahájení činnosti aktérem

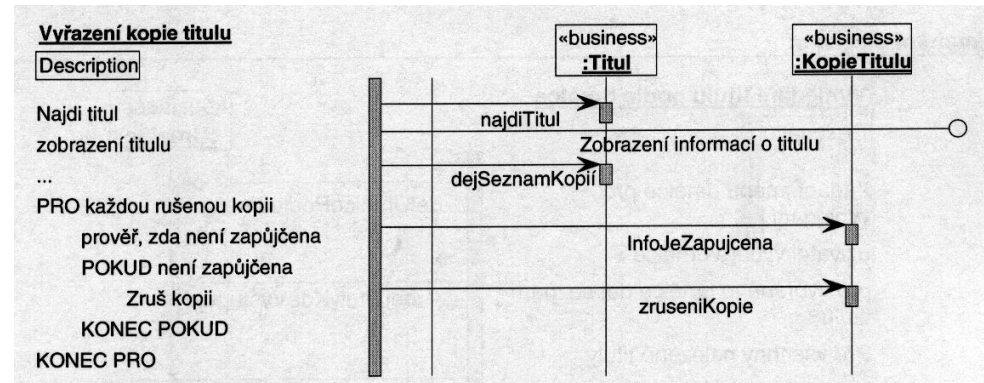


diagram interakcí s pseudokódem

1. p.u. začíná volbou “zobraz obsah košíku”
2. KDYŽ je košík prázdný
 - 2.1 systém zobrazí uživateli, že košík neobsahuje žádné položky
 - 2.2 p.u. končí
3. systém zobrazí seznam všech položek v košíku
- ...

číslované kroky

Toky událostí (III)



Jak podrobně popisovat:

- Závisí na fázi projektu (např. nejprve číslované kroky, později diagram interakcí)
- Záleží na významu a složitosti p.u. (kompromis: **hlavní tok událostí** podrobně, **alternativní toky** stručně nebo dokonce jen vyjmenovat)
- Ne moc podrobné: zachycení požadavků neznamena popis konstrukce systému ve volném textu!

1. jsou zadány údaje o ~~rušených~~ položkách objednávky

Špatně:

- **kdo údaje zadává?**
- **kam je zadává?**
- **co to jsou “údaje o rušených položkách”**

1. => p.u. začíná, když uživatel zadá do formuláře číslo objednávky
2. <= systém vypíše seznam položek objednávky
3. => uživatel vybere rušené položky...

Doporučený vzor kroků:

- **<číslo>[i/o]<něco><akce>**

Alternativní toky událostí



- Zjednodušují a doplňují hlavní tok.
- Náhrada za větvení, zejména pokud není jasné, kam větvení umístit.
- Musí začínat “booleovskou” podmínkou, která daný alternativní tok spouští!
- Měl by být co nejjednodušší.
- Každý alternativní tok musí mít svoje vlastní výstupní podmínky!

| |
|---|
| Případ užití: zobrazení nákupního košíku |
| Vstup. podmínky: zákazník je přihlášen do systému |
| Tok událostí: <ol style="list-style-type: none">1. p.u. začíná volbou “zobraz obsah košíku”2. KDYŽ je košík prázdný<ol style="list-style-type: none">2.1 systém zobrazí uživateli, že košík neobsahuje žádné položky2.2 p.u. končí3. systém zobrazí seznam všech položek v košíku |
| Výstup. podmínky: |
| Alternativní tok 1: <ol style="list-style-type: none">1. zákazník může <i>kdykoli</i> opustit obrazovku košíku |
| Výstup. podmínky: |
| Alternativní tok 2: <ol style="list-style-type: none">1. zákazník může <i>kdykoli</i> opustit systém |
| Výstup. podmínky: |

Use Case ID: VyrizeniNovychSmluv

Primary Actor: PracovnikOddSmluv

Preconditions

1. Pracovnik oddeleni smluv je prihlaseny do systemu
2. System pri zadavani zadosti overil, ze nechybi nezbytny udaje, zejmena rodne cislo zadatele

Flow of Events

1. PU zacina, kdyz PracovnikOddSmluv vybere "vyrizovat nove smlouvy"
2. System zobrazi seznam nevyrizenych smluv s odkazy na podrobnosti setrideny podle data podani
3. IF PracovnikOddSmluv klikne na konkretni pozadavek v seznamu
 - 3.1 System vypise podrobnosti zadane zadatelem
 - 3.3 IF PracovnikOddSmluv klikne na "vytvorit novou smlouvu"
 - 3.3.1 System zobrazi seznam existujicich smluv se stejnym rodnym cislem a odkazy na podrobnosti
 - 3.3.2 PracovnikOddSmluv muze prohlizet podrobnosti o existujicich smlouvach
 - 3.3.2 IF PracovnikOddSmluv kline na "vytvorit novou smlouvu"
 - 3.3.2.1 System zobrazi formular pro zalozeni nove smlouvy s preddefinovanymi informacemy z zadosti
 - 3.3.2.2 PracovnikOddSmluv vyplni pozadovane informace a potvrdi vytvoreni smlouvy
 - 3.3.2.3 System smlouvu ulozi do databaze
 - 3.3.2.4 System vytiskne dve kopie nove smlouvy
 - 3.3.2.5 System vytiskne slozenku
 - 3.3.2.6 PU pokracuje krokem 2

Post-conditions

V systemu je zalozana nova smlouva

(...pokr.)

