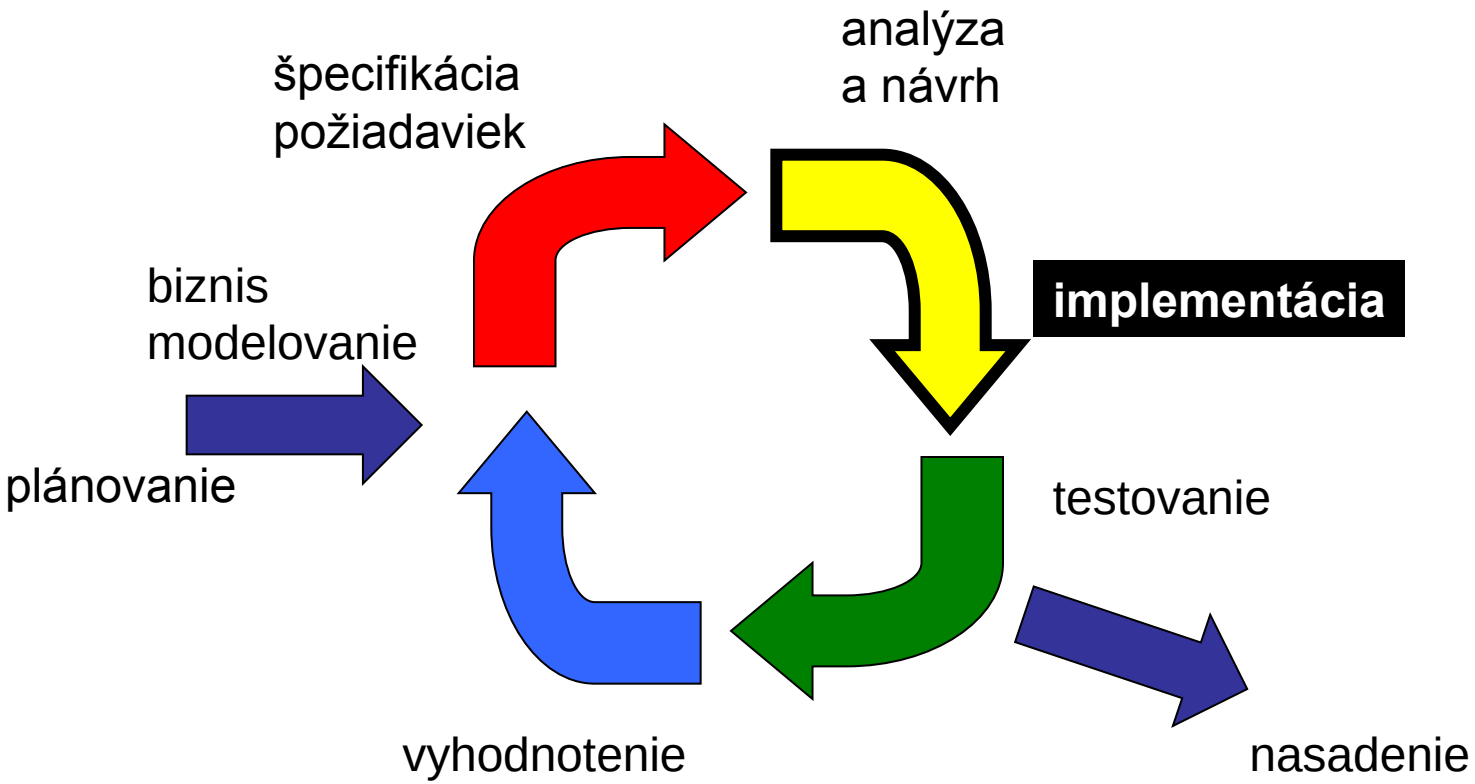
Implementácia, testovanie, nasadenie

KST Katedra softvérových technológií
Fakulta riadenia a informatiky, Žilinská univerzita v Žiline

RUP – iterácie



Cieľ implementácie

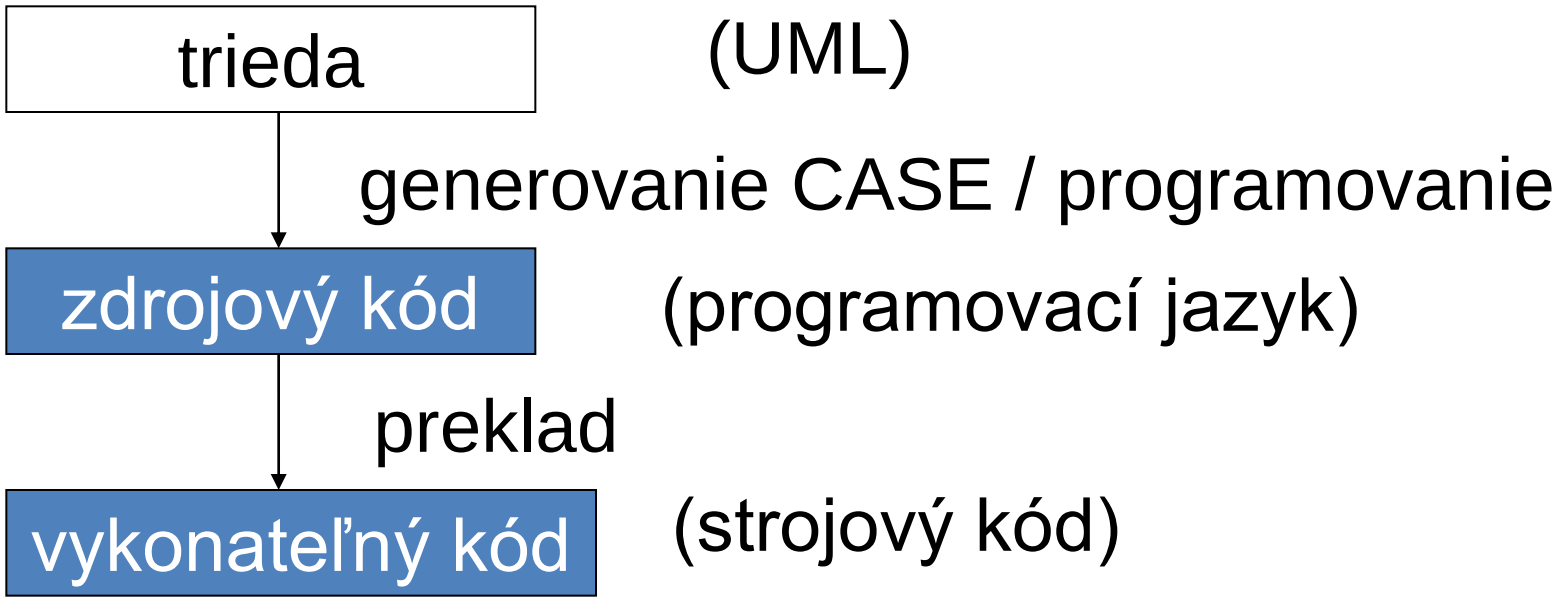
- vytvoriť fungujúci softvér
- model implementácie tvoria softvérové komponenty
  - zdrojový kód prvkov z modelu návrhu
  - vykonateľný (binárny kód) – preklad
  - databázové tabuľky
- konkrétny programovací jazyk
- konkrétny databázový systém

Produkty implementácie

- softvérový komponent – súbor
  - zdrojový,
  - binárny
  - vykonateľný
- UML diagram komponentov
  - štruktúra a závislosti medzi komponentmi
- UML diagram nasadenia
  - spresňuje štruktúru „hardvéru“
  - určuje rozmiestnenie softvérových komponentov

Generovanie/kódovanie - preklad

- softvérový komponent – súbor (zdrojový, binárny kód) – jedna (alebo viac tried)



