

SOFTVÉROVÉ MODELOVANIE

5.

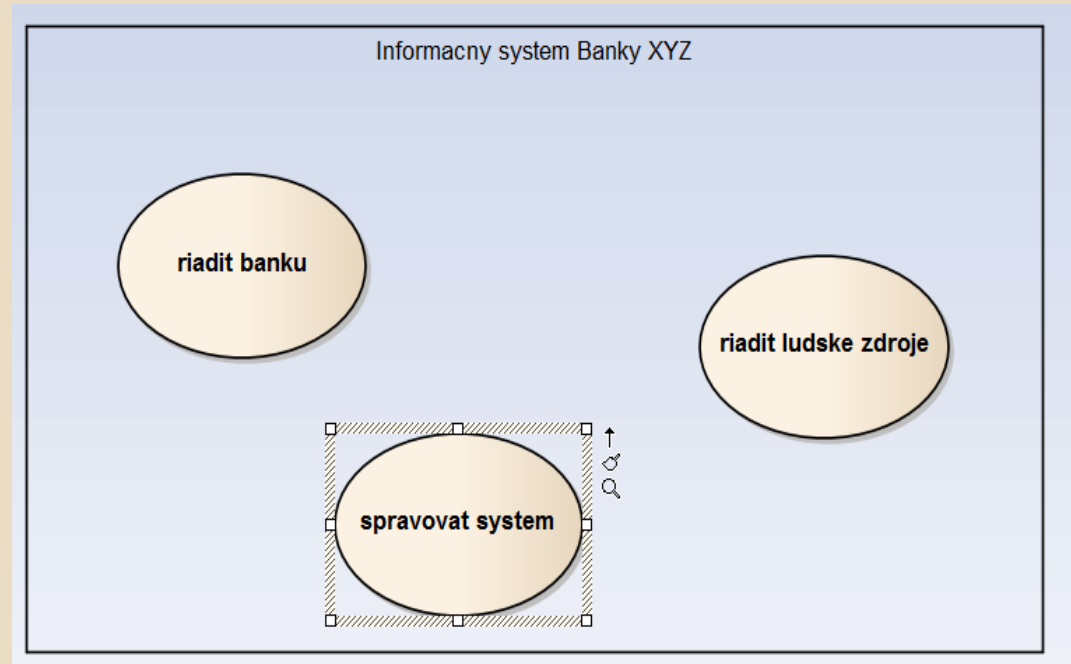
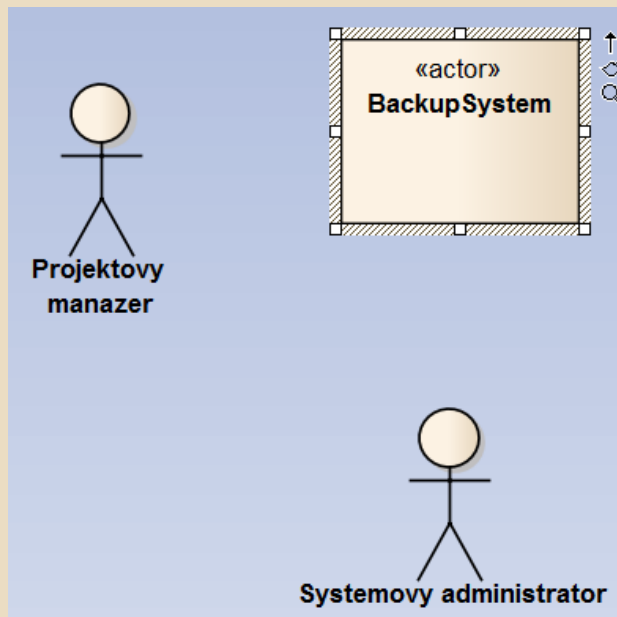
Ján Ružbarský
Marek Tavač

Obsah 5. prednášky

- *Opakovanie*
- ***Stavové diagramy***
 - Stavy
 - Prechody
 - Ďalšie možnosti
- *Vaše otázky*

Opakovanie – prípady použitia

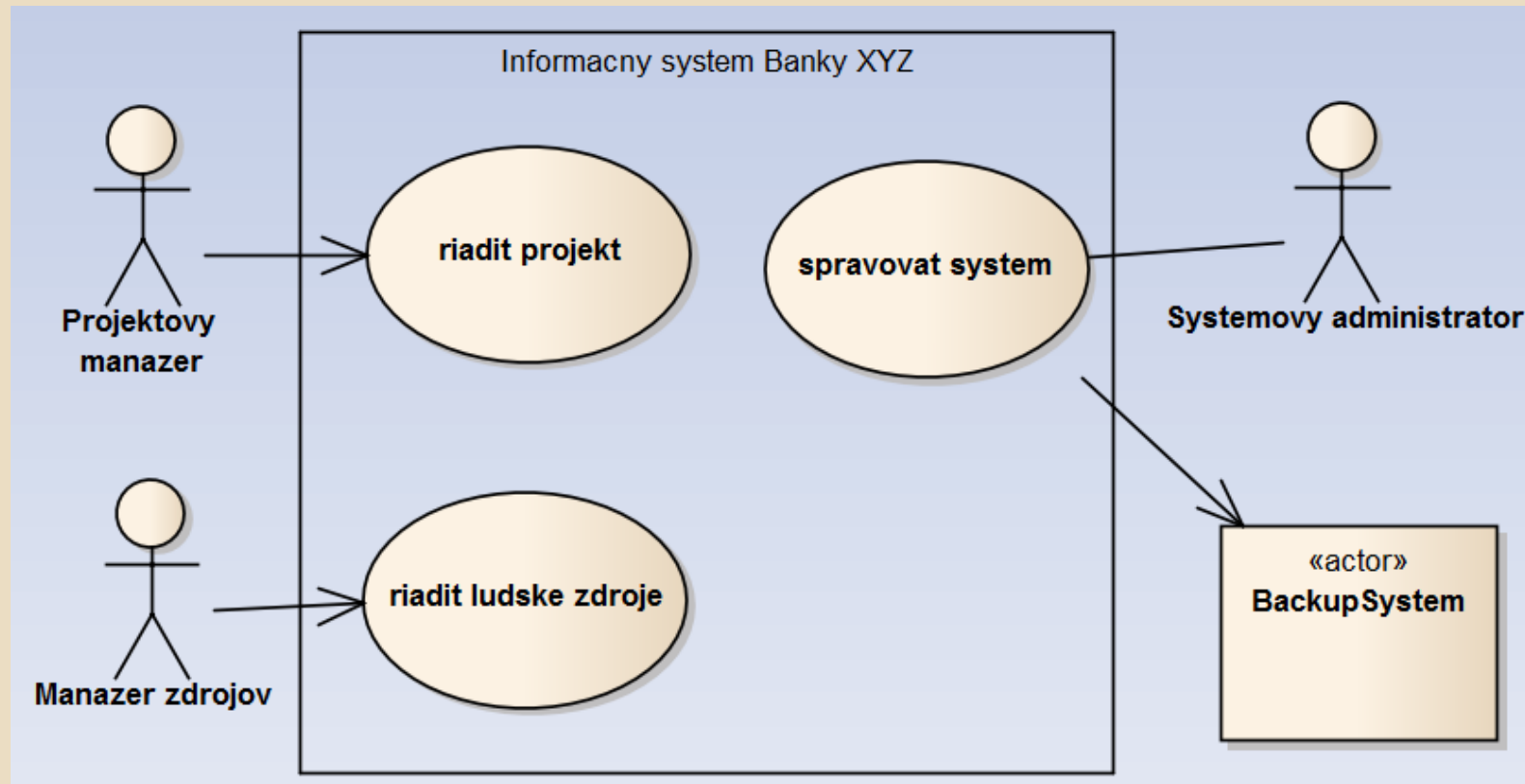
- Popisujú funkčnosť systému, jeho použitie
- UML – žiadny návod ako zachytiť požiadavky
- **Aktor, prípad použitia (use case)**