Alternative Flow 1

1. PU zacina, kdyz PracovnikOddSmluv pracuje s konkretni zadosti a PracovnikOddSmluv se rozhodne zadost zamitnout
- 2 System zobrazi formular zpravy o zamitnuti vctne vyberu preddefinovanych oduvodneni
- 3 IF PracovnikOddSmluv vybere preddefinovane oduvodneni
 - 3.1 System vyplni preddefinovany text do do zpravy
- 4 PracovnikOddSmluv muze editovat zpravu o zamitnuti
- 5 IF PracovnikOddSmluv potvrdi zamitnuti zadosti
 - 5.1 IF zadost obsahuje e-mailovou adresu zadatele
 - 5.1.1 System posle zpravu na e-mail zadatele
 - 5.2 Zatnuti zadosti s oduvodnenim se ulozi do systemu

Post-conditions

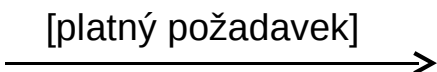
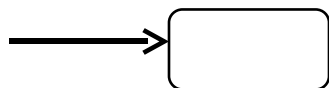
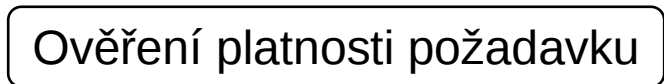
V systemu je ulozeno zamitnuti zadosti vctne oduvodneni

- Další názvy: *Scenarios*
- **Scénář:** jedna konkrétní cesta případem užití
- Jiný pohled na případ užití, vhodné pro složité případy užití
- Neobsahuje větvení!
- **Primární scénář:** normální průchod, kdy vše probíhá normálně, nebo nejpravděpodobnější cesta. Pro daný p.u. je **jediný!**
- **Sekundární scénáře:** jiné cesty popsané méně podrobně nebo jen vyjmenované
 - **alternativní** scénáře: jiné povolené cesty (větvení)
 - **výjimečné** scénáře: pro zpracování chyb
- Otázky pro nalezení sekundárních scénářů:
 - jaké jiné akce mohou být prováděny v tomto bodě?
 - je něco, co by se mohlo pokazit v tomto bodě?
 - nějaké chování, které může nastat kdykoli během primárního scénáře, např. zrušení nebo znovuspuštění?

P.u.: Diagramy aktivit (1)



Diagramy aktivit patří do skupiny dynamických diagramů UML; mohou být také použity pro dokumentaci případů užití. Podrobněji budou probrány později.



začátek případu užití

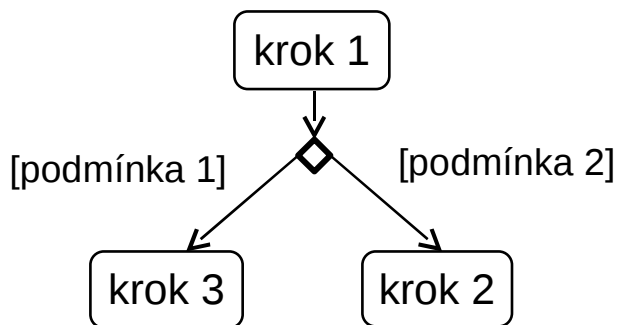
konec případu užití

krok případu užití (reprezentovaný jako stav obsahující akce), nebo **několik kroků** podrobněji popsáných na zvláštním diagramu

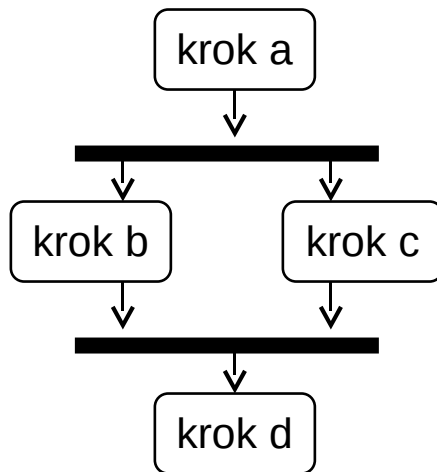
přechod mezi kroky

podmínka přechodu

rozhodnutí s **rozhodovacím bodem**



P.u.: Diagramy aktivit (2)



paralelní kroky (fork & join)

Diagramy aktivit obsahují více modelovacích prvků, které zde neuvádíme, protože nejsou relevantní pro popis případů užití

Nedostatky:

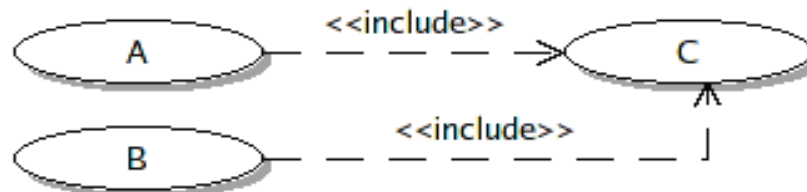
- » prostor pro popis kroků na diagramu je omezený
- » nejsou znázorněny vztahy „užívá“ a „rozšiřuje“ (pokud si je sami nevytvoříte a nepřidáte do notace)
- » diagramy mohou být velmi složité ==> pro každý p.u. je nutné rozhodnout, zda je vhodné použít textový popis, diagram aktivit nebo obojí

Pokročilé modelování případů užití

Vztah <<include>>

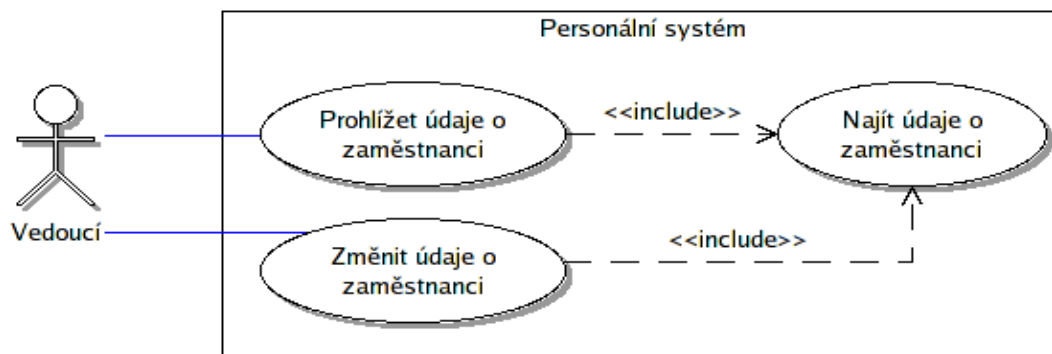


- Další názvy: *užívá*
- Vztah mezi dvěma případy užití
- Společné chování dvou nebo více p.u. může být vyčleněno do zvláštního případu užití (vyhýbá se opakování a kopírování společných kroků)



- A a B nejsou kompletní bez C
- C je vyvoláno alespoň jednou v A a B
- C může být (a často je) použito ve více případech užití
- Pokud je C úplný p.u., může být vyvolán i přímo aktérem
- C je popsáno stejně, jako jiné případy užití

Vztah <<include>> (II)



Případ užití: Změnit údaje o zaměstnanci

Vstup. podmínky: K systému je přihlášený oprávněný Vedoucí.

Tok událostí:

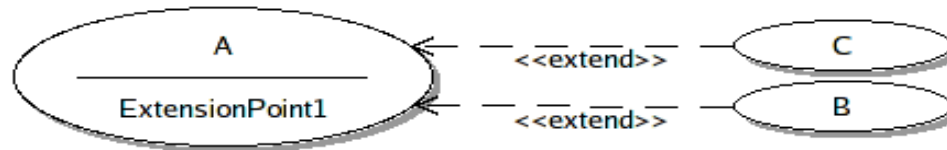
1. PU začíná požadavkem na změnu údajů zaměstnance.
2. INCLUDE(*najít údaje o zaměstnanci*).
3. Systém zobrazí aktuální údaje zaměstnance.
4. ...

Výstup. podmínky:

Vztah <<extend>>



- Další názvy: *rozšiřuje*
- Vztah mezi dvěma případy užití
- Pro vložení **nového chování**, volitelných částí a pro zpracování chyb



- A je úplný p.u., který v ideálním případě neví nic o existenci svých rozšíření B a C
 - => základnímu scénáři je jedno, kdo ho rozšiřuje
- Umístění rozšiřujícího p.u. je označeno *bodem rozšíření* (extension point)
 - => v diagramu a/nebo v popisu rozšiřovaného p.u. “ve vrstvě nad tokem událostí”
 - => rozšiřující p.u. ví, jak se přidat do základního scénáře
- Rozšiřující p.u. B a C jsou často *neúplné* p.u. a mohou mít více fragmentů
- *Podmínka pro rozšíření* je uvedena v popisu jednotlivých rozšiřujících p.u. nebo v podmínce rozšiřujícího vztahu (od UML 2.0)
- Rozšíření se často používají v následných verzích systému

Vztah <<extend>> (II)



Případ užití: RezervujKnihu

Preconditions:

1. Jsou zobrazeny detailní informace o knize

Tok událostí:

1. Čtenář klikne na „rezervovat knihu“.

2. Systém uloží čtenáře do fronty rezervací a zobrazí výsledek.

- Chceme přidat:
 - Odmítnout rezervaci čtenáři, který má nevrácenou knihu
 - Zaslání upomínky těm, kteří již měli rezervovanou knihu vrátit
 - Uložení pokuty těm, kteří navíc již upomínku dostali

Vztah <<extend>> (III)



Body rozšíření



Případ užití: RezervujKnihu

Preconditions:

1. Jsou zobrazeny detailní informace o knize

Tok událostí:

1. Čtenář klikne na „rezervovat knihu“.