Komponent UML

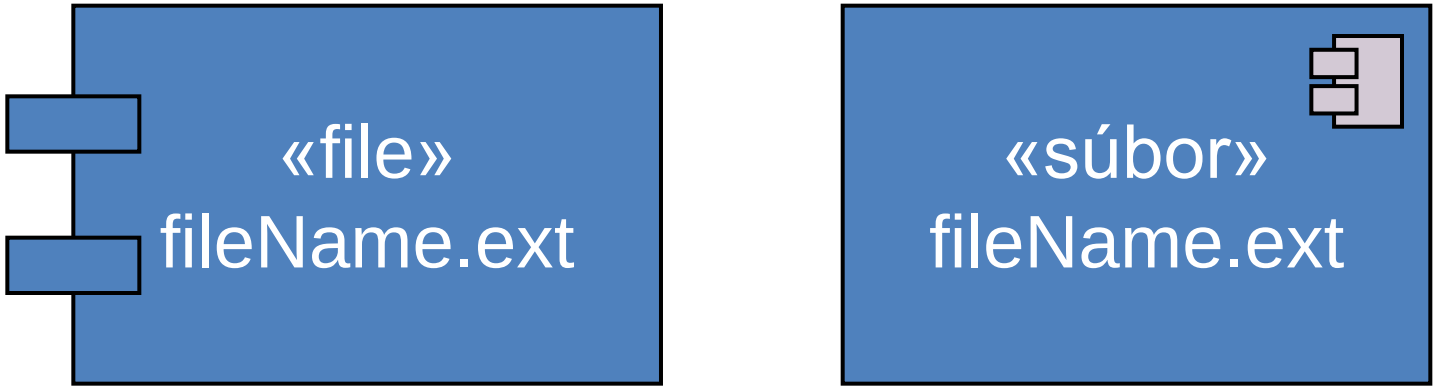


Diagram tried

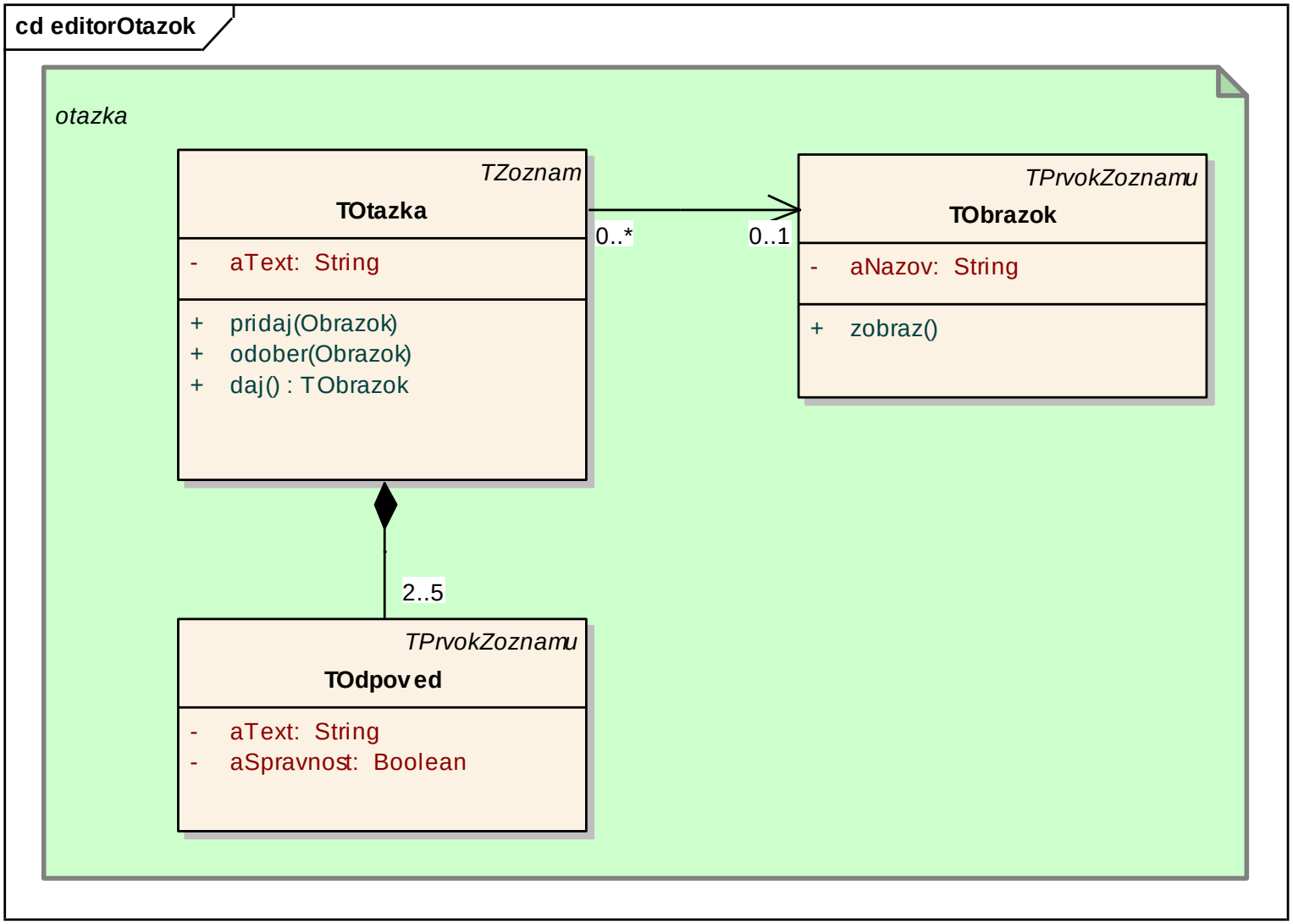
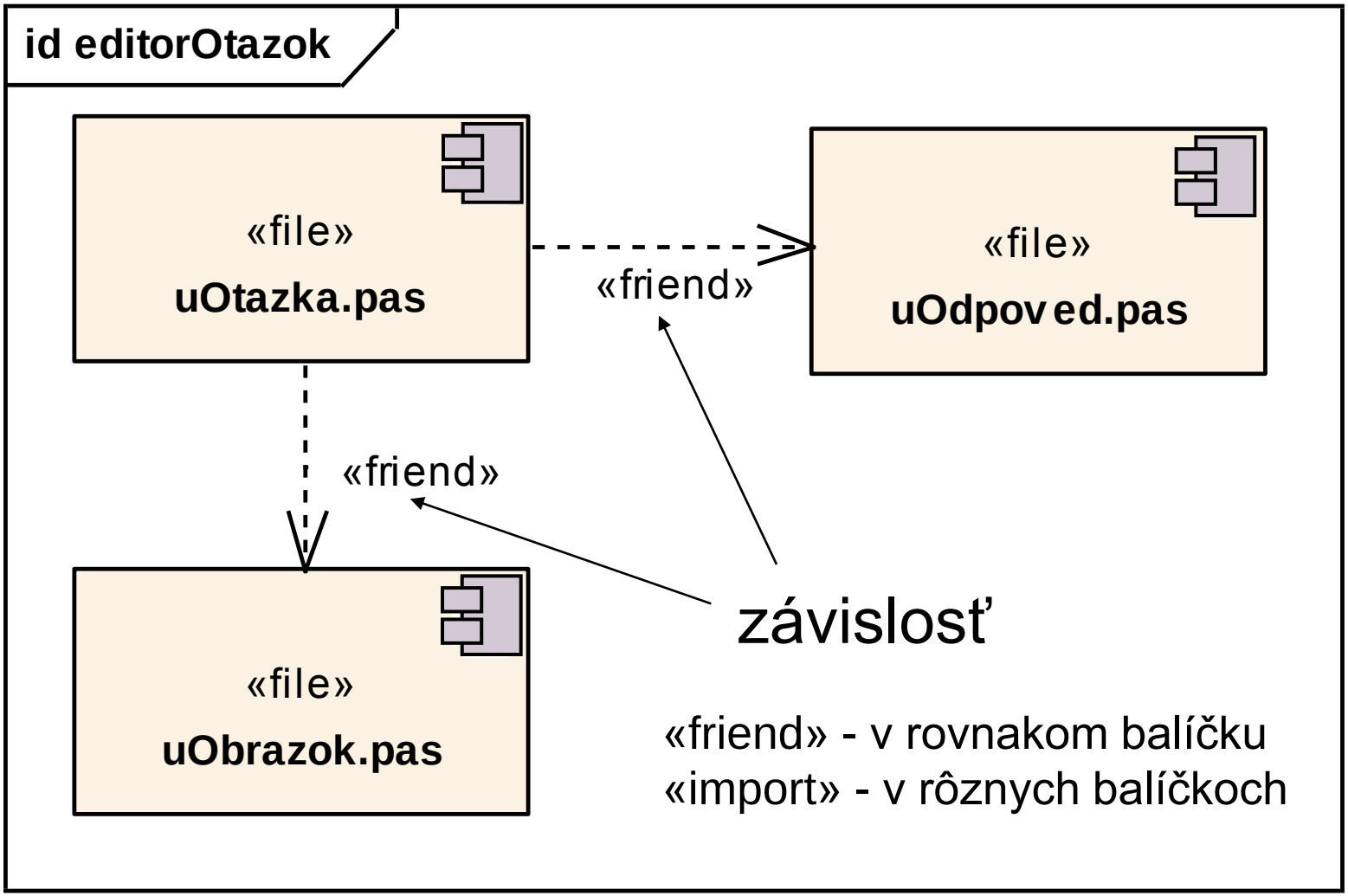
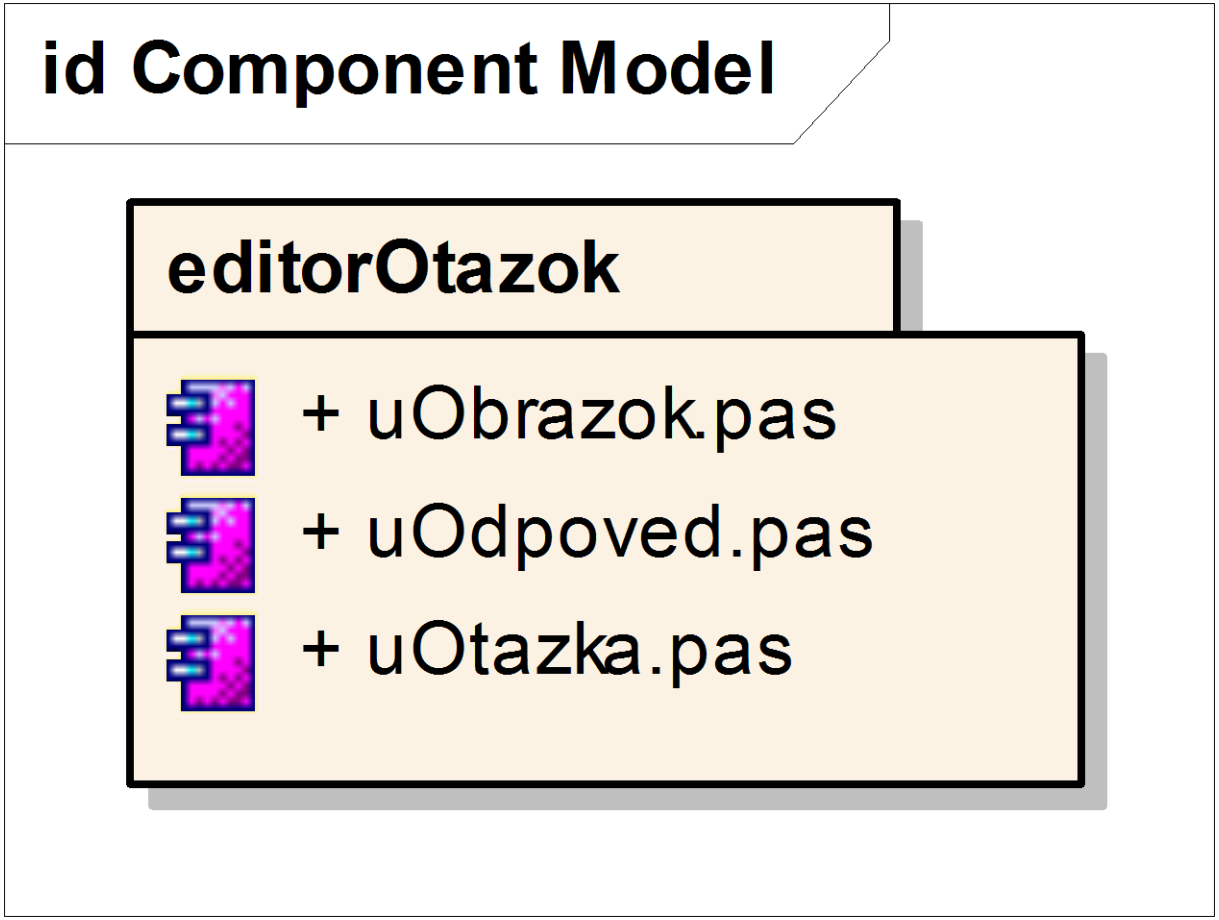