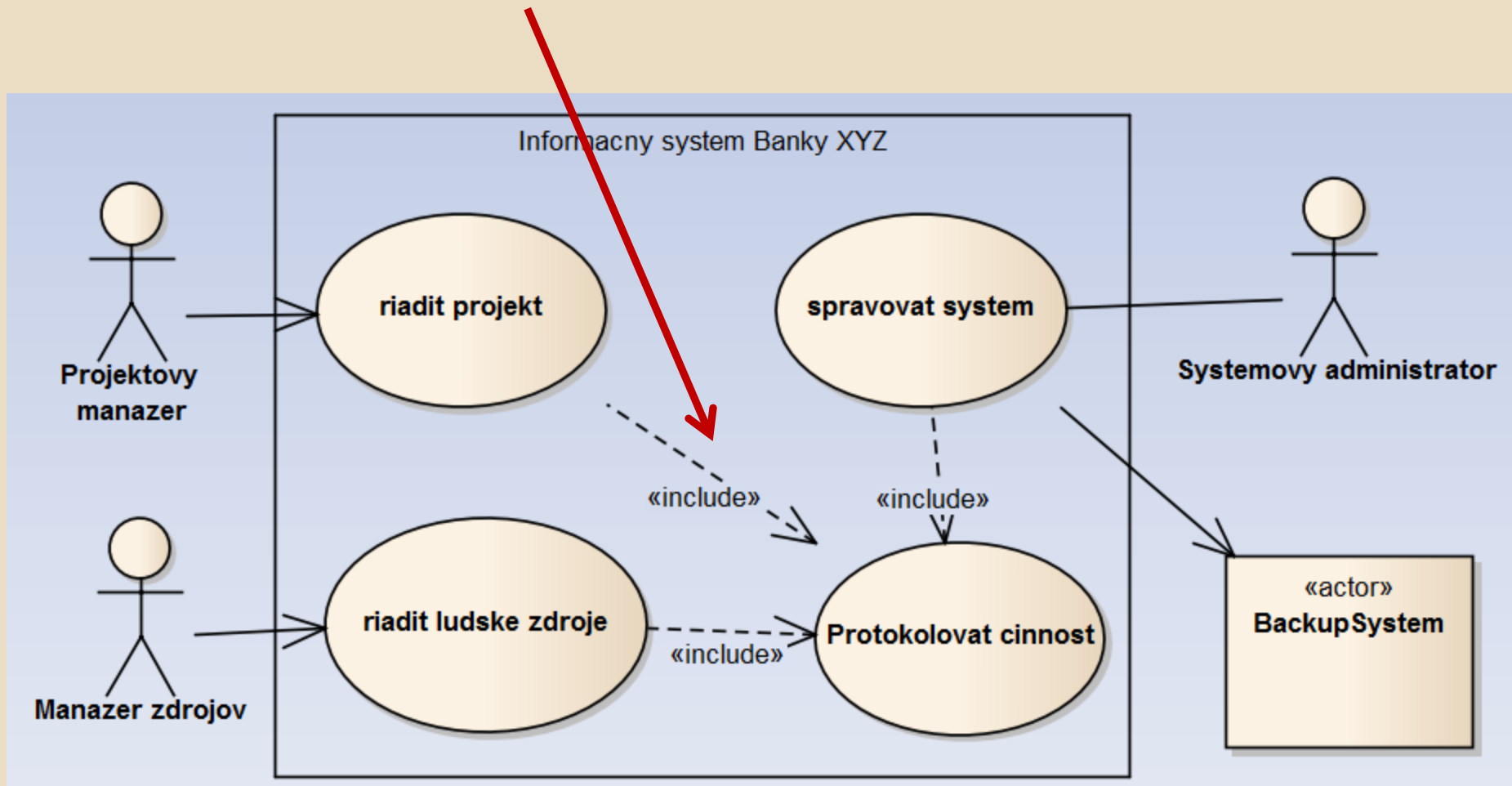
Opakovanie - scenár

- **Scenár**
 - konkrétne vykonávanie use case
 - slúži väčšinou na diskusiu
 - **hlavný úspešný scenár**
 - **sekvencie správania** (postupnosť krokov)
 - **pre-condition, post-condition (quarantee)**
- *Vzt'ah prípad použitia – príbeh v XP*

