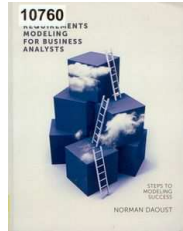


# Literatura

- N. Daoust. **UML Requirements Modeling For Business Analysts.**

Základní přehled UML diagramů  
a způsobu jejich použití.



- N. Wiegers, J. Beatty. **Software Requirements.**

Proces získávání, specifikace  
a validace požadavků.



## Úvod do softwarového inženýrství

IUS 2019/2020

### 2. přednáška

Ing. Radek Kočí, Ph.D.  
Ing. Bohuslav Křena, Ph.D.

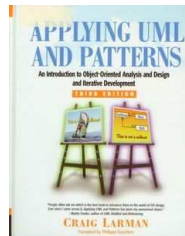
30. září a 4. října 2019

3 / 59

# Literatura

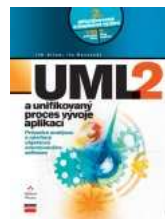
- C. Larman. **Applying UML and Patterns.**

Proces vývoje softwaru,  
základní modely, metodiky, UML diagramy,  
principy OO návrhu, návrhové vzory.



- J. Arlow. **UML 2 a unifikovaný proces vývoje.**

UML diagramy v procesu vývoje,  
zaměřeno na UP.



## Téma dnešní přednášky

- Literatura
- Analýza a specifikace požadavků
- Modely jazyka UML používané při specifikaci požadavků
  - Diagram případů užití (*Use Case Diagram – UCD*)
  - Diagram aktivit (*Activity Diagram*)
  - Stavový diagram (*State Diagram*)

# Zaínteresované strany

Pojem *stakeholder*

- (původně) dočasný držitel peněz či majetku
- člověk nebo skupiny lidí, bez jejichž podpory by organizace přestala existovat
- (obecně chápán jako) zaínteresované strany v projektu – zákazník, uživatel, analytik, návrhář, tester, manažer, ...

Je důležité

- zapojit nejen zákazníka, ale všechny zaínteresované strany (stakeholders)
- tyto strany na začátku *identifikovat* – pokud analytik v průběhu tvorby požadavků zjistí, že existuje ještě někdo, kdo by se měl k něčemu vyjádřit, zdržuje to průběh projektu; pokud se to nezjistí, může být v požadavcích chyba

## Analýza a specifikace požadavků – příklad

*Gerhard (senior manager): "Potřebujeme systém pro sledování chemikálií. Systém by měl sledovat pohyb všech kontejnerů s chemikáliemi, které jsou ve skladě nebo v laboratořích. Tímto by měli chemici přehled o stavu chemikálií a nemuseli by kupovat nové, když jsou k dispozici ve skladu nebo v jiné laboratoři. Dále, ministerstvo chce zprávy o používání chemikálií a jejich dostupnosti, což dnes zabírá mnoho času. Systém by měl umět generovat podklady pro tyto zprávy."*

*Cynthia: "Popsal jste general business objectives. To mi nedává dostatek informací, abych věděla, jaký software vytvořit a jak dlouho to bude trvat."*

⇒ **Jaké mohou být požadavky?**

# Činnosti spojené s vývojem softwaru

- **analýza a specifikace požadavků (8 %)**,
  - **Cíl: Stanovení služeb, které zákazník požaduje od systému, a vymezení podmínek jeho vývoje a provozu.**
  - transformace neformálních požadavků uživatele do strukturovaného popisu požadavků,
  - zdůraznění požadavků uživatele, ne jak toho docílit (realizovat),
  - provedení studie vhodnosti, identifikace a analýza rizik,
  - získávání, analýza, definování a specifikace požadavků,
  - plánování akceptačního testování.
- architektonický a podrobný návrh (7 %),
- implementace (12 %),
- integrace a testování (6 %),
- provoz a údržba (67 %).

## Analýza a specifikace požadavků – příklad

**Příklad komunikace mezi zákazníkem a dodavatelem.**

*K. Wiegers, J. Beatty: Software Requirements. Microsoft Press, 2013.*

*Gerhard (senior manager): "Potřebujeme systém pro sledování chemikálií. Systém by měl sledovat pohyb všech kontejnerů s chemikáliemi, které jsou ve skladě nebo v laboratořích. Tímto by měli chemici přehled o stavu chemikálií a nemuseli by kupovat nové, když jsou k dispozici ve skladu nebo v jiné laboratoři. Dále, ministerstvo chce zprávy o používání chemikálií a jejich dostupnosti, což dnes zabírá mnoho času. Systém by měl umět generovat podklady pro tyto zprávy."*

*Cynthia (IT specialist): "Dobře, vidím, proč je to důležité. Nyní potřebuji pochopit požadavky na systém pro sledování chemikálií."*

*Gerhard: "?? Právě jsem je řekl."*

⇒ **Rozdílné pohledy na to, jaké informace jsou důležité.**

⇒ **Kdo vše se účastní projektu?**

# Nefunkční požadavky

## Požadavky na provoz systému

- statické – např. počet uživatelů, ...
- dynamické – např. čas odezvy, počet transakcí na jednotku času, ...

## Požadavky na výsledný systém

- počítačové vybavení – např. HW náročnost (paměť, ...)
- programové vybavení – např. operační systém, programovací jazyky, ...
- vyvíjený software – např. efektivnost, spolehlivost, odolnost vůči chybám, přenositelnost, bezpečnost, ...