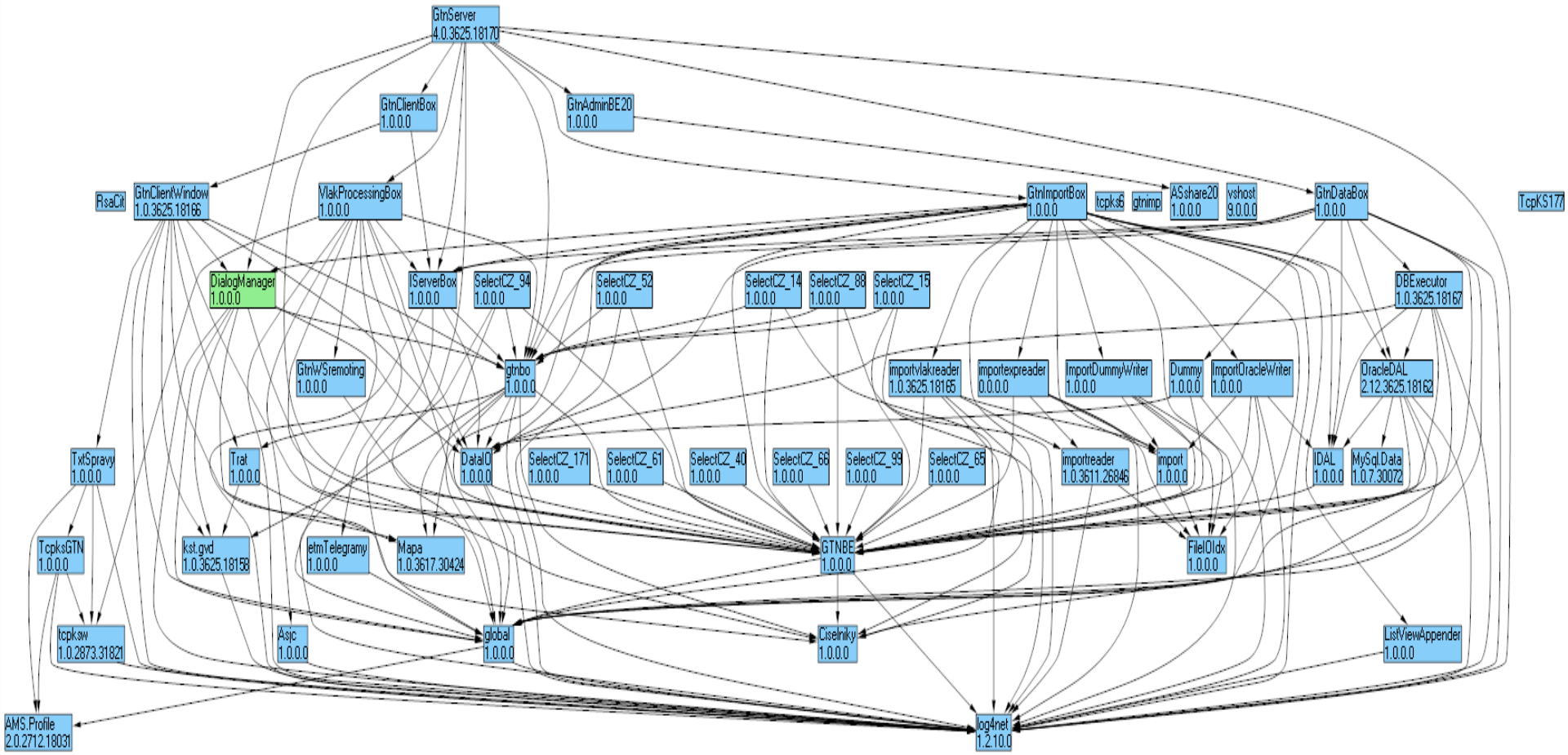
Diagram komponentov



Balíček komponentov



Reálny projekt



CASE: generovanie kódu

UML	*Programovací jazyk*
Trieda	Class
Rozhranie, rola	Interface
Operácia	Metóda
Atribút	Atribút
Asociácia	Atribút
Závislosť	Lokálna premenná, parameter
Prípad použitia	Postupnosť správ
Balíček	Adresár

Generovanie triedy - Deklarácia

```
TOtazka = class (TZoznam)
 private
  aText: String;
 public
  aTOdpoved: TOdpoved;
  aTObrazok: TObrazok;
  procedure pridaj(paObrazok: TObrazok);
  procedure odober(paObrazok: TObrazok);
  function daj: TObrazok;
  constructor Create; overload;
  destructor Destroy; override;
end;
```

atribút

kompozícia

asociácia

operácia

atribúty

metódy

Generovanie triedy - Implementácia

```
destructor TOtazka.Destroy;
begin
  inherited Destroy;
end;

function TOtazka.daj: TObrazok;
begin

end;
```

doprogramovať ručne!