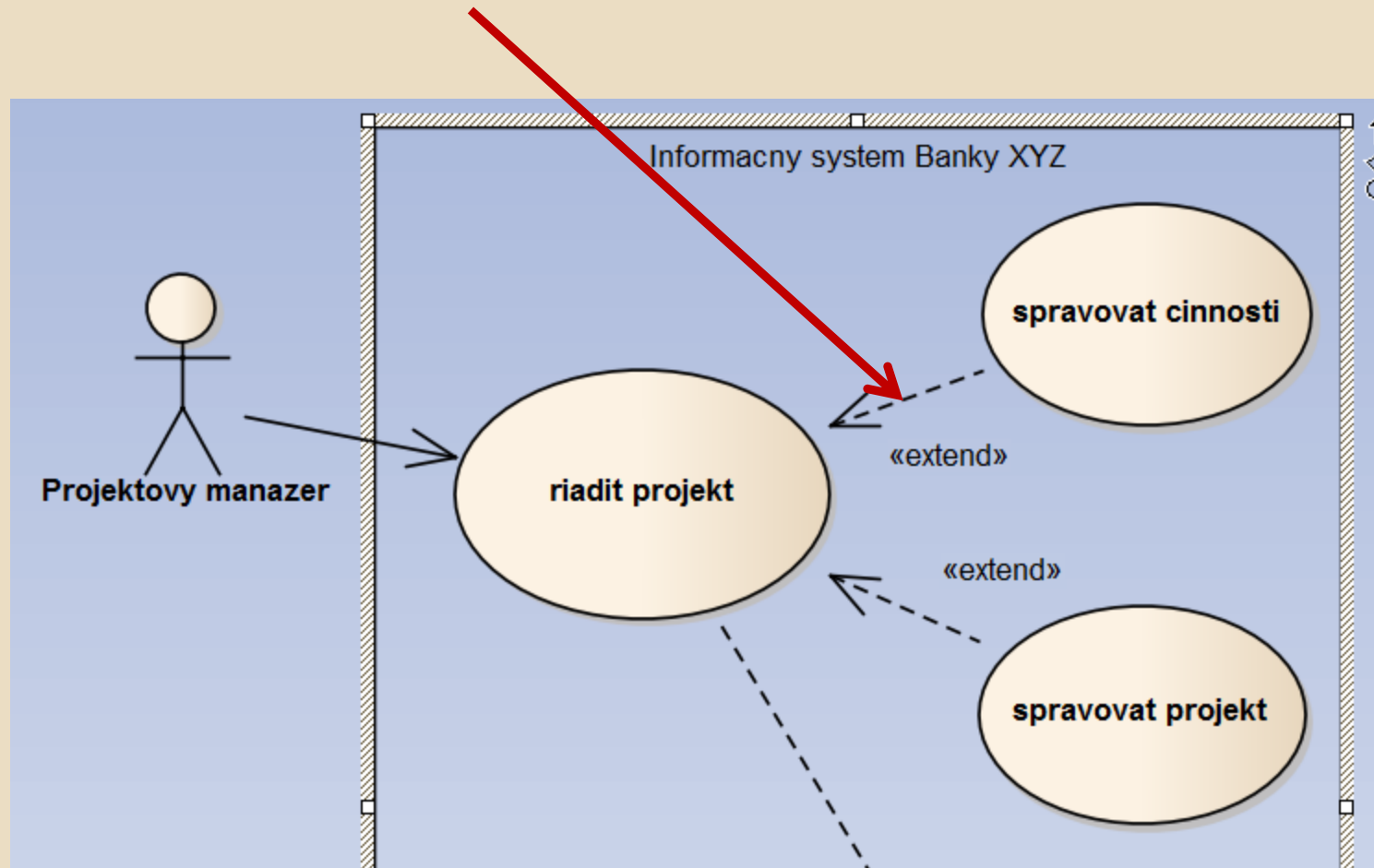
Opakovanie – komunikačná asociácia



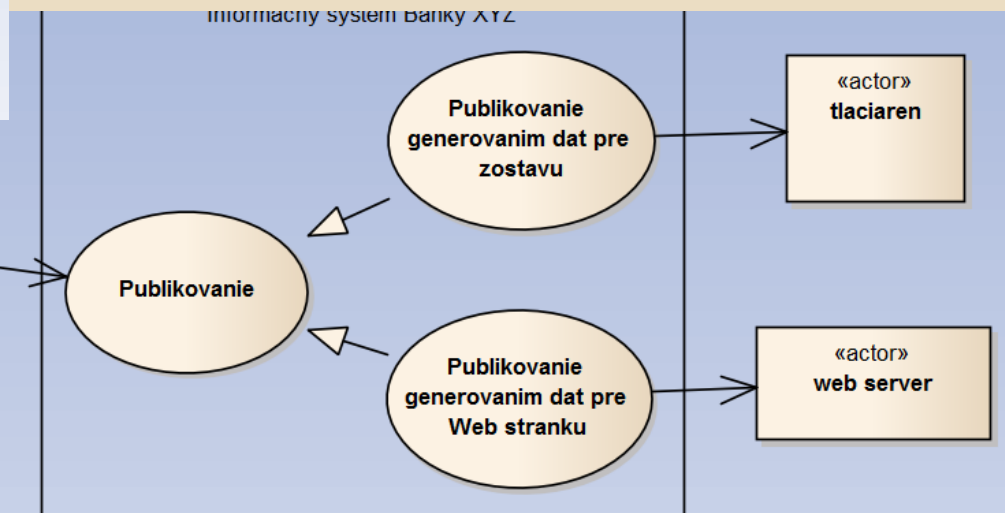
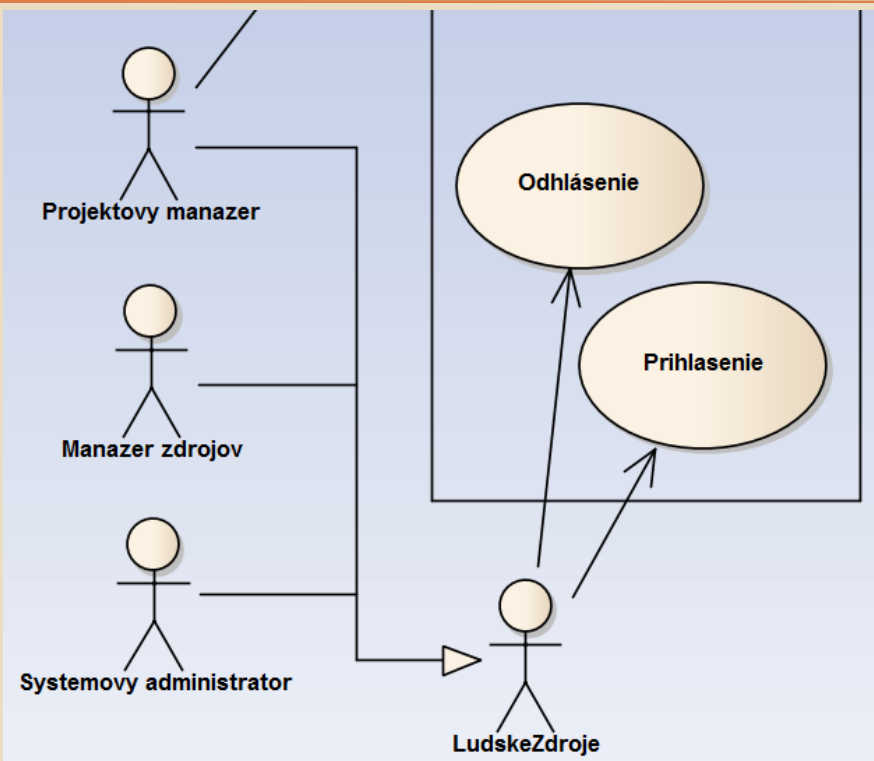
Opakovanie – include



Opakovanie - extend



Opakovanie - generalizácia



- *Generalizácia ?*
- *Extend ?*

Opakovanie - zásady tvorby use case

- *Úplnosť modelu - najdôležitejšia*
