Testování SW

Y36SI3 - Realizace programových systémů

Ondřej Macek

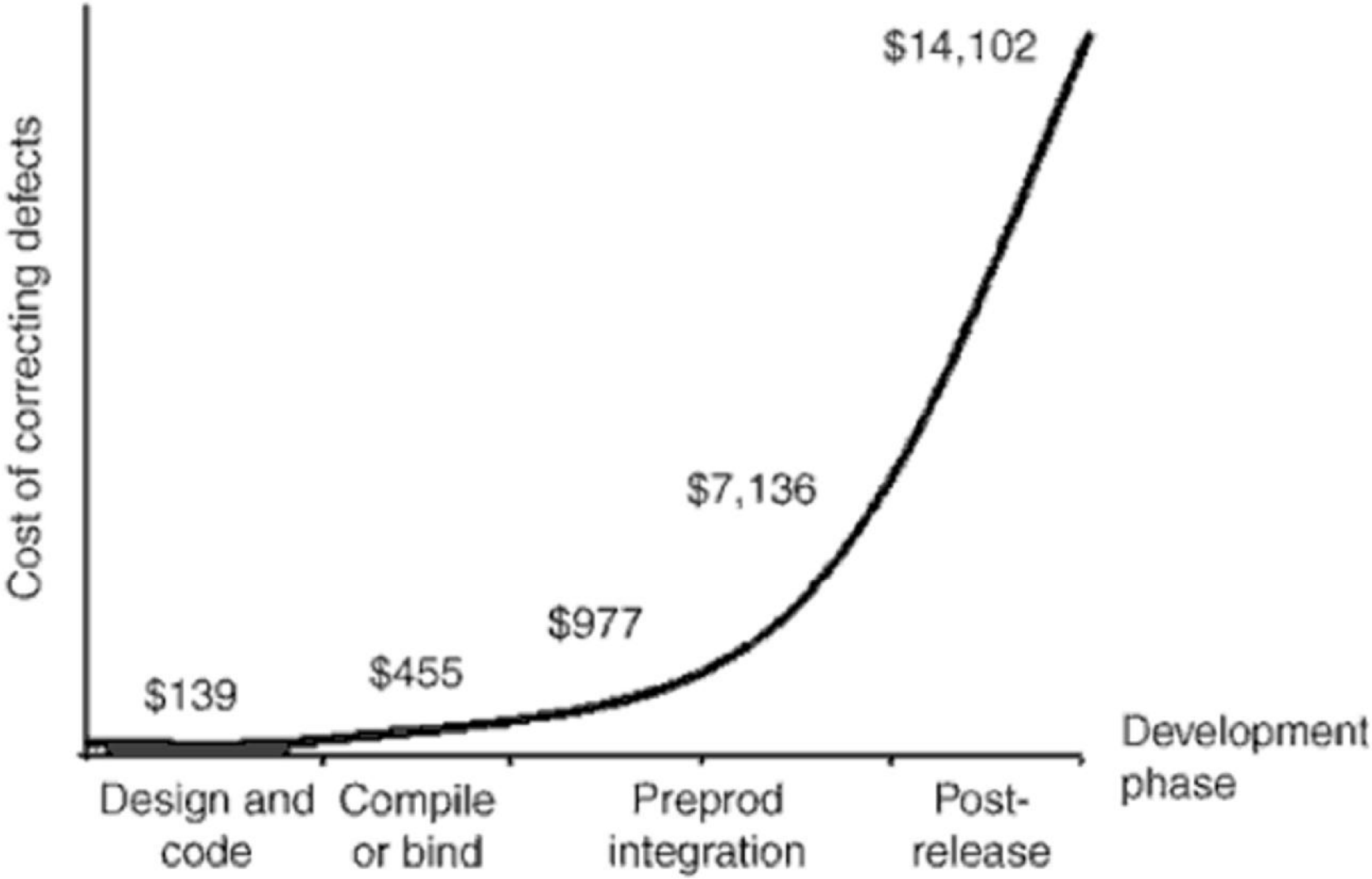
Obsah přednášky

- Testování
- Statické testování
- (Automatické) Black-box testování
- Monkey testing