# Nefunkční požadavky

## Požadavky na vývojový proces

- dodržování norem
- odevzdání systému

## Požadavky na rozhraní

- software → uživatel
- software → jiné součásti systému (HW, SW)

## Externí požadavky

- legislativní požadavky (ochrana informací, ...)

měřitelnost požadavků

# Typy požadavků

- **Obchodní požadavky** (*Business Requirements*)
  - proč zákazník potřebuje systém  
⇒ pochopení a definování cílů a smyslu projektu
  - zaměřeno na obchodní cíle (úspora nákladů, času)
  - *úspora nákladů při práci s chemikáliemi*
- **Uživatelské požadavky** (*User Requirements*)
  - úlohy, které uživatel se systémem provádí  
⇒ co je možné se systémem dělat
  - *zjistit dostupnost chemikálie na skladě nebo v laboratoři*
  - use cases, ...

11 / 59

9 / 59

# Typy požadavků

- **Funkční požadavky** (*Functional Requirements*)
  - chování systému v různých podmínkách  
⇒ co musí být realizováno, aby mohly být vykonány úlohy (user requirements), a tím splněny obchodní požadavky (business requirements)
  - *co vše je potřeba pro zjištění dostupnosti chemikálie*
  - diagram aktivit, ...
- **Nefunkční požadavky** (*Nonfunctional Requirements*)
  - Vlastnosti a charakteristiky, které musí systém splňovat, a omezení, která musí respektovat.

12 / 59

10 / 59

# Metody získávání informací

## Pozorování prací u zákazníka

- prostá specifikace nemusí být úplná, některé detaily nemusí být zachyceny, neboť jsou "zjevné" (nemusí pro každého)  
*Recept: přidejte dvě nebo tři vejce (Za jakých podmínek dvě vejce? Co skořápka?)*
- umožňuje lépe pochopit aktivity a procesy, ověřit získané informace, odhalit dosud neznámé informace
- časově náročné, není vhodné pro všechny projekty a uživatele

## Další metody

- studium dokumentů
- přímá účast na pracích zákazníka
- analýza existujícího softwarového systému

# Metody získávání informací

Kvalitní získávání informací o problémové oblasti a požadavcích snižuje riziko vytvoření systému, který nebude vyhovovat **potřebám** uživatele.

Důležitá je motivace ze strany zákazníka (uživatele).

Pro analytika jsou nutné komunikační schopnosti i zkušenosti.

15 / 59

16 / 59

# Analýza a specifikace požadavků – příklad

*Cynthia: "Popsal jste general business objectives. To mi nedává dostatek informací, abych věděla, jaký software vytvořit a jak dlouho to bude trvat."*

*Cynthia: "Jeden analytik by se měl účastnit práce s některými uživateli, abychom přesně pochopili všechny požadavky."*

⇒ **Jaké jsou techniky získávání informací?**

# Metody získávání informací

## Interview (orientační, strukturované)

- základní běžná forma zjišťování potřeb zákazníka
- orientační – první setkání, získat základní přehled
- strukturované – připravené otázky, získat hlubší představu
- nejen naslouchat, ale navrhnout alternativy

## Dotazníky

- lze obsáhnout velkou skupinu lidí
- dobře připravené otázky s předem definovaným způsobem vyhodnocení

## Pracovní setkání (workshop, elicitation meeting)

- skupina lidí (stakeholders) vyjednává o požadavcích a pracuje společně na specifikaci požadavků
- menší skupiny jsou efektivnější, různá setkání s různou skupinou lidí

13 / 59

14 / 59

# Problémy při specifikaci požadavků

Problémy plynou z použití **přirozeného jazyka**.

- **Vyřazení** – *Používají systém k výpůjčkám knih.*  
Kdo?
- **Deformace, zkreslení** – *Čtenáři si nemohou půjčit další knihu, dokud nevrátí knihy s prošlou výpůjční lhůtou.*  
Když je zaplatí, tak mohou!
- **Zobecnění** – *Každý, kdo si chce vypůjčit knihu, musí mít průkazku.*  
A co výpůjčky mezi knihovnami?

## Slovníček pojmů

- zachycuje obchodní jazyk (terminologii) pro daný projekt,
- řeší synonyma (slova téhož nebo podobného významu, např. dopis – psaní, směle – statečně, holka – dívka)  
výběrem nejčastějšího,
- řeší homonyma (slova s odlišným významem znějící stejně, např. pila, zámek, diskrétní) definicí jejich významu.

19 / 59

## Analýza a specifikace požadavků – příklad

*Cynthia: "Jeden analytik by se měl účastnit práce s některými uživateli, abychom přesně pochopili všechny požadavky."*

*Gerhard: "Jsou to vytížení lidé. Nemají čas vysvětlovat někomu každý detail. Nemůžete sami určit, co se má vytvořit?"*

*Cynthia: "Můžeme vytvořit pouze náš nejlepší odhad, ale my nejsme chemici. Podle mých zkušeností, pokud nebudeme mít čas na pochopení problému, nikdo nebude spokojený s výsledkem."*

*Gerhard: "Na to nemáme čas. Moje požadavky jsem vám dal. Prostě ten systém vytvořte."*

⇒ **Jak je to se zapojením uživatelů do tvorby požadavků?**

# Problémy při specifikaci požadavků

## Přirozená neúplnost a nepřesnost

- nejasná a neúplná formulace požadavků zákazníkem
- neucelená představa uživatele o výsledném softwarovém systému
- problém rozhodování, jaké požadavky už nezačleňovat do specifikace
- pro komunikaci se používá přirozený jazyk, který je nejednoznačný

## Nedostatek znalostí

- vývojář (analytik) se neorientuje v doménové problematice analyzované oblasti, nezná terminologii
  - specialista na doménovou oblast ve vývojovém týmu
- zákazník se neorientuje v problematice vývoje softwaru, nezná terminologii
  - vyčleněný člověk od zákazníka (orientuje se ve vývoji, zaškolení, ...)