 - Nechýba žiadny use case
 - Konzistencia modelu
 - **ZLATÉ** use case, obslužné use case
- *Aktéry nie sú najdôležitejší*
- *Chyby pri tvorbe use case*
 - Málo informácií, príliš veľa informácií
 - Analytik viazaný na implementáciu
 - Časová závislosť medzi prípadmi použitia

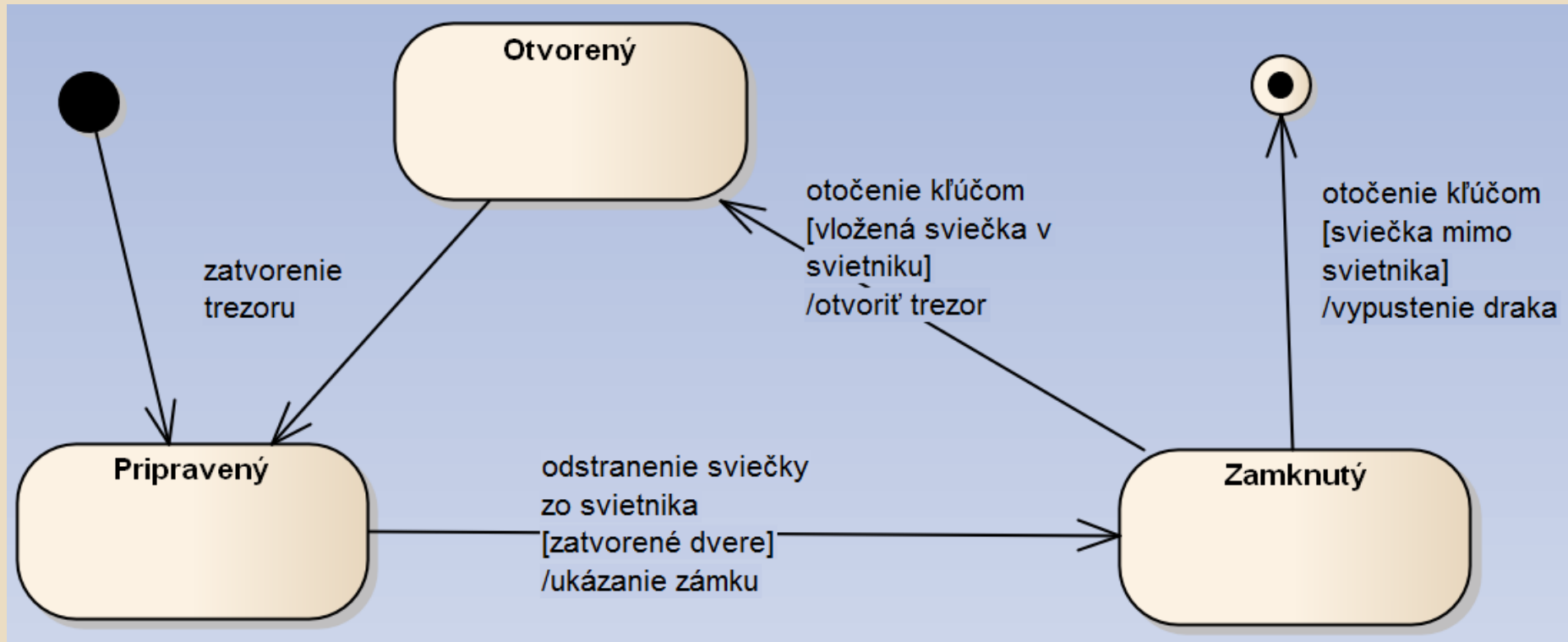
State machine (stavový automat)