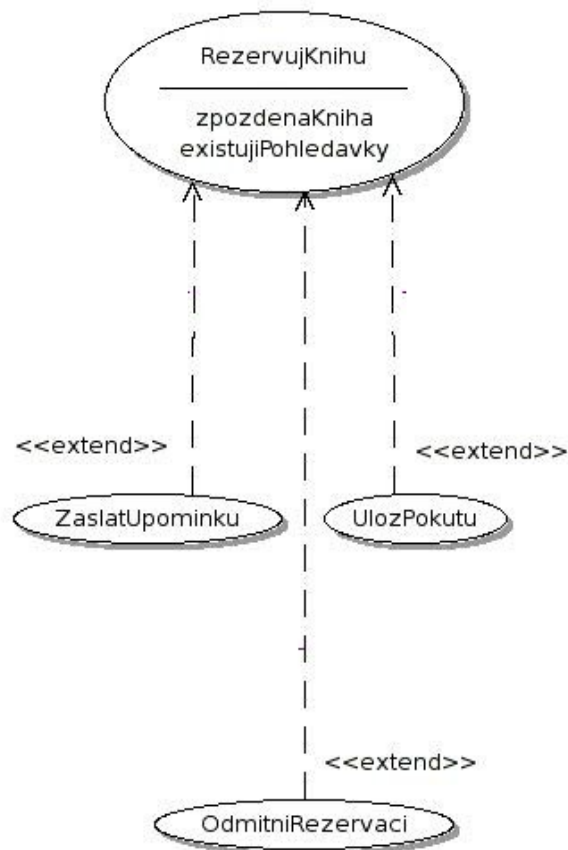
<existujiPohledavky>

2. Systém uloží čtenáře do fronty rezervací a zobrazí výsledek.

<zpozdenaKniha>

- Chceme přidat:
 - Zaslání upomínky těm, kteří již měli rezervovanou knihu vrátit
 - Uložení pokuty těm, kteří navíc již upomínku dostali
 - Odmítnout rezervaci čtenáři, který sám má v nevrácenou knihu

Vztah <<extend>> (IV)



Případ užití: RezervujKnihu

Preconditions:

1. Jsou zobrazeny detailní informace o knize

Tok událostí:

1. Čtenář klikne na „rezervovat knihu“.
<existujiPohledavky>
2. System uloží čtenáře do fronty rezervací a zobrazí výsledek.
<zpozdenaKniha>

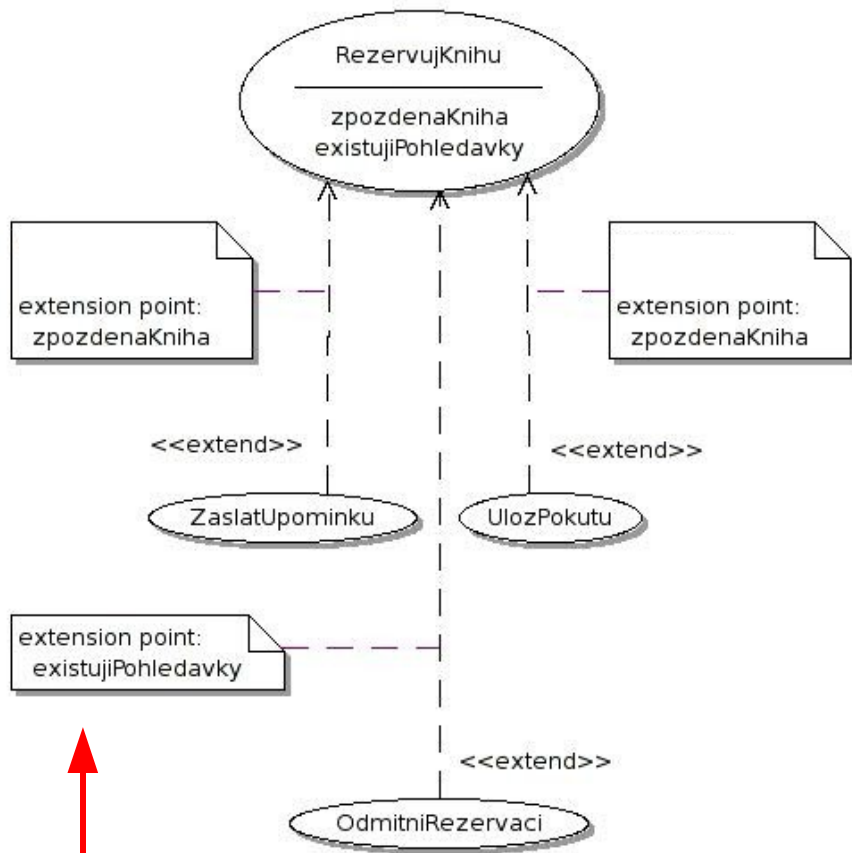
Rozšiřující případ užití: UlozPokutu

1. PRO VŠECHNY čtenáře, kteří mají knihu půjčenou
 - 2.1. IF čtenář měl knihu vrátit AND dostal již upomínku
 - 2.1.1. System zaznamená u čtenáře výši pokuty
 - 2.1.2. System pošle čtenáři info o udělené pokutě

