Testování SW 2

Y36SI3 - Realizace programových systémů

Ondřej Macek

Obsah přednášky

- Další pojmy
- White box testování
- Zátěžové testy
- Kontinuální integrace

Validace vs. Verifikace

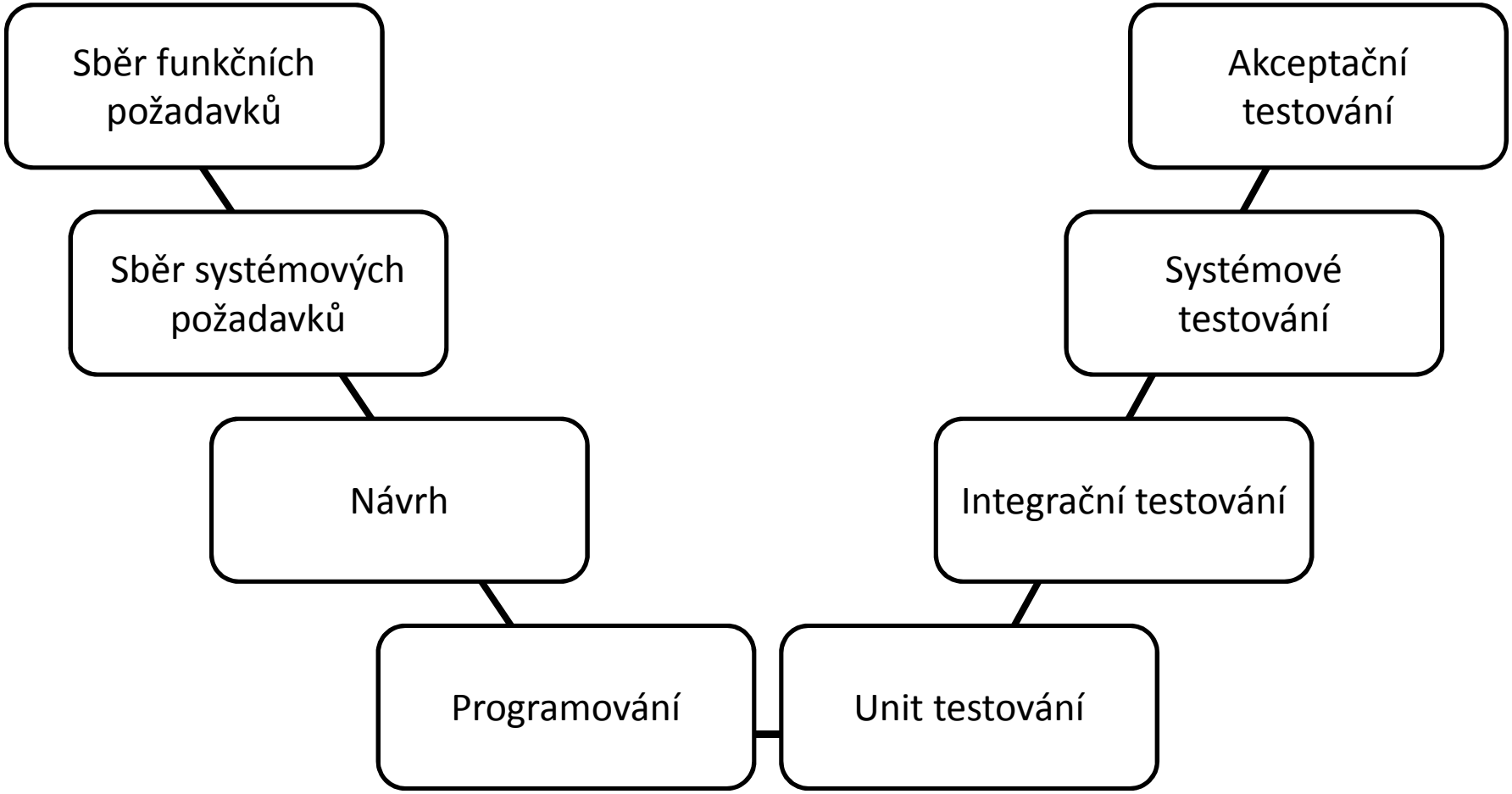
- Vytváříme správnou věc?

  (co děláme)

- Vytváříme tu věc správně?