- *dynamické aspekty systému*
- *modelujú správanie individuálnych objektov*
 - inštancie triedy, use casu alebo celého systému
- *postupnosť stavov počas životného cyklu*
- *zrozumiteľné, jednoduché, efektívne*

Stavové diagramy

- *Stavový diagram v UML vizualizuje state machine (stavový automat)*
- *všetky stavy*
- *podmienky prechodu medzi stavmi*

Príklad



Stavy - States

- *prvky medzi sebou komunikujú*
- *životný cyklus prvku*
- **STAV** - *konkrétna situácia alebo podmienka, v ktorej sa prvok nachádza počas svojho životného cyklu*
- *aktívny stav – aktuálny stav*
- *počiatočný a koncový stav*

Počiatočný a koncový stav

➤ *pseudostavy*

➤ *Počiatočný stav (initial)*

- Ukazuje stav prvku, keď je vytváraný



➤ *Koncový stav (final)*

- Stav prvku v momente rušenia



Jednoduchý stav

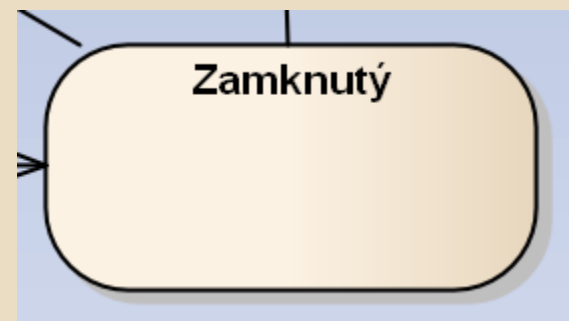
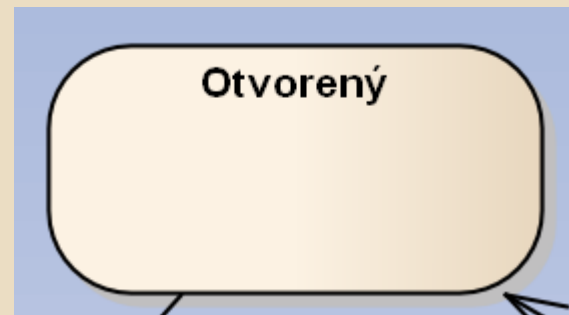
➤ situácia, v ktorej sa prvok nachádza

➤ Príklad:

Neaktívny

Aktívny

Prerušený

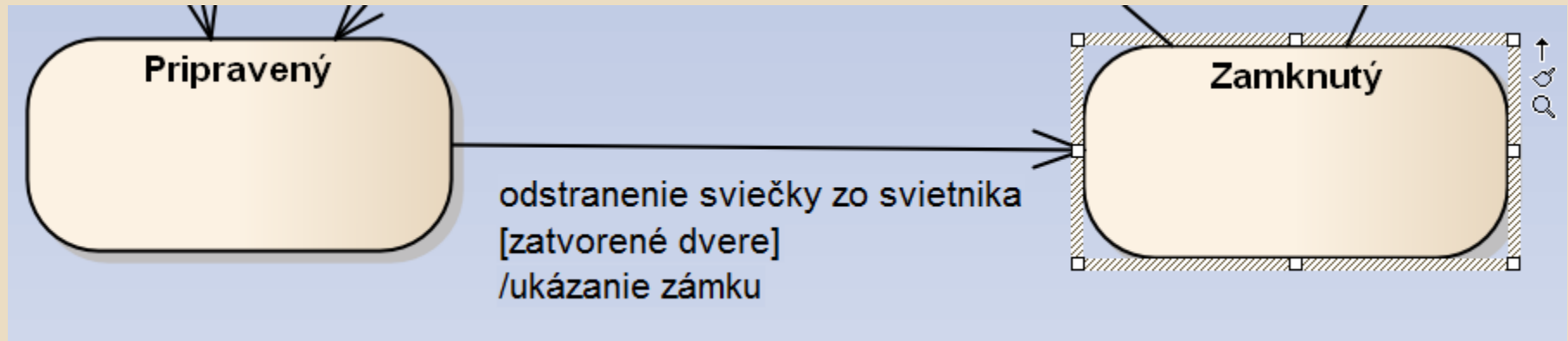


Prechody - Transitions

- vzťahy medzi stavmi
- prechod „**horí**“
- medzi dvoma stavmi aj viac prechodov
- plná čiara so šípkou od zdrojového stavu k cieľovému stavu
 1. Prvok je v zdrojovom stave
 2. Nastane udalosť, je splnená podmienka
 3. Vykoná sa akcia
 4. Prvok je v cieľovom stave

Prechody - Transitions

- *Zdrojový stav (**source state**)*
- *Spúšťacia udalosť (**event trigger**)*
- *Ochranná podmienka (**guard condition**)*
- *Efekt, akcia (**effect**)*
- *Cieľový stav (**target state**)*

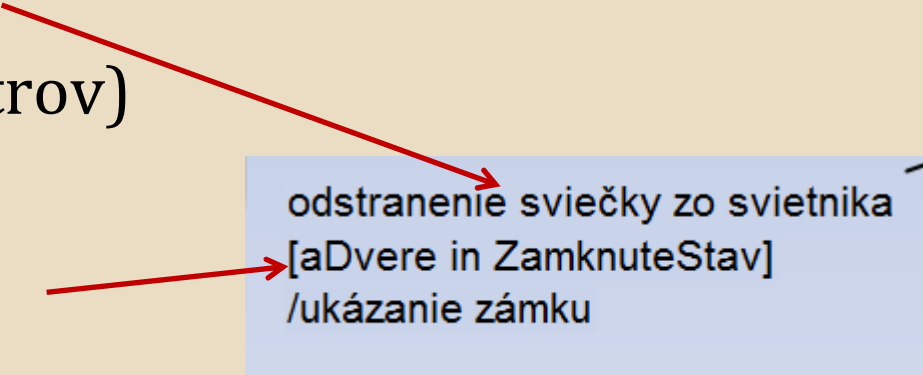


Spúšťacia udalosť[ochranná podmienka]/efekt

- automatický prechod

➤ *Spúšťacia udalosť*

- Meno (zoznam parametrov)