Motivace k testování

- Chyby SW stojí USA $59,9 miliard
  - $22,2 miliardy by mohlo být odstraněno v testech
- Pád rakety Ariane 5
  - Testování odhalilo chybu (škoda, že až zpětně)
- WATTSEM HUMPHREY z Carnegie Mellon University Software Engineering Institute
  - většina z testovaných programových projektů obsahovala 10 chyb na 1 000 řádků. (V projektu s milionem řádků to je 10 000 chyb, což by prý vyžadovalo 50 lidí pracujících další rok, aby vše bylo (téměř) dokonale v pořádku)

Kolik stojí opravení chyby? (Motivace 2)



Co je to testování?

- Plánování testu
- Provedení testu
- Porozumění výsledkům testu
- Nápravná opatření

- Zavedení testování jako standardu

Charakter testu

- Specifikace (Proč)
- Promyšlenost (Jak, Čím)
- Opakovatelnost
- Odpovědnost (Kdo, Co)
- Ekonomická stránka (ROI)

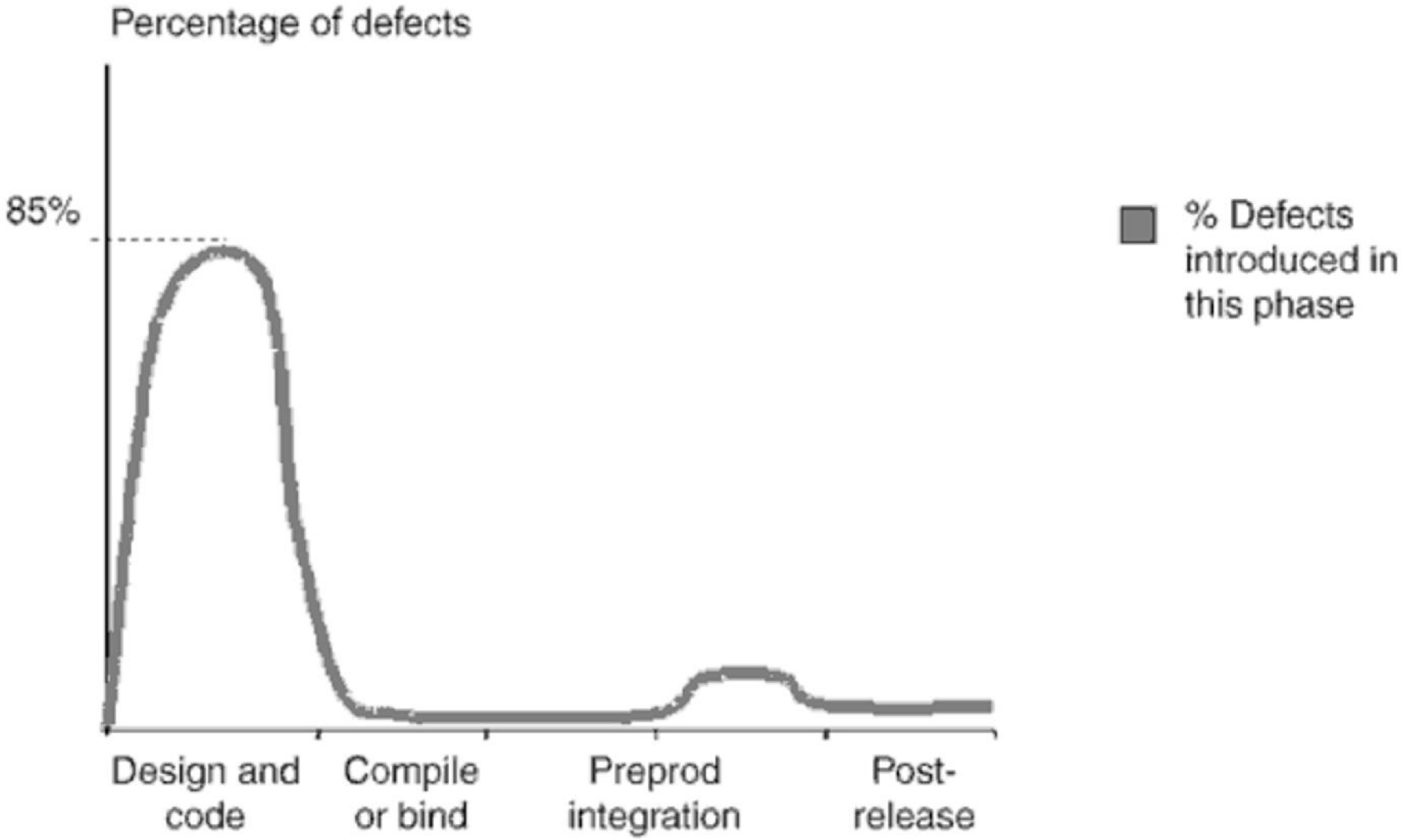
Kdo provádí testování?