stavový diagram
diagram aktivít
sekvenčný diagram

delphi - ukázať

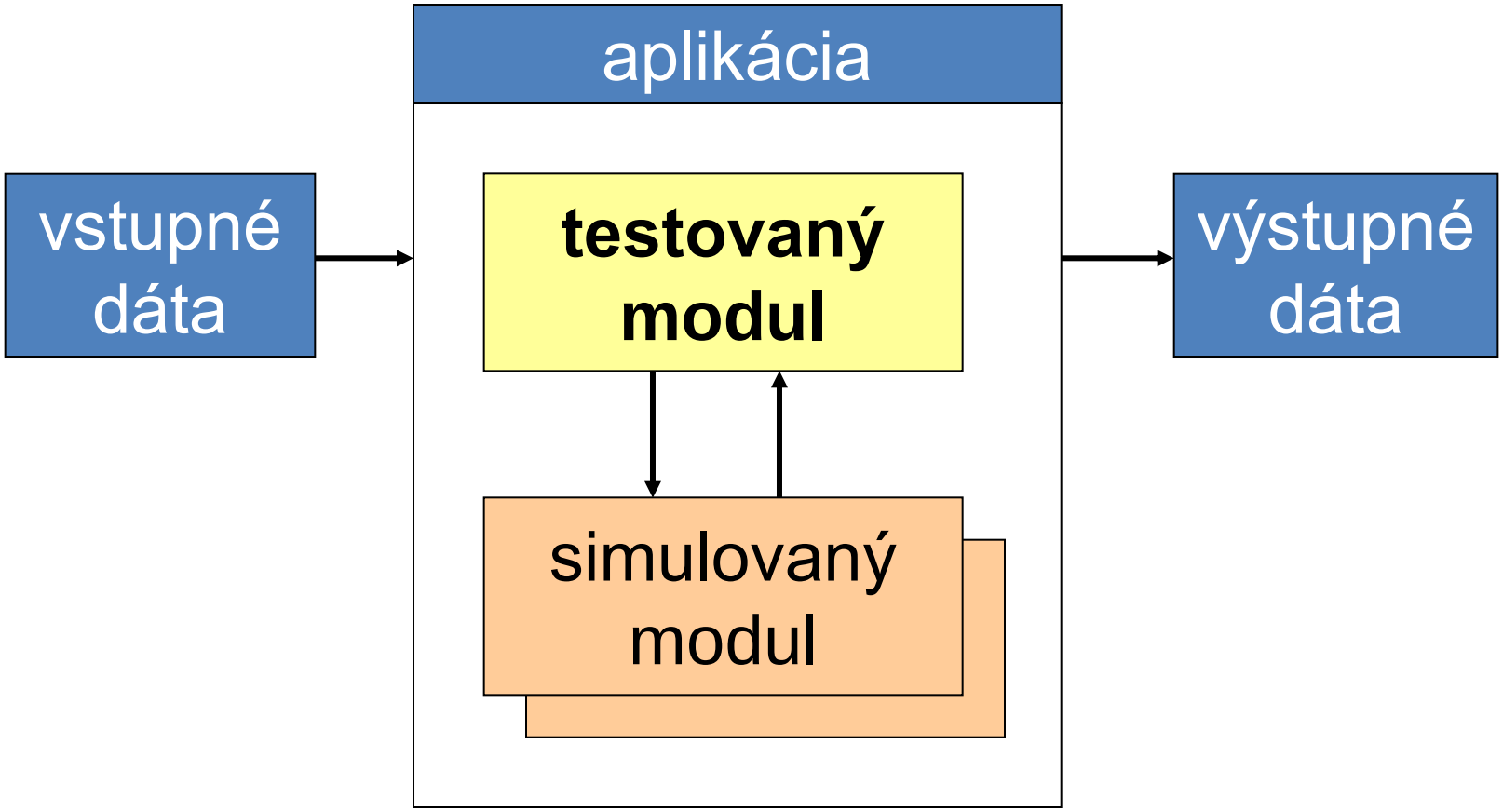
Stratégia implementácie

postup, ako sa realizujú jednotlivé softvérové súčiastky

- zhora nadol
- zdola nahor
  - -vplyv stratégie návrhu
  - -vplyv na stratégiu testovania

- štýl programovania
- princípy objektového a štruktúrovaného programovania

Implementácia zhora nadol

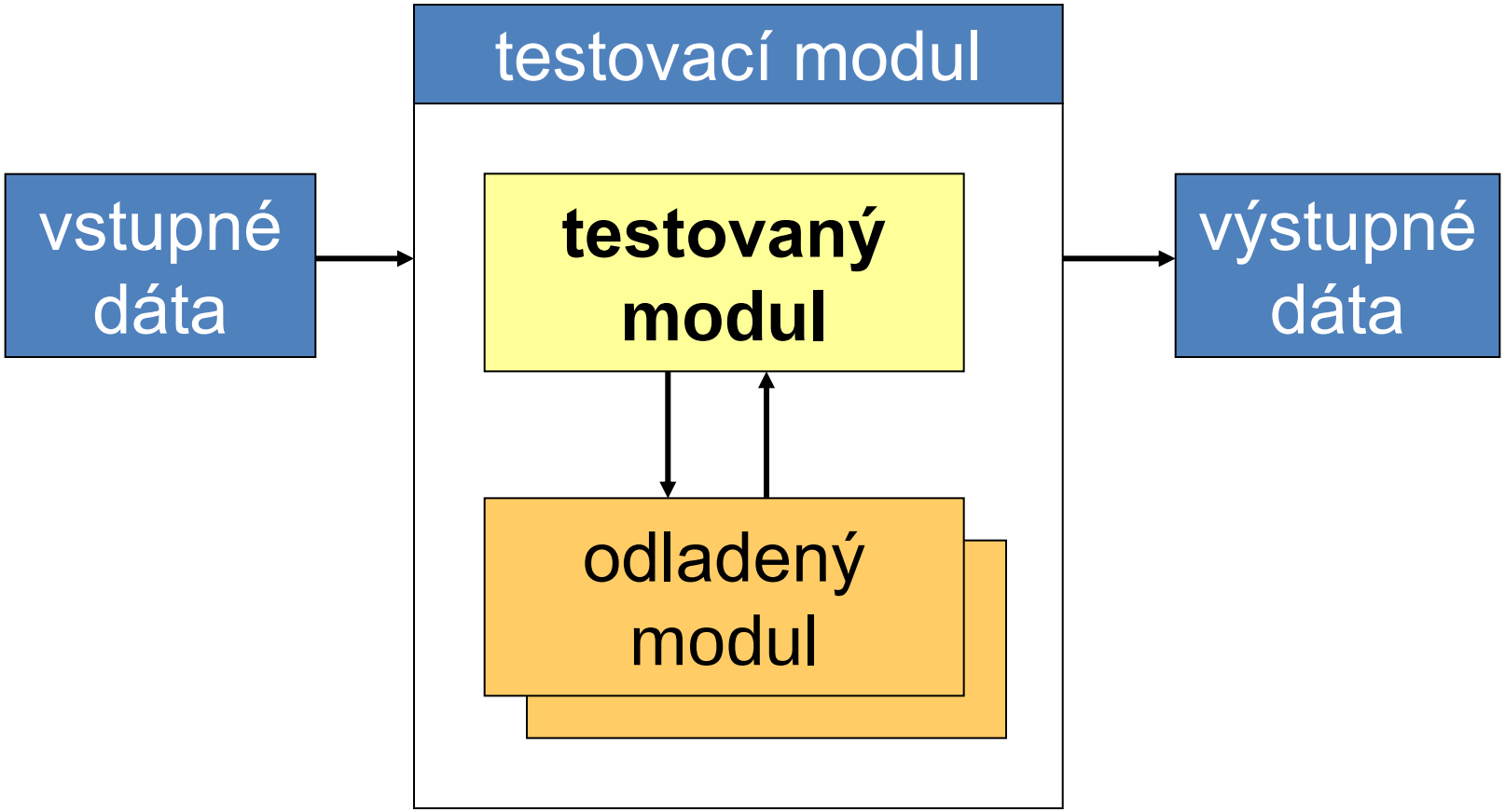


Implementácia zhora nadol

výhody
- skorá prezentácia systému
- skorá identifikácia závažných chýb
- viacnásobné overenie logiky systému

nevýhody
- testovanie modulov je jednoduchšie
- neexistujúce moduly sa simulujú

Implementácia zdola nahor



Implementácia zdola nahor

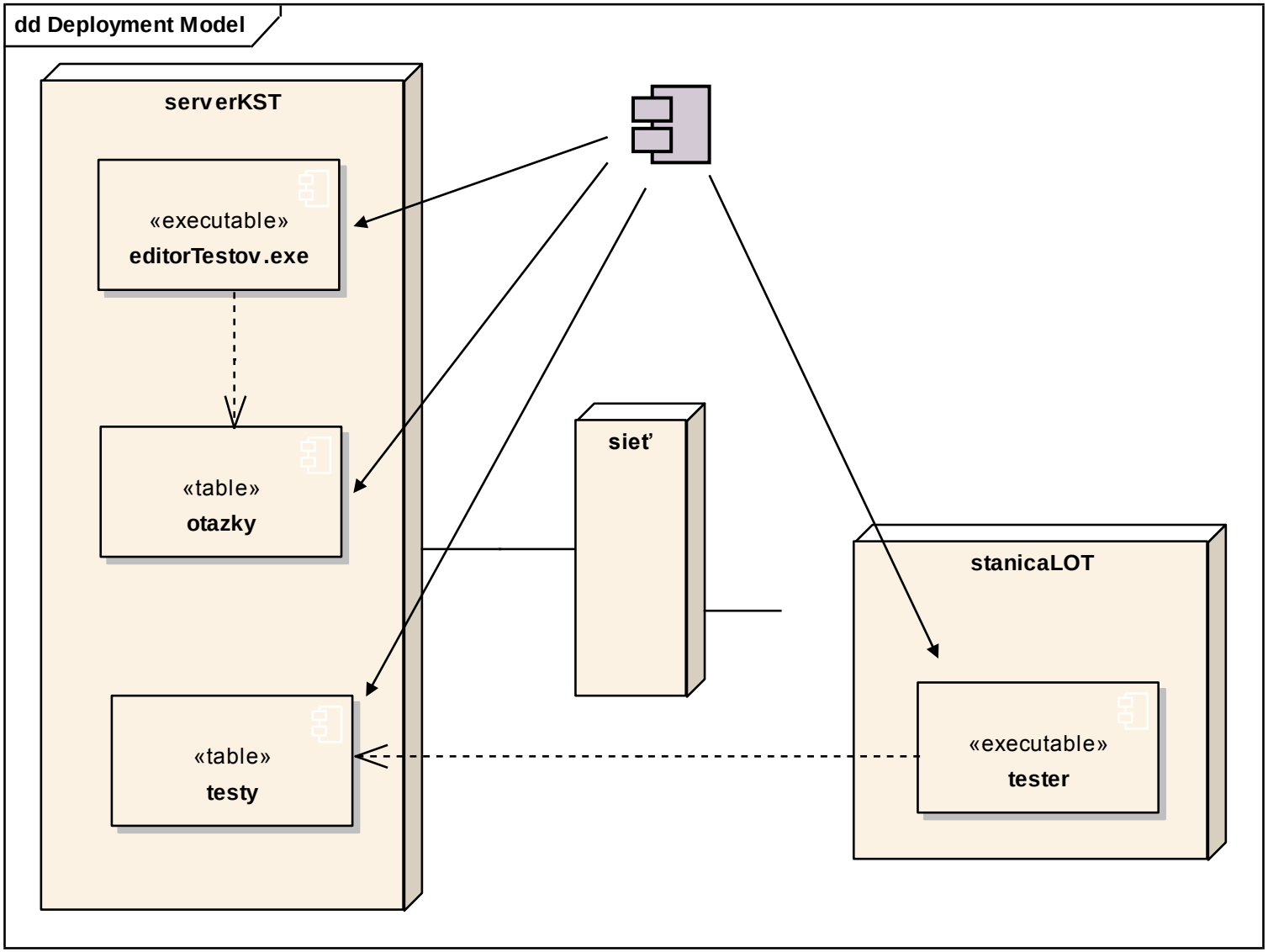
výhody
- samostatné odladenie modulov

nevýhody
- logika systému sa testuje až nakoniec
- systém sa prezentuje až po dokončení
- tvorba špeciálnych testovacích modulov