17 / 59

## Problémy při specifikaci požadavků

### Nekonzistence požadavků

- různí uživatelé mají různé požadavky a priority
- různé požadavky uživatele a zákazníka (objednavatele)
- požadavky jsou mnohdy rozporné

### Další problémy

- špatná predikovatelnost dopadu nového systému na organizaci, kde se nasadí
  - otázka naplnění *obchodních požadavků*
- problémy s testováním a validací požadavků
  - zapojení zákazníka
  - prototypování, pravidelná setkání, ...

## Postup při specifikaci požadavků

## Analýza požadavků

- *studie vhodnosti* = odhad, zda je reálné vytvořit systém s danými vlastnostmi za daných podmínek; musí být provedena rychle a levně
- zkoumání současného stavu
- modelování, prototypování, . . .

## Specifikace požadavků

- transformace informací z analýzy do dokumentu
- specifikace nefunkčních požadavků, ...

## Validace požadavků

- vyhodnocení požadavků, simulování, prototypování, ...
- definování kritérií pro akceptování produktu

# Prototypování

## Motivace

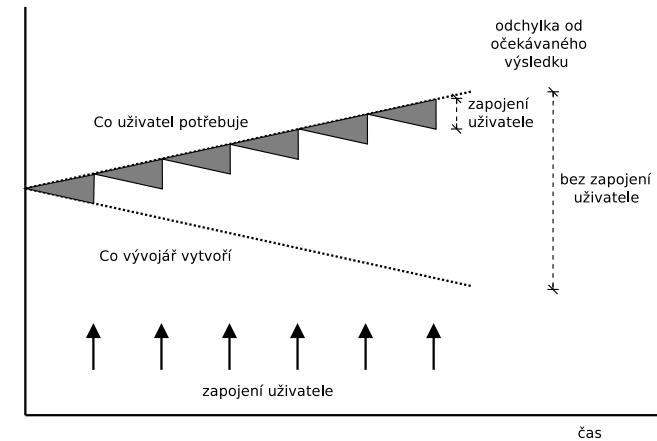
- uživatelé mají problémy s přesnou specifikací svých požadavků, avšak dokáží lépe formulovat požadavky v reakci na práci s "reálným" produktem ⇒ prototypem
- prototypování dokáže ověřit správnost návrhu v počátečních fázích ⇒ **validace požadavků** (snižují se rizika projektu)
- může nastínit různé alternativy návrhu

## Prototyp

- částečná implementace produktu
- prezentuje vnější rozhraní systému
- prototyp je většinou implementován rychle s cílem demonstrovat potenciální rozhraní či chování systému, není kladen důraz na kvalitu návrhu a programování

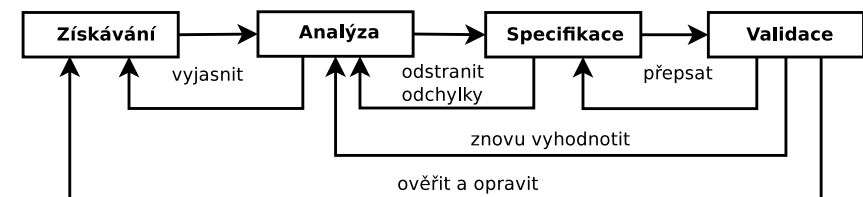
## Zapojení uživatelů

**Zapojení uživatele/zákazníka do procesu tvorby je důležité.**



## Postup při specifikaci požadavků

Tvorba požadavků je iterativní proces.



## Získávání informací

- definice cílů projektu
- identifikace uživatelských požadavků, ...
- interview, pozorování práce, ...

# Dobrá specifikace požadavků

Specifikace by měla být

- **úplná**
  - obsahuje všechny důležité požadavky a definice reakcí systému na všechny třídy vstupních údajů
  - specifikujte situace, ve kterých se porušuje akceptovatelné chování
- **konzistentní**
  - požadavek není v rozporu s jinými požadavky
  - buďte konzistentní v používání názvů
- **verifikovatelná**
  - existuje proces kontroly, zda software splňuje požadavek
  - měřitelnost splnění požadavků