  (jak děláme)

V-model


Sběr funkčních požadavků
Sběr systémových požadavků
Návrh
Programování
Unit testování
Integrační testování
Systémové testování
Akceptační testování

Logování

WHITE BOX TESTING

White-box testing

Základní myšlenka:

To co je implementováno je implementováno správně

- Je dostupná aplikace i kód
- Ověření toho, že všechny vnitřní komponenty správně fungují
    - Unit testing

Co je to unit?

Unit testing

(Testování aplikačních jednotek, Jednotkové testování)

- testujeme nejmenší části programu
- postupně postupujeme k větším celkům

metoda -> třída -> modul -> knihovna

Unit testing – základní pojmy

- Test harness
  - skript který automaticky volá testy
    - testovací metodu pozná podle názvu třídy, anotace

- Výsledek testu: pass/fail
  - assertTrue()
  - assertEquals()
  - fail()

Unit testing – základní pojmy 2

- Mock object
  - Jak testovat když není navazující třída hotová?

- Code Coverage
  - Počet otestovaných řádků
  - Branch coverage
  - Segment coverage
  - Condition testiting

Strategie Unit testování

- Data Flow Testing (DFT)
- Path Testing
- Loop Testing

Strategie Unit testování

- Arange - Act – Assert
- Testy se zaměřují na (business) funkčnost
- Vše by mělo být testovatelné nezávisle na souborovém systému, db, web servicies…

Co/jak unit testovat?

Nástroje unit testing

- JUnit
  - NetBeans (Tools > Create Junit Test)
- TestNG
  - inspirace z Junit, vylepšení
    - (anotace JDK5, rozdělení testů na různé počítače)
- xUnit frameworks
  - C++, C#, PHP, Ruby, Python
- dbUnit
- Code Coverage
  - http://emma.sourceforge.net
  - NetBeans - *http://codecoverage.netbeans.org*
  - Eclipse - *http://cobertura.sourceforge.net*

Test Driven Development

- Obrácení standardního přístupu k implementaci

- Test-first přístup

Integrační a Regresní testování

- Integrační testování – nezpůsobilo přidání nové unit do systému chyby někde jinde?

- Regresní testování – nezpůsobila úprava stávající funkčnosti chyby v systému?

TESTOVÁNÍ ZÁTĚŽE

Testování zátěže (Performance/Load)

Hlavní myšlenka:

1. Ověřit, že SW obstojí v běžném provozu
2. Zjistit, kdy to spadne

- Má smysl ho provádět až po dokončení SW

- Začít testovat na prázdné DB – co je pomalé na prázdné určitě nebude rychlé na plné

Plán testování zátěže

- Promyslet jak bude aplikace vytížená – nalezení špiček
- Aktivní vs. Připojení uživatelé

Aktivita	Požadovaný čas na odpověď	Počet aktivních uživatelů ve špičce

Response time

- Doba od zadání požadavku po přijetí odpovědi na něj.

- Co vlastně zahrnuje?

Zajímavost

Průzkum ukázal, že uživatel je soustředěn na systém zpracovávající jeho požadavek maximálně cca 8 vteřin, pak svou pozornost obrátí jinam

Reakce na výsledky zátěžových testů

- Typické řešení: koupit lepší železo
- Úprava kódu
- Varovná zpráva
- Omezení počtu transakcí

Nástroje testování zátěže

- Jmeter

  http://jakarta.apache.org/jmeter/

DALŠÍ TYPY TESTOVÁNÍ

Další testy

- Instalační testy
- Smoke test
- Usability test
- Interface test
- ….

KONTINUÁLNÍ INTEGRACE

Kontinuální integrace

- Jednotný repositář
  - Pravidelné commity
- Automatický build
  - Každý den ukončit buildem
- Automatické testy
  - Testy v kopii produkčního prostředí

Kontinuální integrace

- Snadný přístup k posledním spustitelným souborům

- Automatické nasazení

Nástroje kontinuální integrace

- Cruise control
  http://cruisecontrol.sourceforge.net/index.html
- Team Server
  http://www.jetbrains.com/teamcity/index.html
- Hudson
  http://hudson-ci.org/

Zajímavá literatura

Software Testing Help: ***WHITE BOX TESTING: NEED, SKILL REQUIRED AND LIMITATIONS***

http://www.softwaretestinghelp.com/white-box-testing/

[cit. 4. 11. 2010]

North D.: ***INTRODUCING BDD***

http://blog.dannorth.net/introducing-bdd/

[cit. 4. 11. 2010]

Fowler M.: ***CONTINUOUS INTEGRATION***

http://www.martinfowler.com/articles/continuousIntegration.html

[cit. 4. 11. 2010]