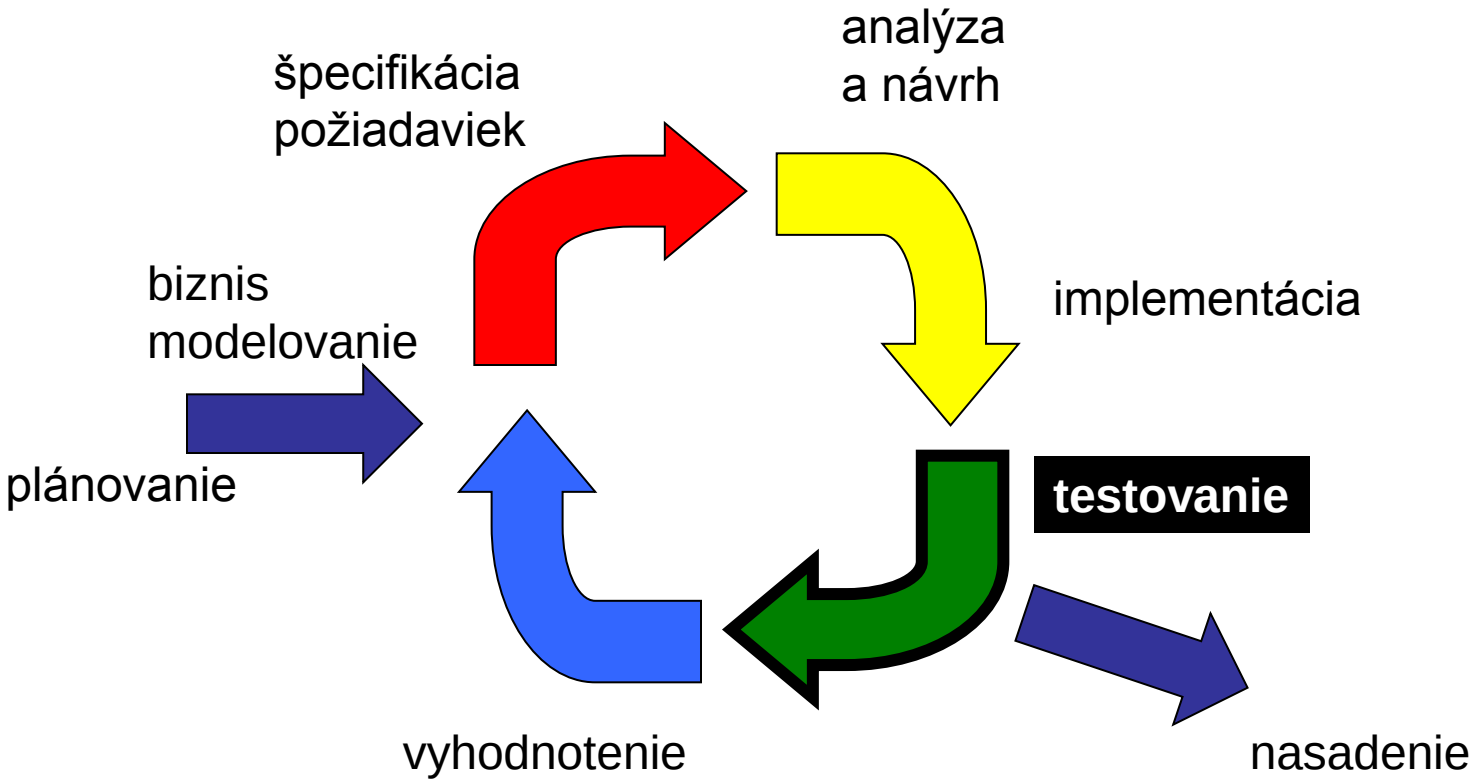
Štýl programovania

- mená samo dokumentujúce
- mená konzistentné – rovnaký spôsob tvorby v celom systéme
- ukrývanie informácií
- vnáranie – pozor na priveľkú hĺbku
- vedľajšie efekty minimalizovať
- pretty programming – odsadenie, jeden prikaz – jeden riadok...

Diagram nasadenia



RUP – iterácie



Model testovania

UML nemá žiadny diagram

Model tvoria:

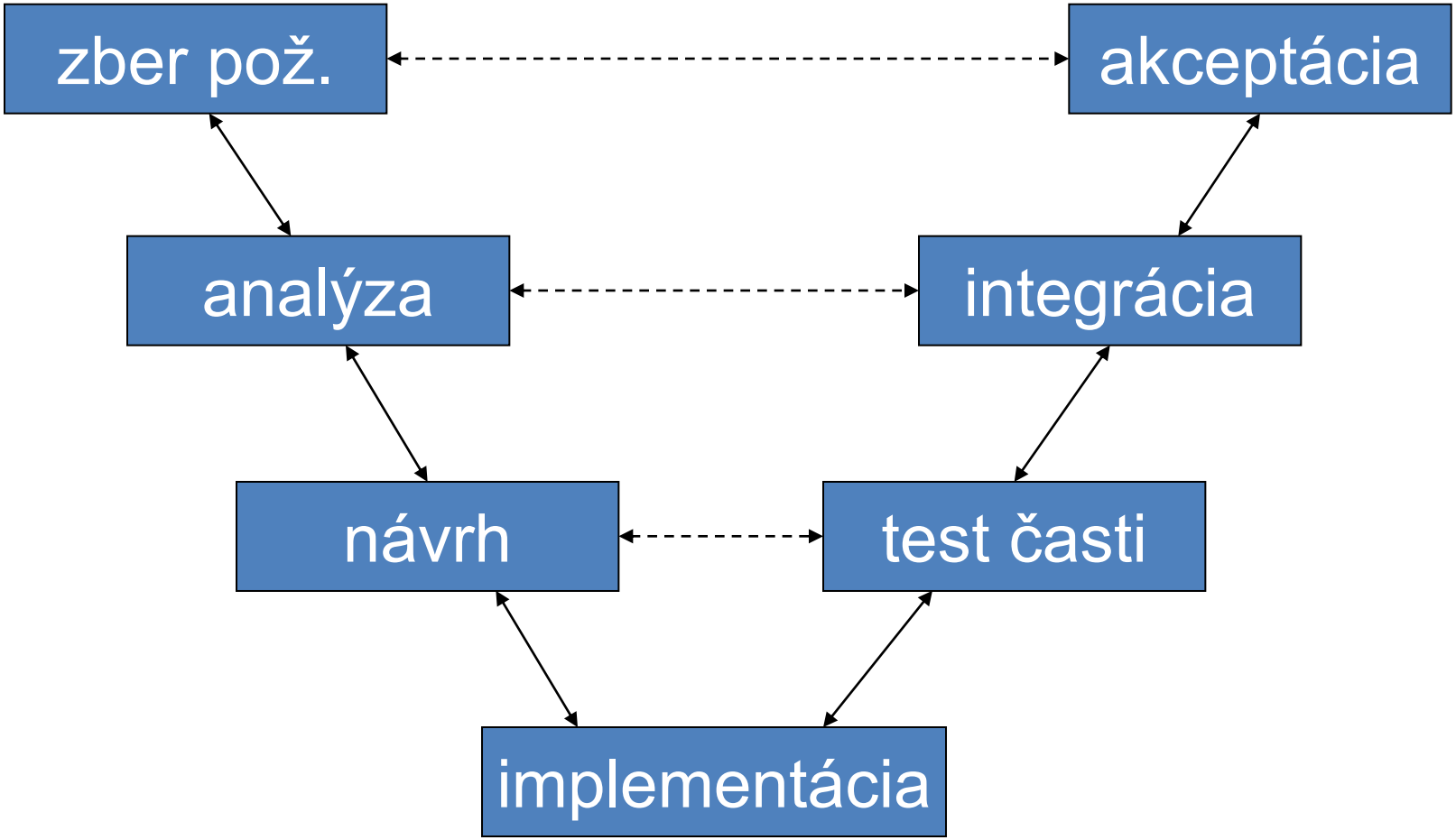
- testovacie úlohy – čo testovať, aké vstupy, výstupy
- testovacie procedúry – ako vykonať úlohu
- testovacie komponenty – automatizácia procedúry