Rozšiřující případ užití: OdmitniRezervaci

1. System vyhledá výpůjčky čtenáře
 2. IF existuje zpožděná výpůjčka
 - 2.1. System zamítne rezervaci

Vztah <<extend>> (V)



Použité body rozšíření

Případ užití: RezervujKnihu

Preconditions:

1. Jsou zobrazeny detailní informace o knize

Tok událostí:

1. Čtenář klikne na „rezervovat knihu“.

<existujiPohledavky>

2. Systém uloží čtenáře do fronty rezervací a zobrazí výsledek.

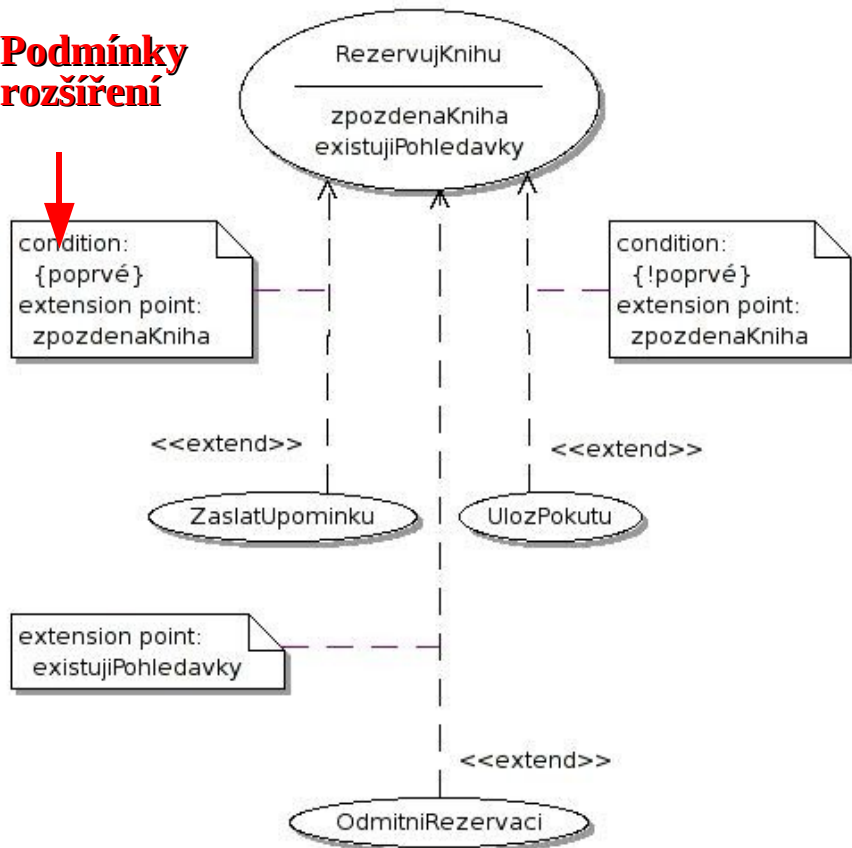
<zpozdenaKniha>

- Pokud vztah <<extend>> nspecifikuje body rozšíření, aplikuje se na všechny
- Rozšiřující p.u. musí mít přesně tolik segmentů, kolik bodů rozšíření používá
- Je dovoleno aby dva p.u. rozšiřovali základní p.u. ve stejném bodě, pořadí je pak ale náhodné

Vztah <<extend>> (VI)



Podmínky rozšíření



Případ užití: RezervujKnihu

Preconditions:

1. Jsou zobrazeny detailní informace o knize

Tok událostí:

1. Čtenář klikne na „rezervovat knihu“.