27 / 59

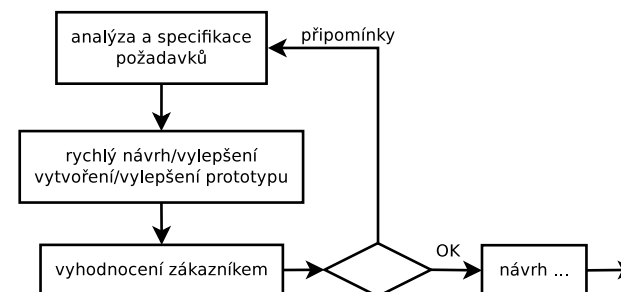
# Dobrá specifikace požadavků

Během procesu specifikace požadavků

- **Udržujte specifikaci čitelnou pro zákazníka.**
  - zákazník se musí umět orientovat ve specifikaci
- **Ve specifikaci nenavrhuje řešení.**
  - cílem specifikace je získat úplný a správný pohled na potřeby zákazníka
  - realizace požadavků je záležitostí dalších etap
- **Validujte požadavky.**
  - prototyp snižuje riziko špatného pochopení požadavků
  - slabá specifikace ⇒ špatný odhad nákladů
- **Zainteresuujte uživatele**
  - uživatel se musí podílet na procesu formování a validace požadavků
  - nechte si výsledek zkontrolovat a potvrdit druhou stranou

28 / 59

# Prototypování



## Problém dokončení prototypu

- není kladen důraz na kvalitu návrhu a implementace prototypu ⇒ prototyp je špatně udržovatelný
- po vyjasnění specifikace a návrhu se prototyp dále nepoužívá
- *pokud se prototypy od začátku realizují plnohodnotně, lze je využít v implementaci (např. metodika RAD)*

25 / 59

# Dobrá specifikace požadavků

Specifikace by měla být

- **seřazená podle důležitosti**
  - poznačte si datum vytvoření požadavku
  - seskupte požadavky do tříd důležitosti
- **sledovatelná**
  - původ (smysl) požadavku je jasný
  - na každý požadavek je možné se odkazovat v další dokumentaci
- **modifikovatelná**
  - struktura a styl specifikace je konzistentní a bez redundancí
  - snadné úpravy a doplňování požadavků
- **jednoznačná**
  - neumožňuje více interpretací
  - požadavky pište jasně a jednoznačně (přirozený jazyk je zdrojem nejednoznačnosti)

26 / 59

# Specifikace požadavků – dokumentace

## Dokumentace spojená se specifikací požadavků

- různorodá, od textu až po formální specifikace
- čím formálnější podoba, tím méně lidí je schopno tuto podobu akceptovat
- praxe ukazuje, že pro mnoho softwarových projektů se používá kombinace strukturovaného jazyka, vizuálních modelů a dalších prezentačních technik (tabulky, matematické výrazy, ...)
- přiměřený rozsah dokumentace!

## Formy dokumentace požadavků

- strukturovaný text
- tabulky
- vizuální modely
- spustitelné modely
- formální modely
- ...

# Cena chyb ve specifikaci

Přibližný odhad nákladů na opravu chyb ve specifikaci

Etap	Náklady (člověko-hodiny)
Specifikace	2
Návrh	5
Implementace	15
Akceptační testování	50
Údržba	150

Dobře identifikované požadavky snižují cenu vývoje softwaru!

# Specifikace požadavků – modelování

## Modelování dat

- *Entity Relationship Diagram (ERD)* – strukturovaný model dat
- *Class Diagram* – objektově orientovaný model dat a protokolu (rozhraní, zodpovědnosti)

## Modelování funkčních požadavků

- *Data Flow Diagram (DFD)* – specifikace chování systému; strukturovaný přístup
- *Use Case Diagram (UCD)* – diagram případů užití, specifikuje možnosti použití systému; jazyk UML
- UCD je doplněn dalšími modely z jazyka UML, např. *Activity Diagram*

Podívejte se na učební text zaměřený na jazyk UML 2.0, který máte k dispozici na  
privátních stránkách předmětu IUS!

# Myšlenka pro analytika ...

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, zjistí, že je to přesně to, co vlastně celý čas chtěl.



# Detail (Specifikace) případu užití

- Konkretizace (specifikace) případu užití.
- Neexistuje standard, většinou se však využívá tabulka.
- Příklad užití má svůj
  - název,
  - jedinečný identifikátor a
  - specifikaci.
- Specifikace případu užití má:
  - vstupní podmínky,
  - tok událostí a
  - následné podmínky.

## Specifikace případu užití

název	<b>Případ užití: Platit daň z přidané hodnoty</b>
identifikátor	<b>ID: UC1</b>
účastníci	<b>Účastníci:</b> Čas finanční úřad
stav před	<b>Vstupní podmínky:</b> 1. Je konec fiskálního čtvrtletí?
kroky	<b>Tok událostí:</b> 1. Příklad užití začíná na konci fiskálního čtvrtletí. 2. Systém určuje výši daně z přidané hodnoty, kterou je třeba odvést státu. 3. Systém odesílá elektronickou platbu finančnímu úřadu.
stav po	<b>Následné podmínky:</b> 1. Finanční úřad přijímá daň z přidané hodnoty.