Testovanie

pre každú iteráciu

- integračné testy – v priebehu iterácie
  - každý vytvorený produkt
- systémové testy – koniec iterácie
  - spustiteľná verzia aplikácie

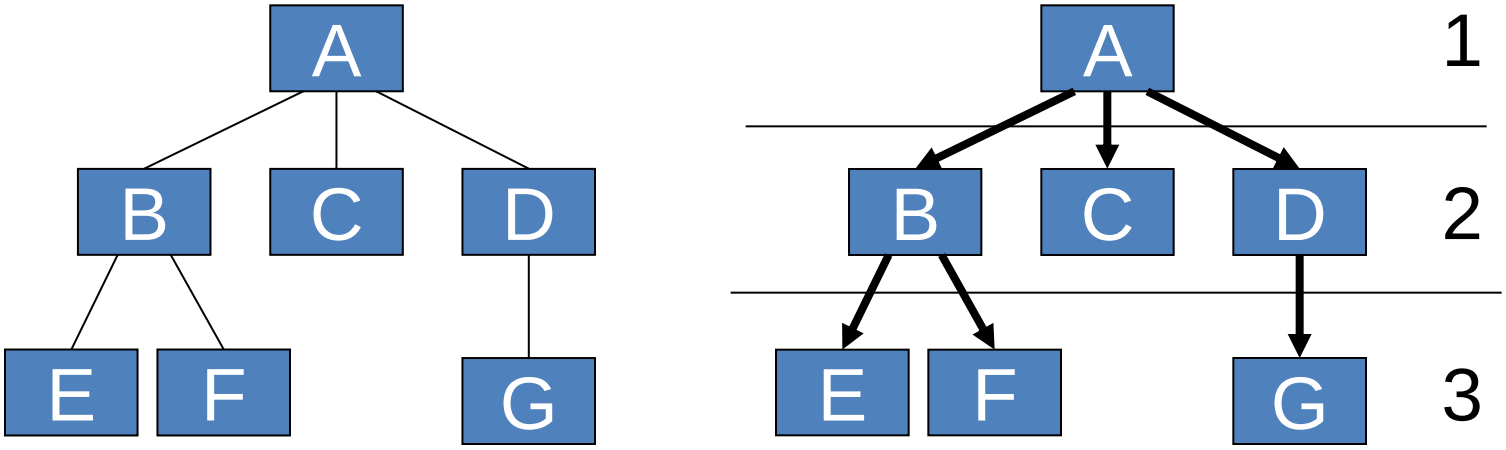
Testovanie


zber pož.
akceptácia
analýza
integrácia
návrh
test časti
implementácia

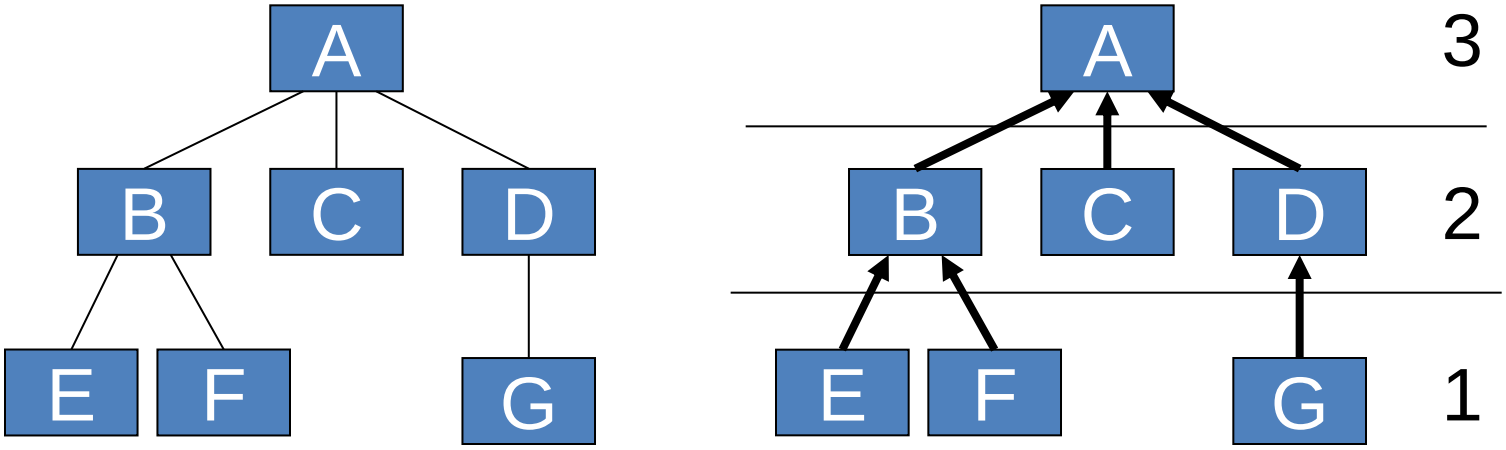
Stratégie testovania

- testovanie zhora nadol
- testovanie zdola nahor
- jednofázové testovanie
- sendvičové testovanie
- testovanie porovnávaním