<existujiPohledavky>

2. Systém uloží čtenáře do fronty rezervací a zobrazí výsledek.

<zpozdenaKniha>

Rozšiřující případ užití: UlozPokutu

1. PRO VŠECHNY čtenáře, kteří mají knihu půjčenou

2.1. IF čtenář měl knihu vrátit AND dostal již upomínku

2.1.1. Systém zaznamená u čtenáře výši pokuty

2.1.2. Systém pošle čtenáři info o udělené pokutě

Rozšiřující případ užití: OdmitniRezervaci

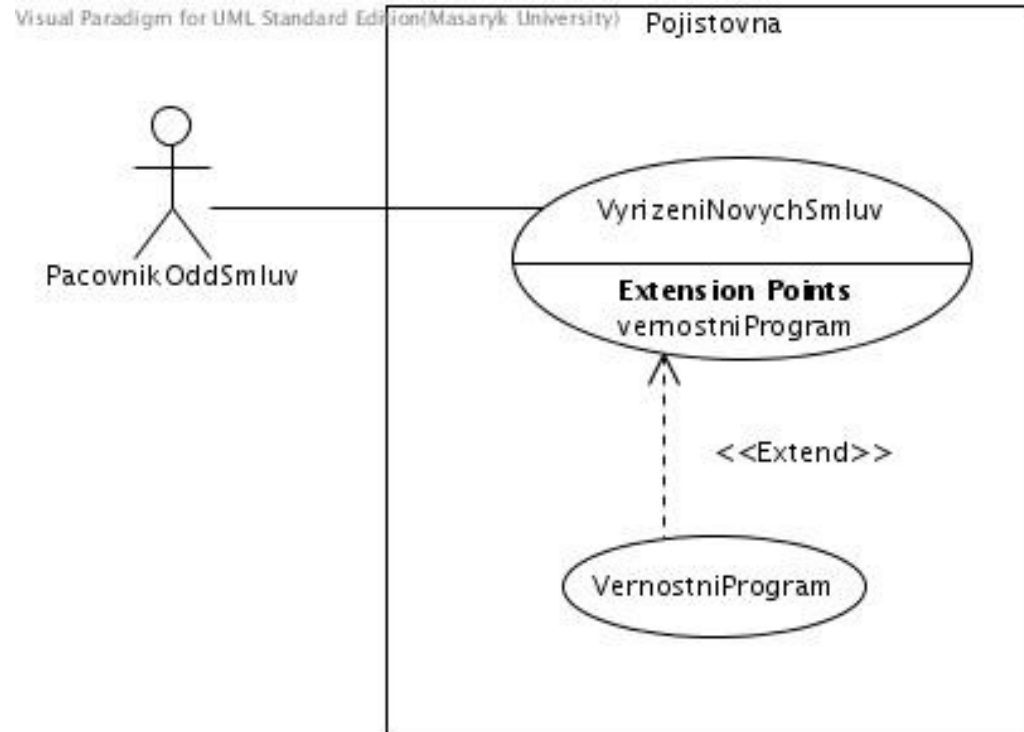
1. Systém vyhledá výpůjčky čtenáře

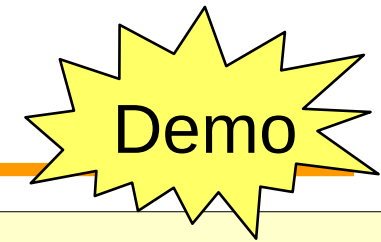
2. IF existuje zpožděná výpůjčka

2.1. Systém zamítne rezervaci

Pojišťovna: Rozšíření p.u. (I)

Demo





Use Case: VyrizeniNovychSmluv

Primary Actor: PracovnikOddSmluv

Preconditions

1. Pracovnik oddeleni smluv je prihlaseny do systemu
2. System pri zadavani zadosti overil, ze nechybi nezbytné údaje, zejména rodne číslo zadatele

Flow of Events

...

3.3.2 IF PracovnikOddSmluv kline na "vytvorit novou smlouvu"

3.3.2.1 System zobrazí formulář pro založení nové smlouvy s předdefinovanými informacemi z žádosti

<vernostniProgram>

3.3.2.2 PracovnikOddSmluv vyplní požadované informace a potvrdí vytvoření smlouvy

3.3.2.3 System smlouvu uloží do databáze

...

Use Case: VernostniProgram

Extension point: vernostniProgram

Flow of Events

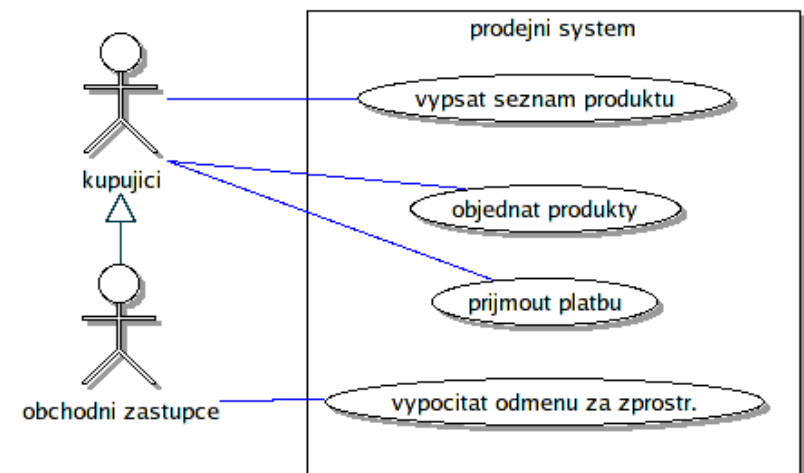
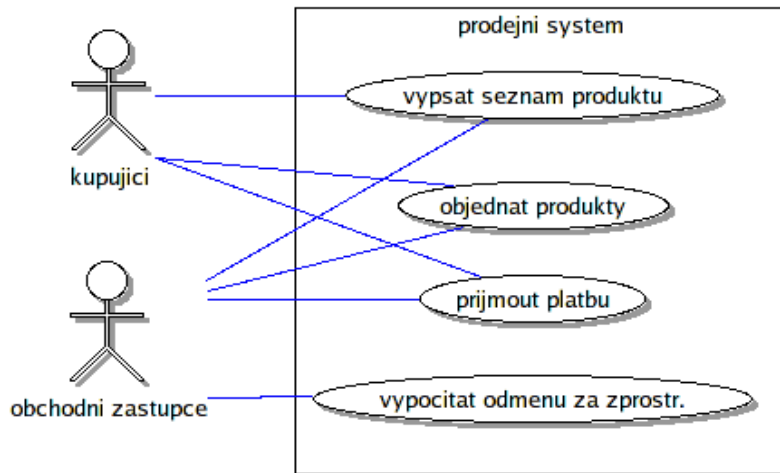
1 System zjistí průměrnou výši pojistného z dřívějších smluv a platební morálku pojistnice a vypočítá slevu na pojistném