# Diagram případů užití

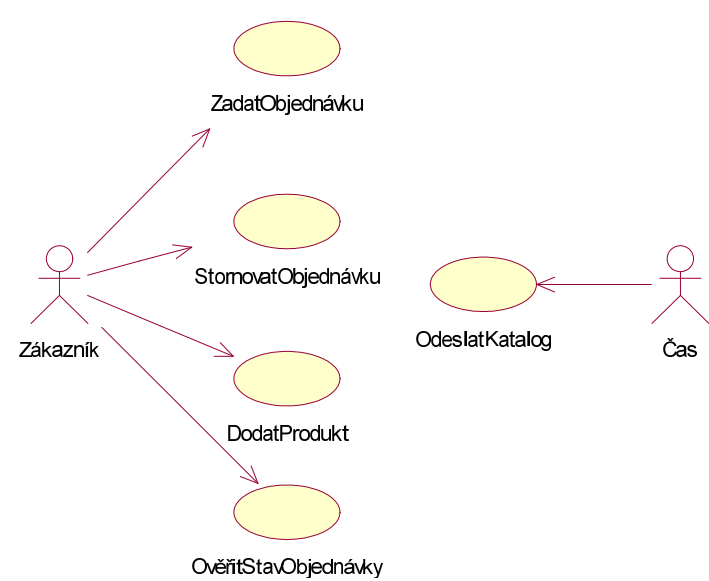
Use-case-driven approach (např. metodika RUP)

- Klíčovými aktivitami při specifikaci požadavků je *nalezení účastníků* a *nalezení případů užití*.
- K tomu se využívají diagramy případů užití doplněné o *detaily (specifikace) případů užití*.

## Prvky diagramu užití

- *hranice systému*
- *účastník (aktor)* – subjekt, který se systémem pracuje (může mít i speciální podobu, např. čas nebo jiný systém)
- *případ užití* – funkce, kterou systém vykonává jménem jednotlivých účastníků nebo v jejich prospěch.
- *interakce* – ukazuje účast aktora na provádění případu užití

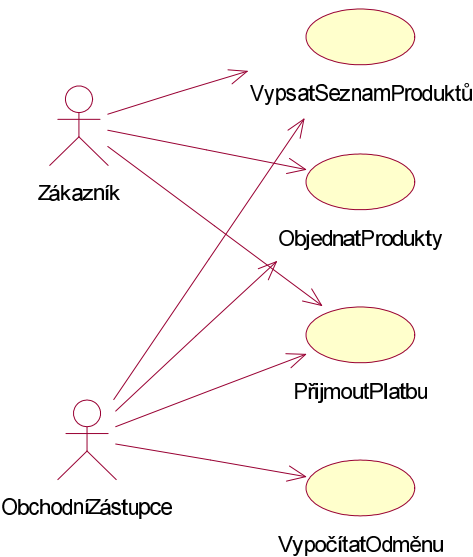
## Diagram případů užití



# Pokročilé techniky modelování případů užití

- Pokročilé techniky:
  - zobecnění účastníka
  - zobecnění případu užití
  - relace «include»
  - relace «extend»
- Pokročilé techniky používejte pouze pro zjednodušení modelu.
- Příklad užití je způsob zápisu požadavků. Musí být tedy čitelný i pro uživatele. Je proto žádoucí, aby byl co nejjednodušší.
- Uživatelé těžko chápou zobecnění účastníka.
- Význam relace «extend» často nechápou ani analytici či návrháři.
- Závěr: Pokročilé techniky používejte co nejméně!

## Diagram bez zobecnění účastníka



# Specifikace případu užití – alternativní toky

Případ užití: Zobrazit košík
ID: UC11
Účastníci: Zákazník
Vstupní podmínky: 1. Zákazník je přihlášen do systému.
Tok událostí: 1. Příklad užití začíná volbou "zobrazit obsah košíku". 2. KDYŽ je košík prázdný: 2.1 Systém oznámí Zákazníkovi, že košík neobsahuje žádné položky. 2.2 Příklad užití končí. 3. Systém zobrazí seznam všech položek v nákupním košíku zákazníka včetně ID, názvu, množství a ceny každé položky.
Následné podmínky:
...

## Specifikace případu užití – alternativní toky

...
Alternativní tok 1: 1. Zákazník může kdykoliv opustit obrazovku košíku.
Následné podmínky:
Alternativní tok 2: 1. Zákazník může kdykoliv opustit systém.
Následné podmínky:

Relace <<include>>

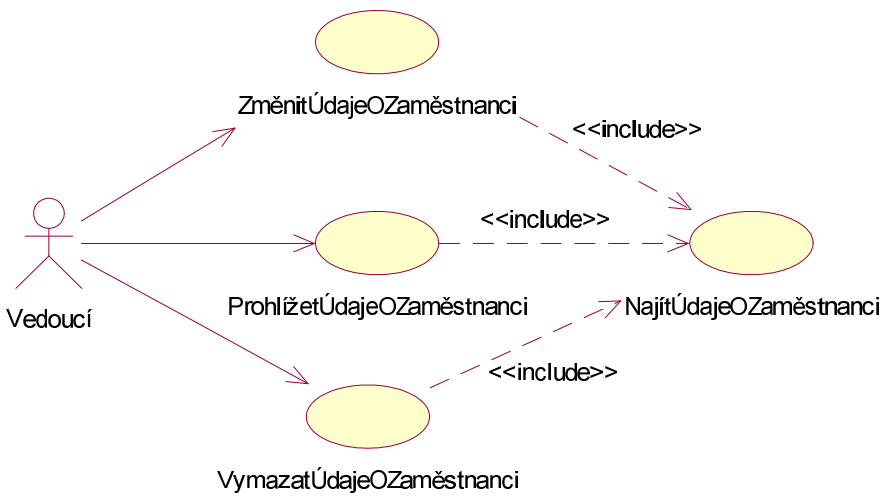
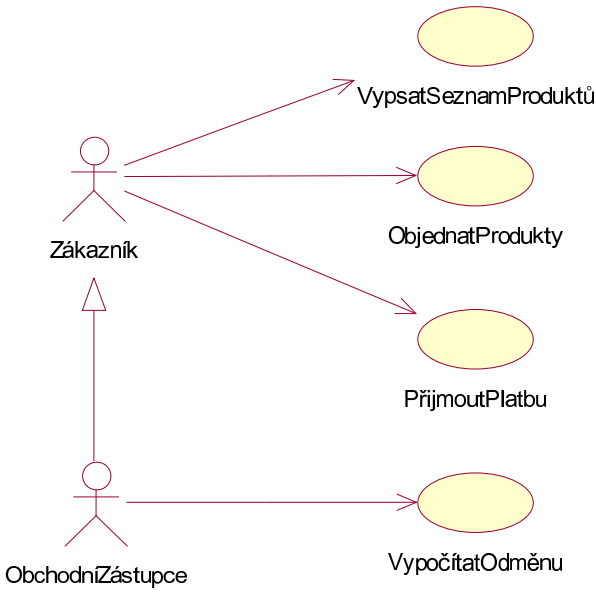
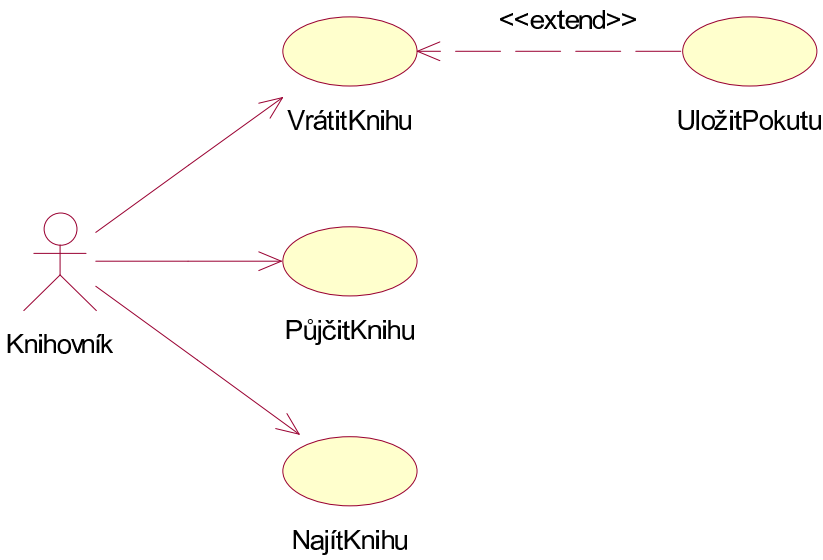


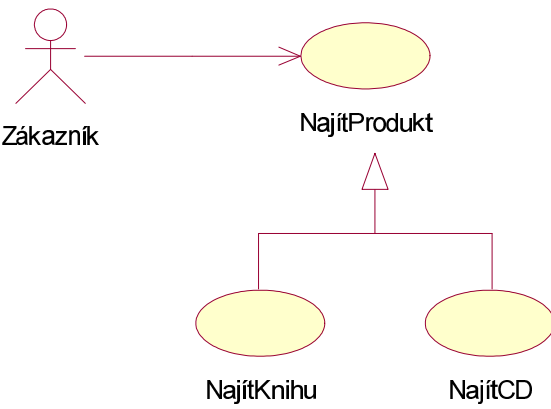
Diagram se zobecněním účastníka



Relace <<extend>>



Zobecnění případu užití



# Diagram aktivit – Prvky

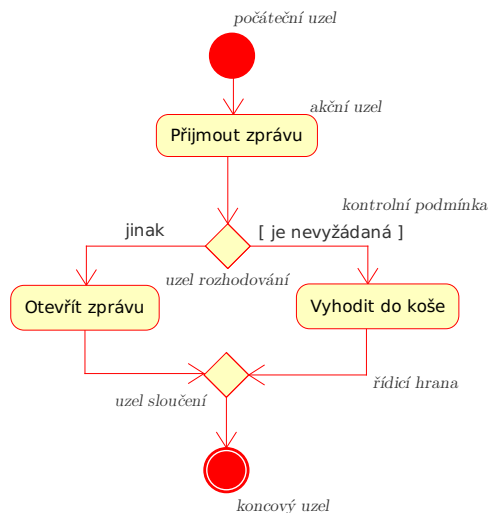
## Prvky diagramu

- uzly
  - akční uzly: modelují aktivitu
  - řídicí uzly: modelují rozhodování; počáteční uzel; koncový uzel; ...
  - objektové uzly: modelují objekty podílející se na aktivitách
- hrany
  - řídicí hrany: modelují přechody mezi uzly
  - objektové hrany: modelují cesty objektů mezi uzly