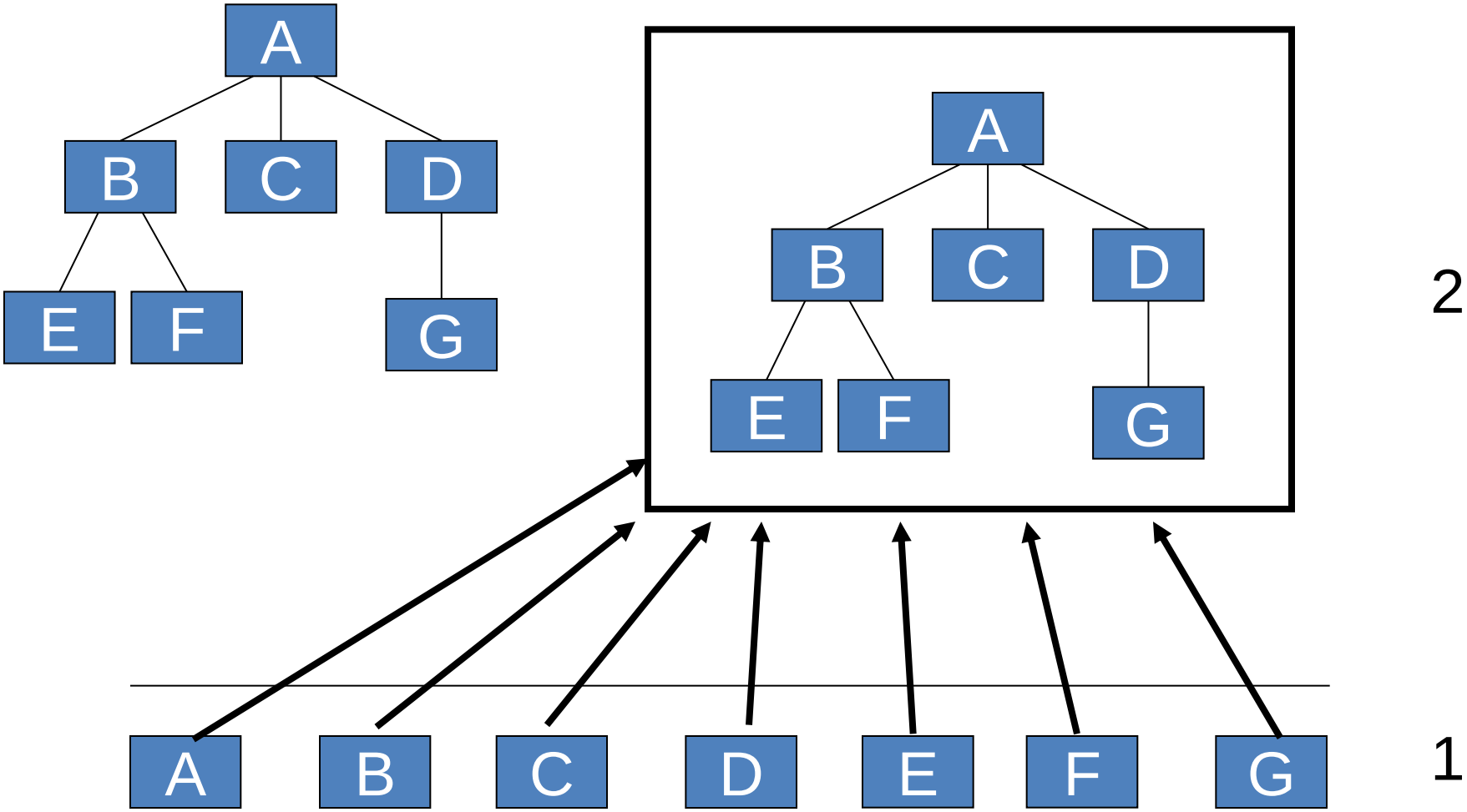
Testovanie zhora nadol



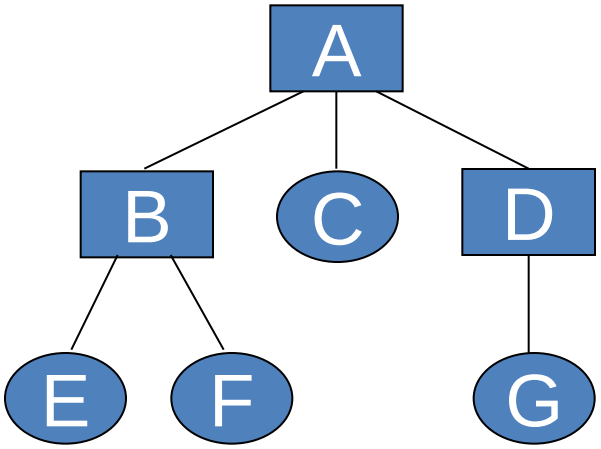
Testovanie zdola nahor

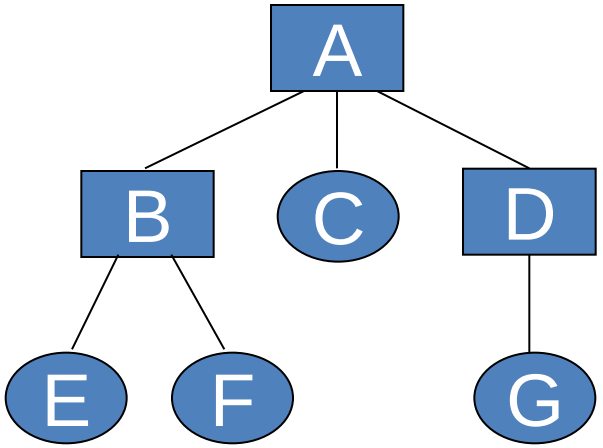


Testovanie jednofázové



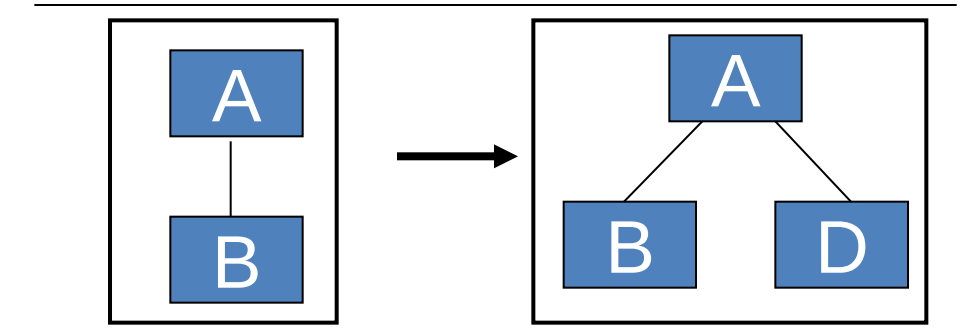
Testovanie sendvičové



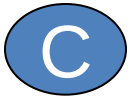


3

logický modul



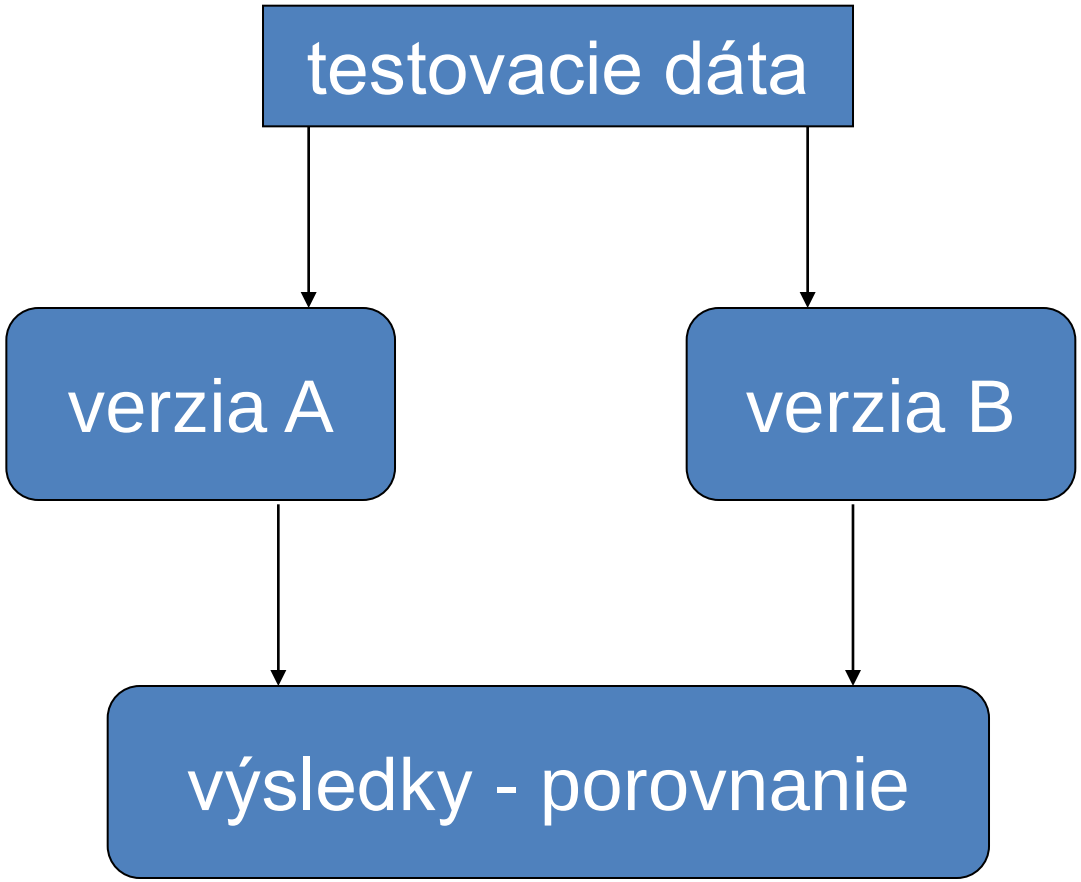
2



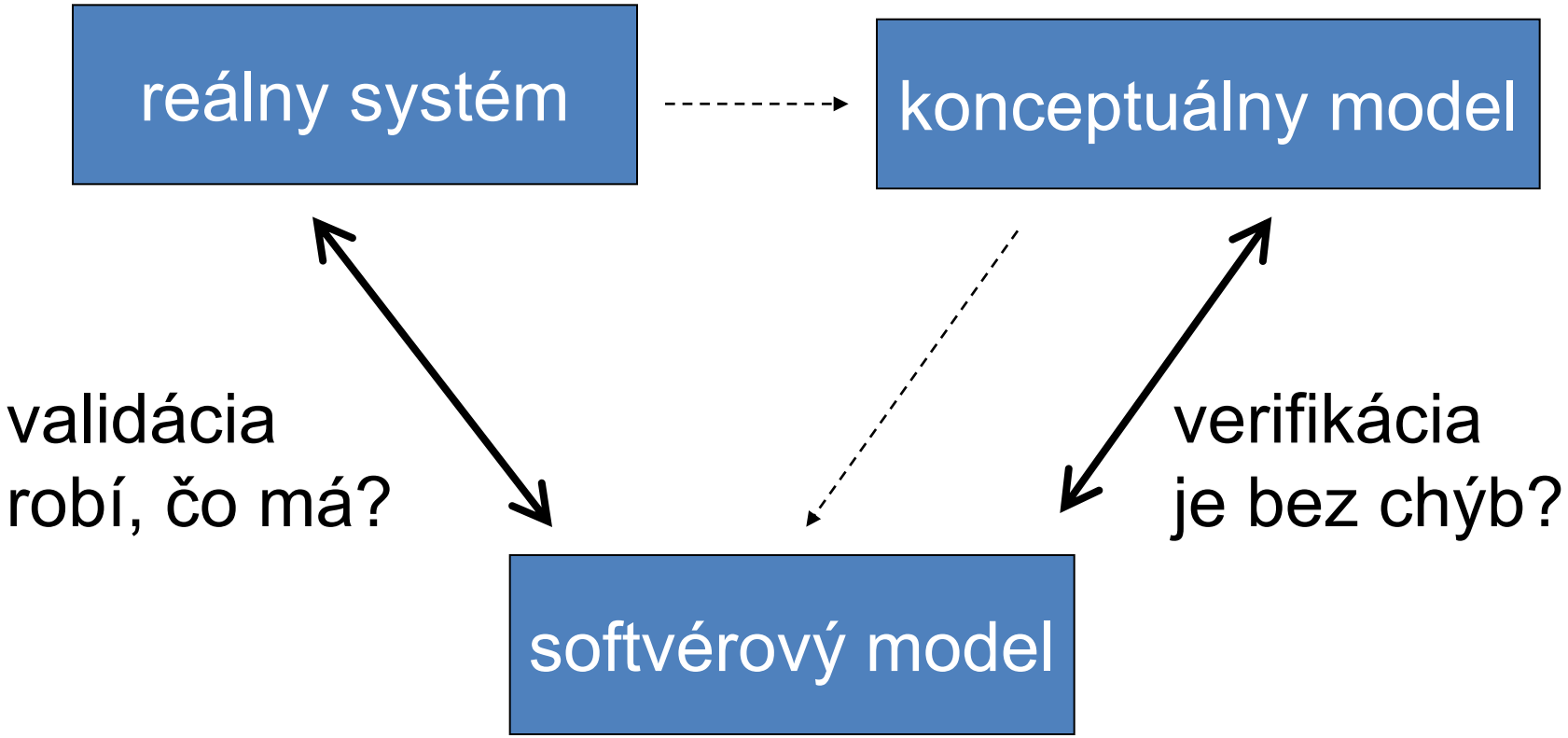
funkčný modul

1

Testovanie porovnávacie



Verifikácia - Validácia



Akceptácia

užívateľ testuje, či systém spĺňa zadanie

- správnosť – plní požiadavky
- robustnosť – odolný voči chybám
- výkonnosť – pamäťová a časová náročnosť
- dokumentácia – úplnosť a zrozumiteľnosť