2 System upraví cenu pojištění ve formuláři podle vypočítané slevy

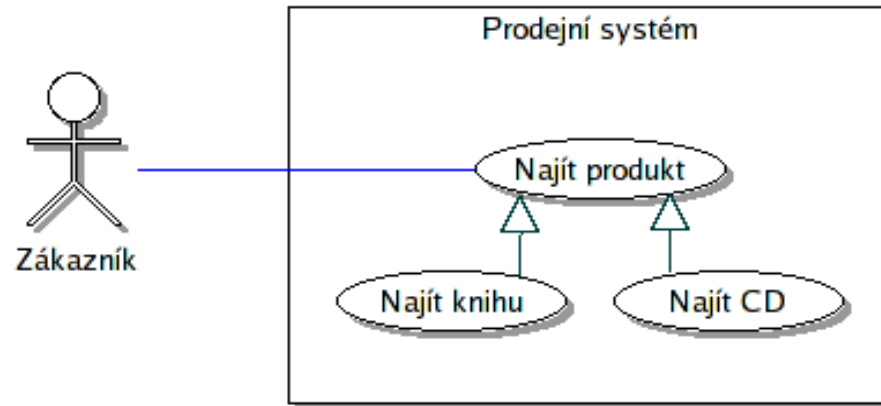
Dědičnost aktérů



- Aktér *obchodní zástupce* je podtypem aktéra *kupující*
- Specializovaný aktér plní stejnou roli jako obecnější aktér, a pravděpodobně další roli navíc
- Specializovaný aktér se účastní ve všech případech užití, ve který se účastní obecnější (dědí spouštění případů užití „nadaktéra“)



Dědičnost případů užití



Dědičnost případů užití (II)



- Specializovaný p.u. může
 - dědit rysy z rodiče
 - přidávat nové rysy
 - přepisovat (měnit) zděděné rysy

| Rys | Dědění | Přidání | Změna |
|----------------------|--------|---------|-------|
| Vztah | A | A | N |
| Bod rozšíření | A | A | N |
| Vstupní podmínka | A | A | A |
| Výstupní podmínka | A | A | A |
| Krok v toku událostí | A | A | A |

- V dokumentaci specializovaného p.u. se doporučuje následující notace, kdy v závorce je číslo, které odpovídá rysu z rodiče

| Rys je ... | Příklad |
|---------------------------------|--|
| Zděděný beze změny | 3. (3.) Zákazník vloží požadovaná kritéria. |
| Zděděný a přecíslovaný | 6.2 (6.1) Systém sdělí zákazníkovi, že odpovídající produkt nebyl nalezen. |
| Zděděný a změněný | 1. (o1.) Zákazník vybere „vyhledej knihu“. |
| Zděděný, změněný a přecíslovaný | 5.2 (o5.1) Systém zobrazí stránku s detaily maximálně pěti knih. |
| Přidaný | 6.3 Systém znovu zobrazí vyhledávací stránku. |

Podrobnější model rozhraní



Může být užitečné podrobněji dokumentovat rozhraní systému (vymezené pomocí případů užití) např.:

- **Obrazový scénář** (pro grafická uživatelská rozhraní)
 - pouze náčrtky ukazující různé obrazovky
 - diagram ukazující posloupnost obrazovek
 - prototyp uživatelského rozhraní (prověření motivace !!!)
- **Definice protokolu** (pro systémová rozhraní)
 - např. při výměně XML zpráv definujte tyto zprávy
- **Interakční diagramy** na různých úrovních abstrakce
 - mezi aktéry a systémem, asynchronní události
 - popis vlevo
 - strukturovaná angličtina nebo závorky pro různé cesty
 - různé úrovně abstrakce !