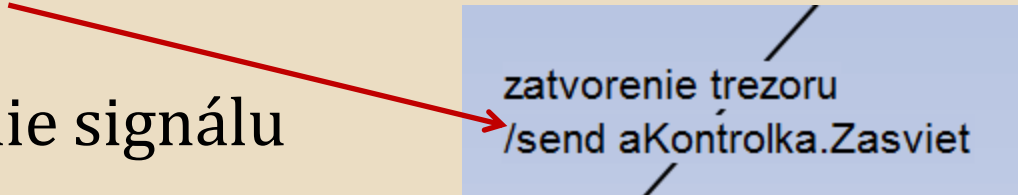
odstranenie sviečky zo svietnika
[aDvere in ZamknuteStav]
/ukázanie zámku

➤ *Ochranná podmienka*

- [podmienka]

➤ *Efekt*

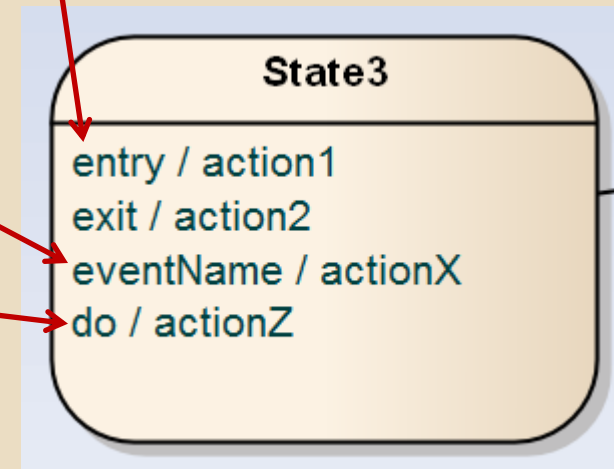
- Návratová premenná=cieľový objekt.meno akcie(zoznam parametrov)
- send – posielanie signálu



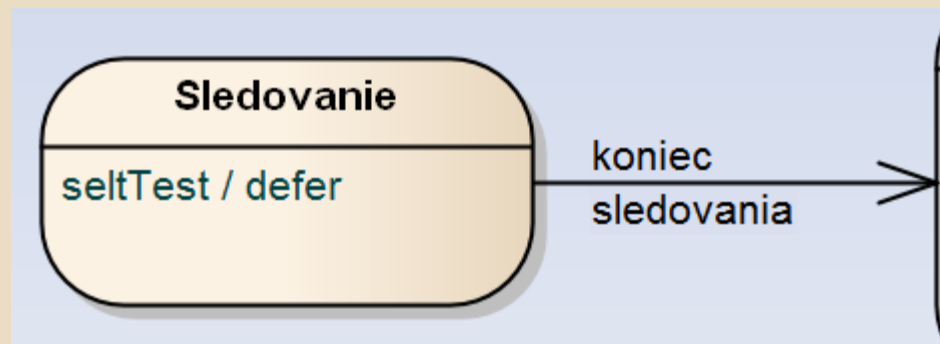
zatvorenie trezoru
/send aKontrolka.Zasviet

Ďalšie možnosti

- *Meno stavu – môže byť aj anonymný stav*
- *Vstupné/výstupné akcie (entry/exit effects) - majú sa vykonať vždy pri vstupe do alebo odchode zo stavu*
- *Interné prechody (internal transitions) – pri reakcii na udalosť sa nemení stav*
- *Do-aktivity – aktivity vnútri stavu*