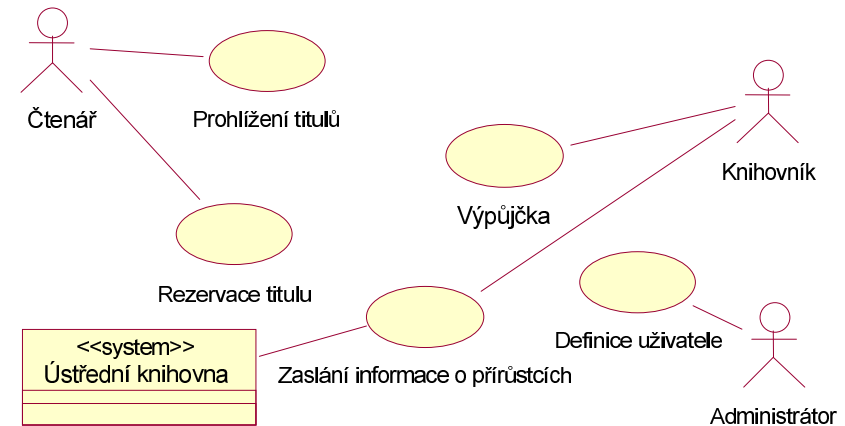
## Možnosti modelování

- tok událostí (včetně toku dat)
- rozhodování
- větvení a spojení
- iterace
- paralelní toky

# Diagram aktivit – Příklad



# Diagram případů užití – alternativní notace



# Diagram aktivit

## Diagramy aktivit (Activity Diagrams)

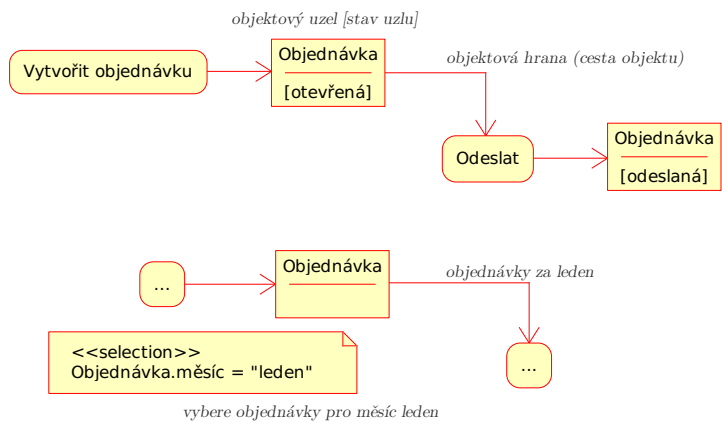
- reprezentují objektově orientované vývojové diagramy
- modelování obchodních procesů, účastní se více objektů
- lze je připojit k libovolnému modelovanému elementu
  - případ užití
  - třída
  - komponenty
  - ...

## Využití diagramu aktivit

- modelování scénářů případů užití
- modelování detailů operace nebo algoritmů
- modelování obchodního procesu

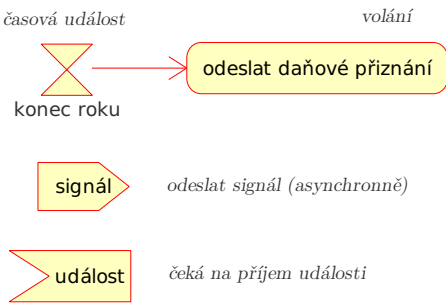
# Diagram aktivít

- objektové uzly a hrany
- stav objektu, výběr objektů



# Diagram aktivít

## Akční uzly



# Stavový diagram

## Stavové diagramy

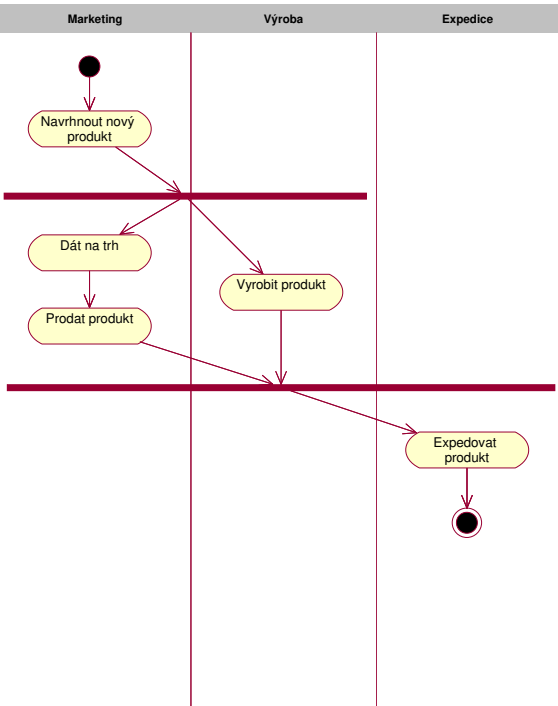
- modelování životního cyklu jednoho reaktivního objektu
- vycházejí ze stavového automatu (Harel)
- mohou modelovat dynamické chování těchto reaktivních objektů
  - třídy, resp. instancí tříd (nejčastější)
  - případy užití
  - podsystémy
  - systémy

## Reaktivní objekt

- reaguje na vnější události
- životní cyklus je modelován jako řada stavů, přechodů a událostí
- chování je důsledkem předchozího chování (následný stav závisí na aktuálním stavu)