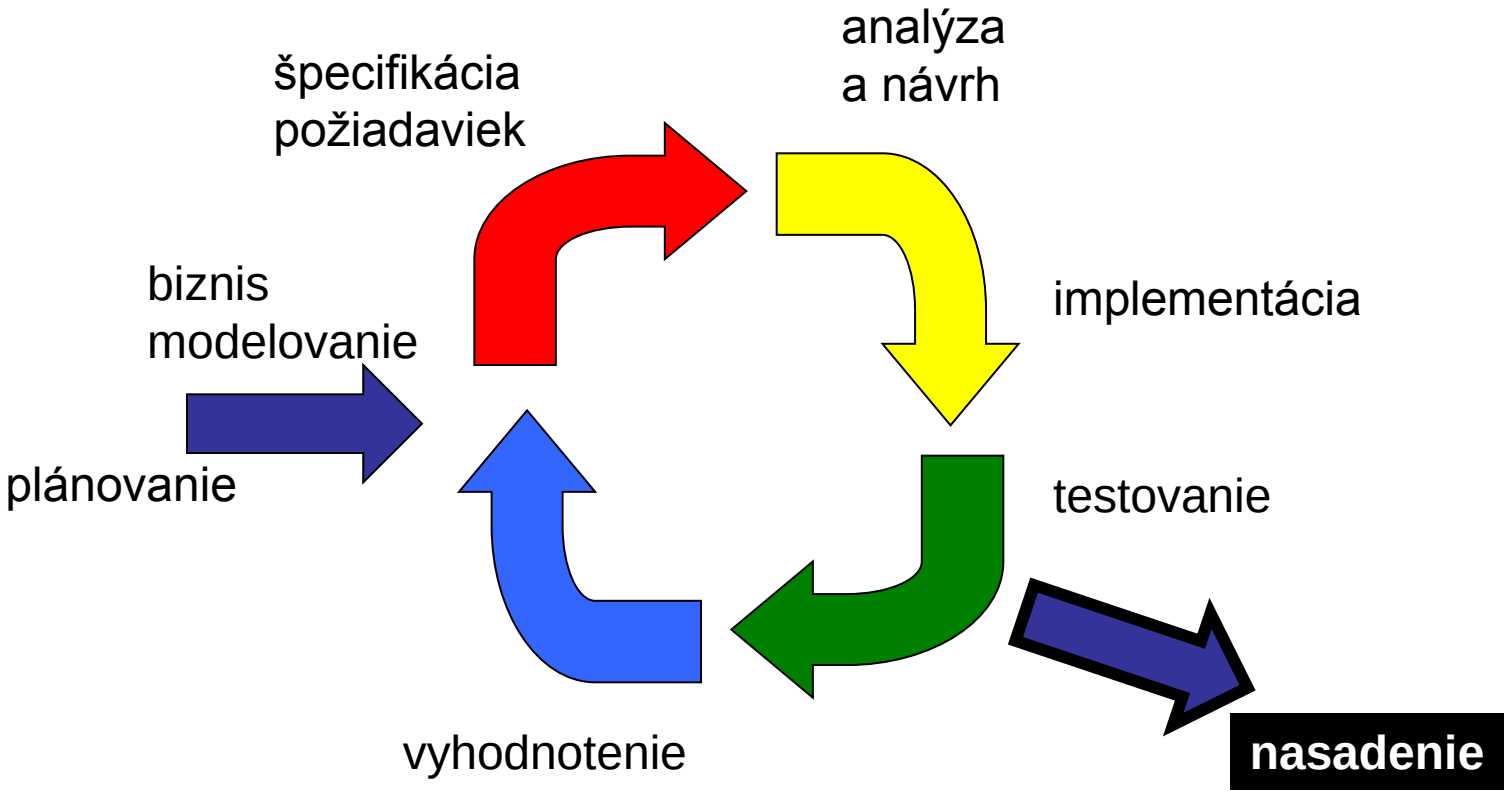
Testovanie alfa, beta

produkt na všeobecný predaj – nie je na objednávku

alfa testovanie
- vývojárska firma
- užívateľ sledovaný vývojármi
- známe prostredie nasadenia

beta testovanie
- užívatelia na svojich počítačoch
- rôznorodé prostredie – neznáme pre vývojárov
- správa od užívateľa > modifikácia > servis pack

RUP – iterácie



Nasadenie

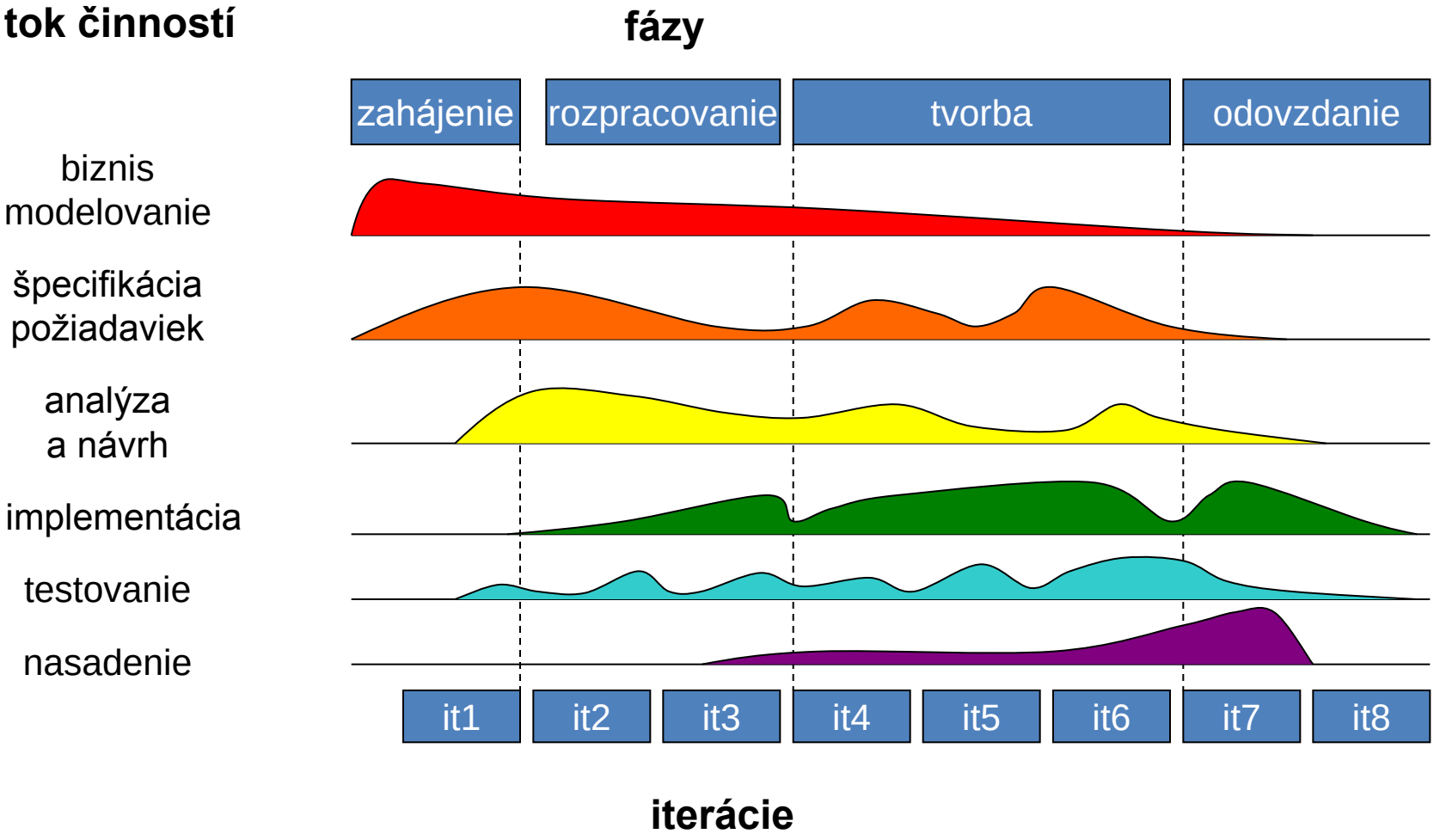
- inštalácia systému (u zákazníka)
- testovanie po inštalácii (beta)
- plán zálohovania a obnovy
- uvedenie do rutinnej prevádzky

Dokumentácia

- diagramy UML
- zdrojové texty programov
  - komentáre – nová a aktuálna informácia
  - identifikátory – seba dokumentujúce
- užívateľská príručka

CASE – generovanie dokumentácie

RUP – schéma (obsah x čas)



Ďakujem za pozornosť

Vaše otázky...