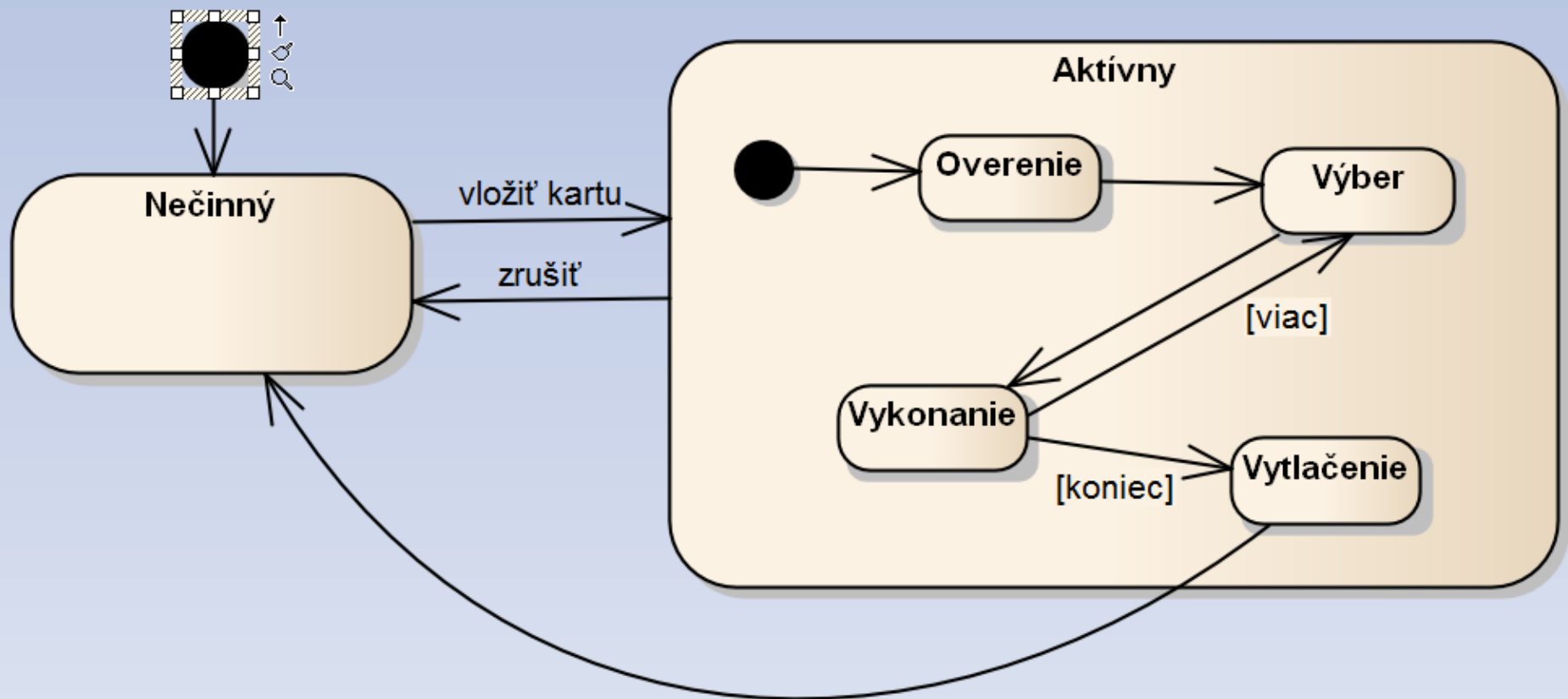
➤ *Odložené udalosti (deferred events)*



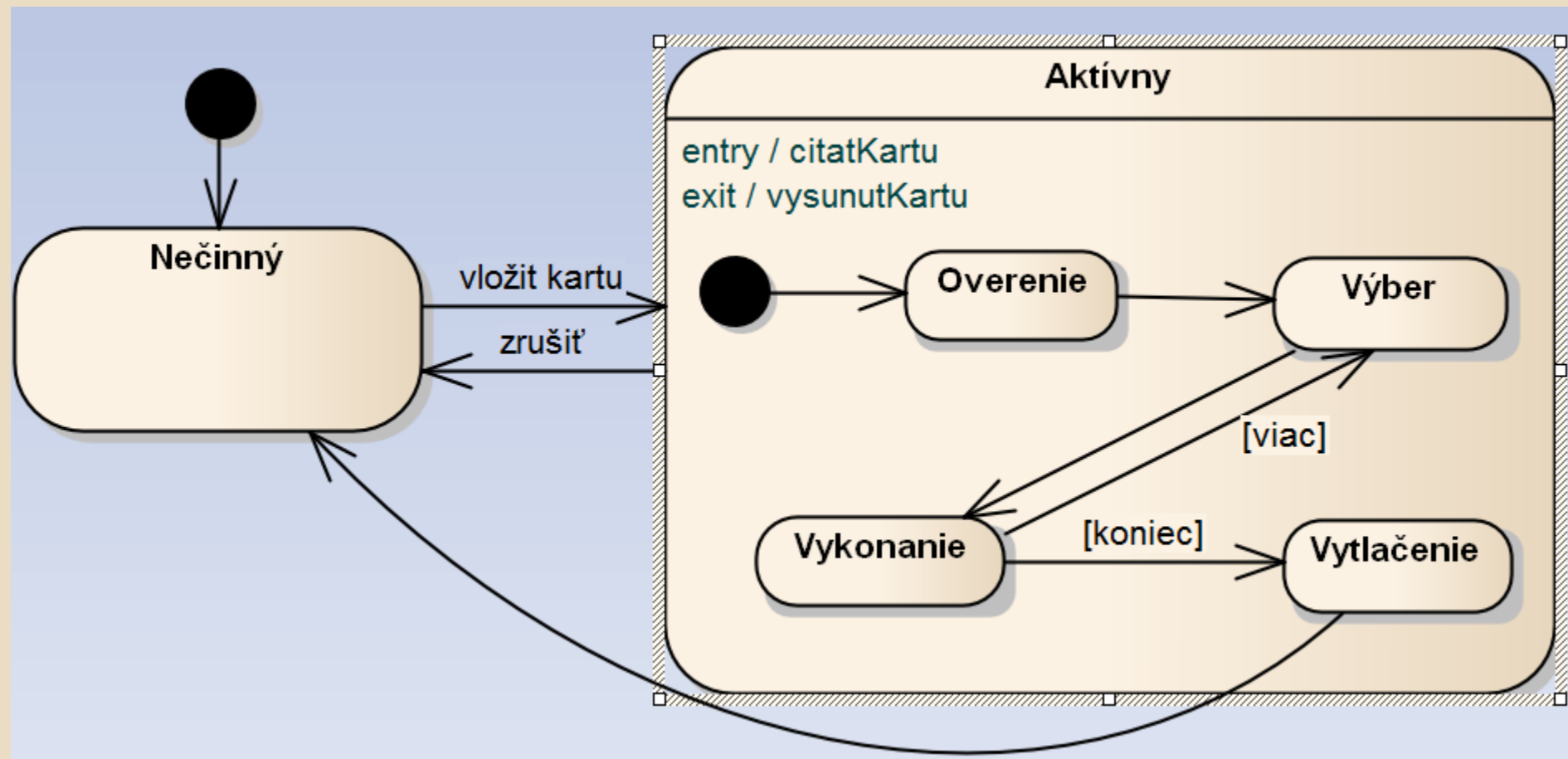
Substavy - substates

- ***substate (substav)*** – stav vložený do iného stavu
- ***composite state (zložený stav)***
 - súbežné (orthogonal) substavy
 - sekvenčné (nonorthogonal) substavy