- **vymezení problému** (iterativní proces)
- odhalení **hranic systému**
- diskuse **požadavků s koncovými uživateli**
- specifikace **vnějšího chování** systému, funkčních požadavků pro **vývojáře systému**
- výchozí bod pro **návrh** (diagramy tříd, interakční diagramy)
- **rozdělení podrobnější analýzy** požadavků podle případů užití
- **rozdělení konstrukčních iterací** podle případů užití
- odvození testovacích případů, odvození **uživatelské dokumentace**
- znovupoužití, architektura (viz literatura)
- velikost projektu a **odhad ceny** (viz kniha G. Schneider)
-



- Odvodte případy užití z firemních procesů nebo procházením seznamu aktérů
- Jako první identifikujte primární scénáře
 - zamyslete se nad vztahy <<include>>
 - odsouhlaste s uživateli
 - **system by měl být optimalizován pro primární scénáře**
- Soupis alternativ provádějte až po akceptaci základních p.u.
 - zamyslete se nad vztahy <<extend>>
 - alternativy nepopisujte detailně
- Diagramy se průběžně zpřesňují

Rady pro modelování p.u. (II)



- Vytvářejte krátké a jednoduché p.u.
 - jen nezbytné detaily
 - z praxe: hlavní tok událostí by se měl vejít na jednu stránku
- Zaměřte se na **co**, ne na **jak**
 - modelujeme **co** aktéři dělají, ne **jak** to dělají

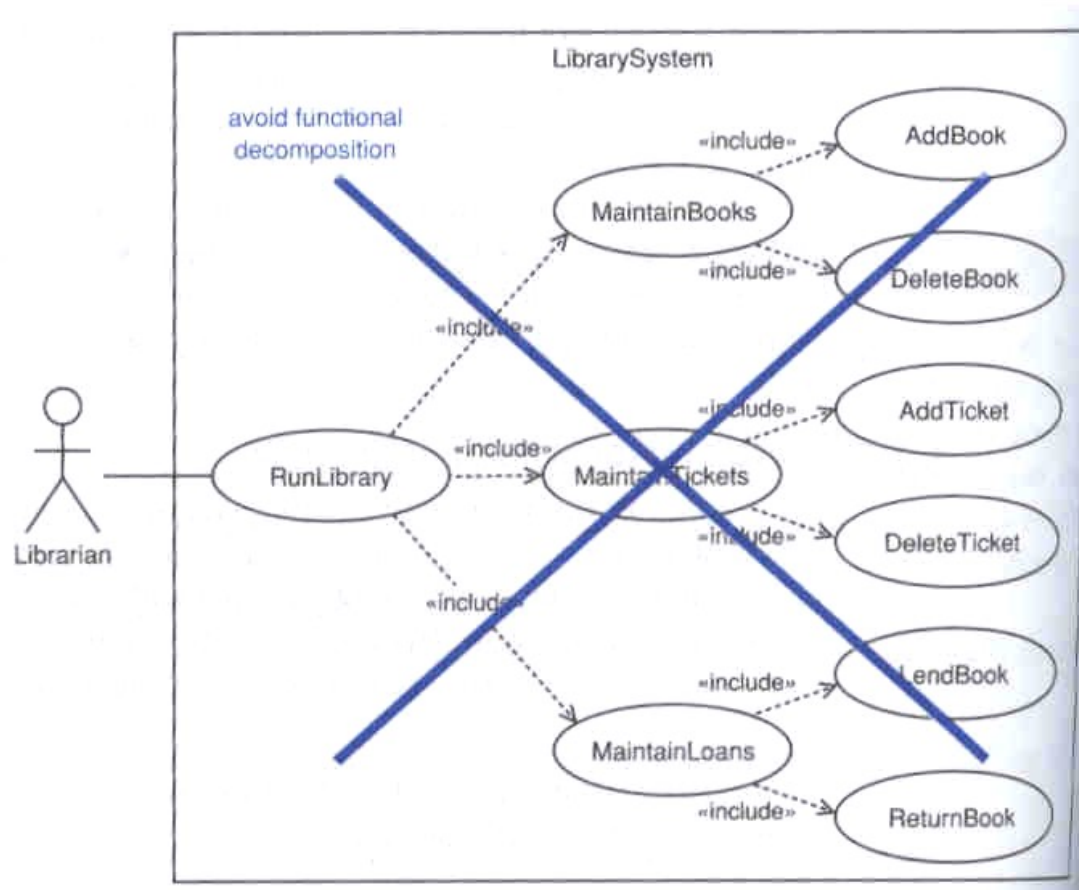
...
4. Systém požádá Zákazníka o potvrzení objednávky.
5. Zákazník klikne na tlačítko OK.
...

...
4. Zákazník potvrdí objednávku.
...

Rady pro modelování p.u. (III)



- Vyhněte se funkční dekompozici





- mnoho knih, zaměřených podle předmětných oblastí (obchod, telekomunikace, aj.)
- dobré všeobecné knihy o případech užití:
 - **Applying Use Cases**, G. Schneider: krátká, dobrý úvod, průvodce tvorbou a dokumentací případů užití, jak definovat architekturu založenou na případech užití a cenový odhad na jejich základě;
 - **Software Reuse**, Jacobson, Griss, Jonsson: tvorba složitých a znovupoužitelných systémů a architektur, aplikace případů užití při řešení takových systémů;

UseCases se používají (nejen) pro stanovení požadavků u velkých projektů:

<http://www.w3.org/TR/mmi-use-cases/>