- Uživatel
- Tvůrce
- Tester

Testerův charakter

- Technická způsobilost
- Tvůrčí myšlení
- Kritické myšlení
- Praktické myšlení

Kdy se chyby v SW berou?



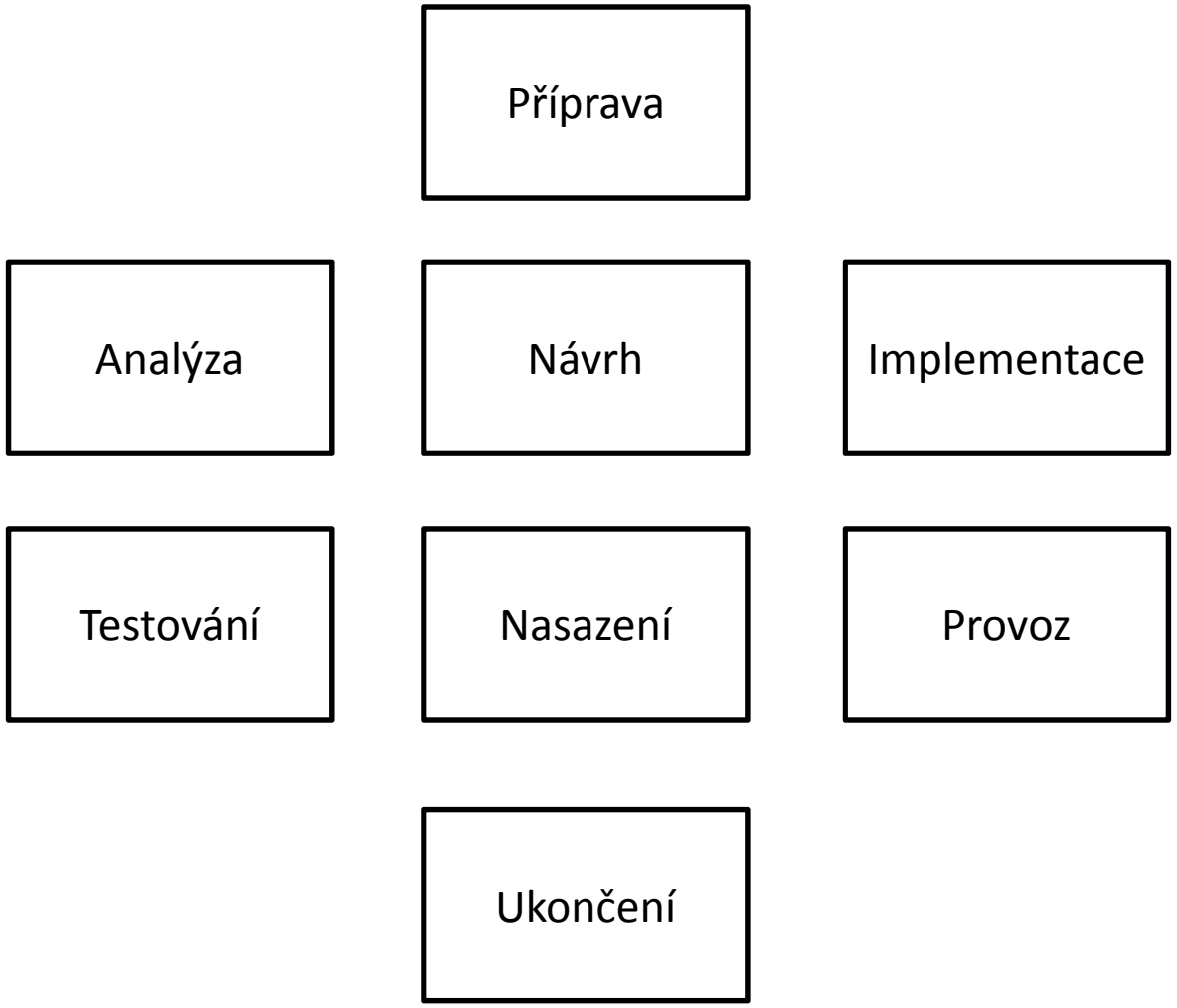
Jak určit oblasti testování?