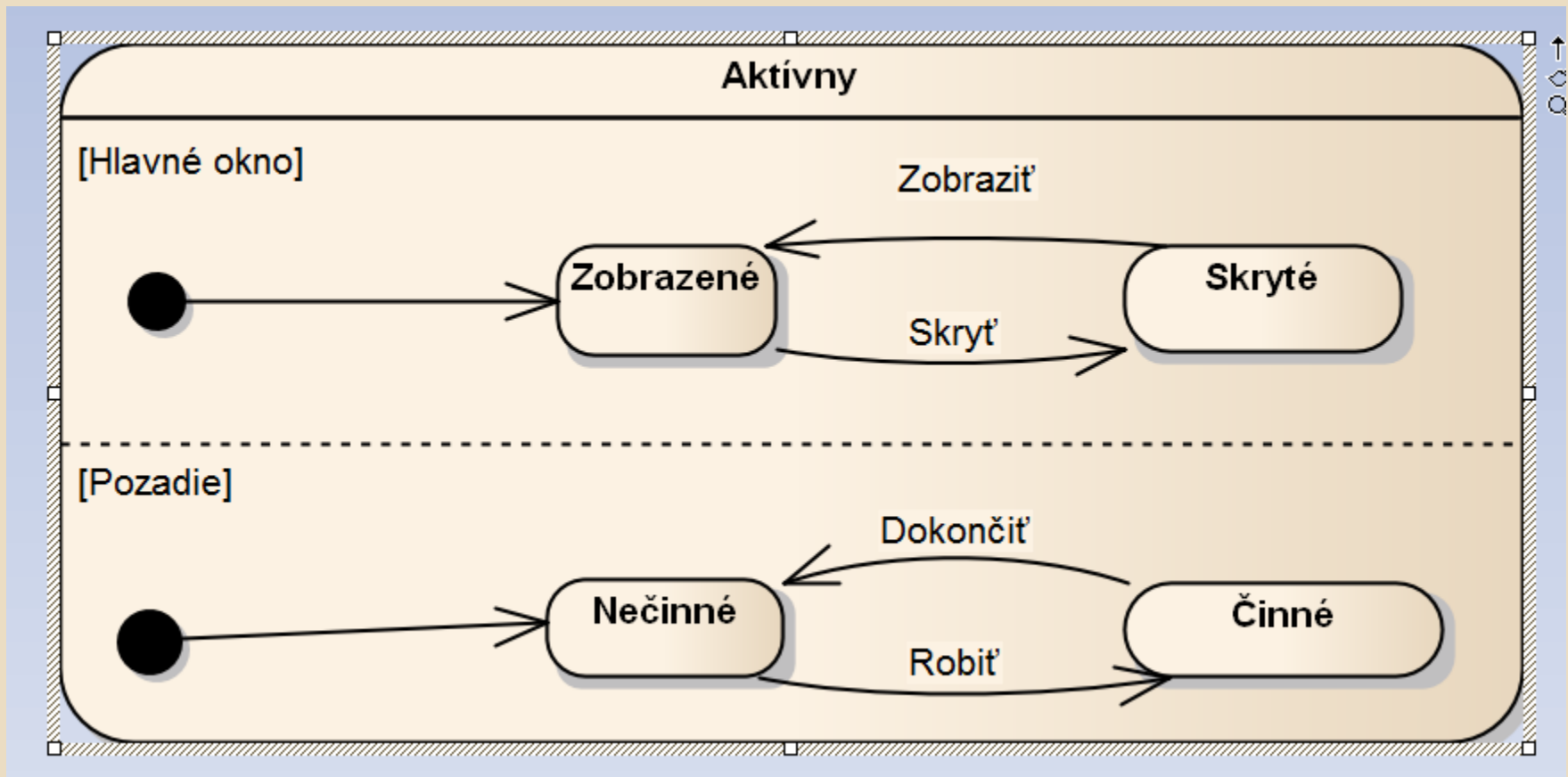
Sekvenčné systémy



Sekvenčné systémy

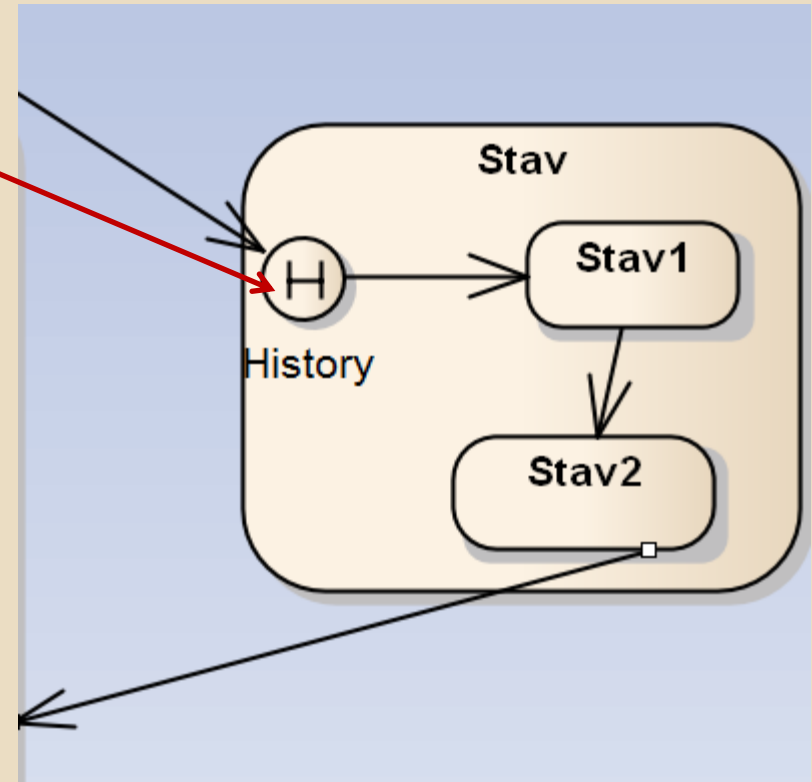


Súbežné substavy



History states

- *pamätá si posledný aktívny substav*
- *plytká história*
- *hlboká história*



Ako na to ?

- *Určte čo modelujeme stavovým automatom ?*
 - *Trieda, use case – hľadám susedov, rodičov*
 - *Celý systém – zameriam sa na správanie systému ako celku*
- *Nadefinujte počiatočný a koncový(-é) stav(-y)*
- *Hľadajte udalosti – interface*
- *Hľadajte top-level stavy a spájate ich prechodmi*
- *Identifikujte entry/exit akcie*
- *Rozšírte stavy o substavy – ak je potrebné*
- *Skontrolujte udalosti - interface*
- *Skontrolujte akcie – vzťahy, operácie a metódy*
- *Skontrolujte postupnosť stavov*

Otázky ?

Ďakujem za pozornosť