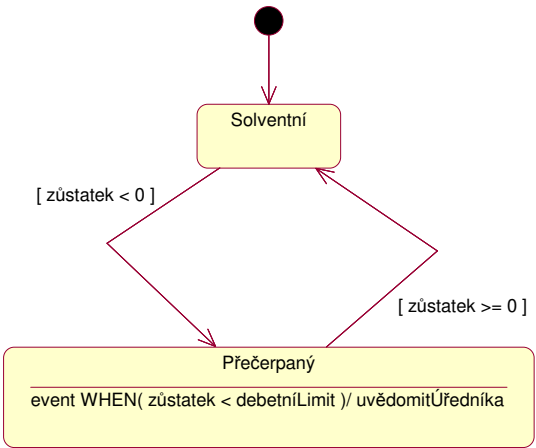
# Diagram aktivít

- rozvětvení, spojení
- oddíly aktivit



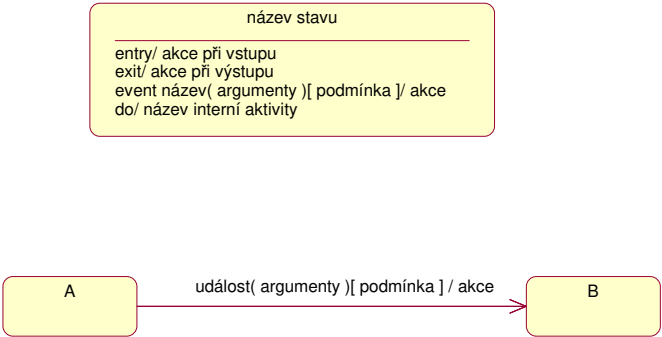
# Stavový diagram

- událost změny (WHEN)



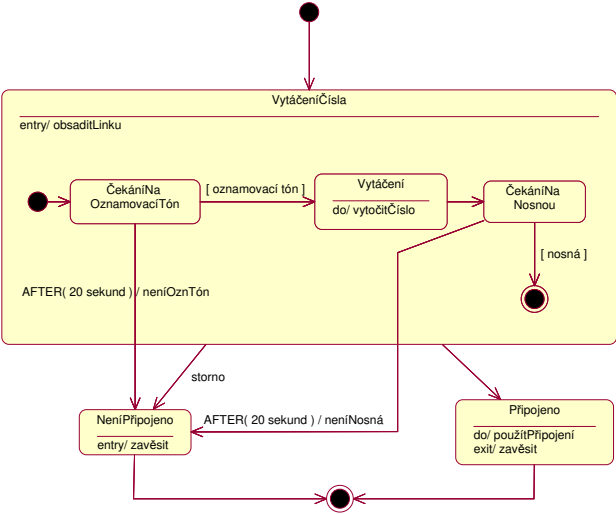
# Stavový diagram

- stav
- přechod mezi stavy



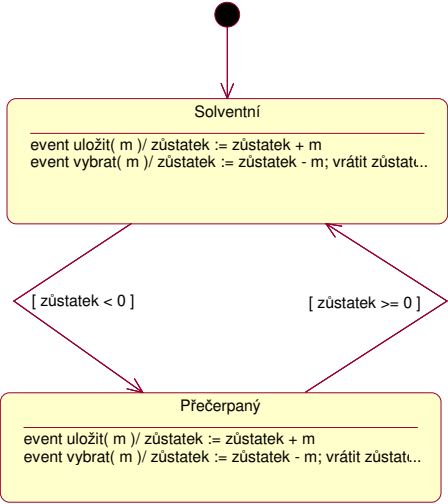
# Stavový diagram

- sekvenční složený stav
- událost AFTER



# Stavový diagram

- událost volání



# Volby do AS FIT

## Volby do AS FIT pro funkční období 2019 až 2022

- Volby proběhnou elektronicky dne **24. října 2019**.
- Vybíráte až 4 kandidáty v obvodu studentů bakalářských a magisterských studijních programů.
- Písemný návrh na kandidáta lze podat do 16. října 2019 na sekretariát děkana FIT.
- <https://www.fit.vut.cz/fit/news/2520/>

59 / 59

# Akademický senát VUT v Brně

## AS VUT v Brně

- schvaluje předpisy platné pro celou školu,
- volí kandidáta na rektora a schvaluje rozpočet VUT v Brně,
- je volen akademickou obcí celé školy,
- je složen ze dvou komor:
  - komora akademických pracovníků
  - studentská komora
- FIT zastupují 2 zaměstnanci a 1 student,
- <https://www.vutbr.cz/vut/struktura/as>

### Vnitřní předpisy VUT, které se nejvíce dotýkají studentů:

- Studijní a zkušební řád VUT v Brně,
- Stipendijní řád VUT v Brně,
- Disciplinární řád VUT v Brně.
- <https://www.vutbr.cz/uredni-deska/vnitрни-predpisy-a-dokumenty>

57 / 59

# Akademický senát FIT

## AS FIT VUT v Brně

- schvaluje předpisy FIT, které doplňují vnitřní předpisy VUT v Brně,
- volí děkana a schvaluje rozpočet FIT,
- je volen akademickou obcí FIT (a tedy i studenty),
- je složen ze dvou komor:
  - komora akademických pracovníků (8 členů)
  - studentská komora (4 + 1 člen)
- <https://www.fit.vut.cz/fit/as>

### Vnitřní předpisy a normy FIT, které se nejvíce dotýkají studentů:

- Pravidla o organizaci studia na FIT,
- Registrace předmětů a individuální studijní plány v předmětech v bakalářském a magisterském studiu,
- Podmínky pro přiznání stipendia studentům FIT,
- Disciplinární řád pro studenty FIT,
- <https://www.fit.vut.cz/fit/info/predpisy/>
- <https://www.fit.vut.cz/fit/info/smernice/>

58 / 59