- Určit největší rizika
- Určit nejkritičtější místa z hlediska business požadavků
- Určit co mi pomůže odhalit další chyby
- Zjistit jak bude systém používán
- ROI

Jak určit oblasti testování?

- Aplikační logika
- Aplikační bezpečnost
- Databáze
- (G)UI
- Prostředí aplikace
  - OS
  - Síť
  - SW/knihovny třetích stran

Kdy začít s testováním? A kdy s ním skončit?



Co kdy testovat?

Fáze vývoje	Testy
Analýza	1. Statické testy požadavků
Design	1. Statické testy všech návrhových dokumentů
Konstrukce	1. Statické testy • Kódu • Příruček 2. Testy funkčnosti 3. Zátěžové testy 4. Výkonnostní testy 5. Akceptační testy 6. Testy instalace

Checklist

- Seznam toho co jsem měl zkontrolovat
- Dá se přenášet z projektu do projektu
- Měl by se vyvíjet

STATICKÉ TESTOVÁNÍ

Statické testování

Statické testování

- Správnost
- Úplnost
- Přesnost
- Realizovatelnost
- Formální stránka

Předmět testování

- Analýza
- Návrh
- Kód
- Dokumentace

Statické testování

- 85% chyb vzniká v analýze
- It is good if it looks good
  - Čili „vjeřily by ste my?“
- Statické testování lze aplikovat i na kód
  - Dají se použít i různé statické analyzátory kódu

BLACKBOX TESTING

Black-box testing

- Základní myšlenka:

Chová se program tak jak má (jak bych čekal)?

- Testování správného provádění všech navržených funkcí produktu

- Není třeba znát kód, stačí fungující aplikace

Black-box testing

- Usability
- Zátěže
- Výkonu
- Alpha a Beta Testing

Black-box testing techniky

- Odhadnutí místa výskytu chyby
- Testy pomocí grafové analýzy (Graph Based Testing Methods)
- Analýza mezních hodnot (Boundary Value Analysis)
- Analýza pomocí tříd ekvivalence

Black-box testing

Klady

- Tester nemusí být technik.
- Ověří rozpory mezi systémem a specifikací
- Test cases mohou být vytvořeny hned jak je hotová funkční specifikace

Nevýhody

- Jsou třeba rozsáhlá testovací data (vstupy)
- Je těžké odhalit všechny důležité vstupy v omezeném čase
- Vysoká pravděpodobnost změny testovaného blackboxu během testování

AUTOMATICKÝ BLACK-BOX TESTING

Automatické testování uživatelského rozhraní (Black-box)

- Využívá nástrojů, které
  - zaznamenávají akce uživatele a dovedou je automaticky zopakovat
  - simulují chování uživatele

- Kontrola správnosti výsledku
  - porovnání se screenshotem obrazovky
  - něco sofistikovanějšího

Automatické testování uživatelského rozhraní

- Abbot + Costello
  (http://abbot.sourceforge.net/doc/overview.shtml)
  - Java
  - opensource
- IBM Rational Function Tester
  - fuzzy logika – vyrovná se s drobnými změnami UI
- Selenium
  (http://seleniumhq.org/)
  - pro webové aplikace (Firefox plugin)
  - akce uživatele zaznamenává jako JavaScript

MONKEY TESTING

Náhodné testování software

Pokud bude bilión opic bilión let mlátit do psacích strojů, napíšou něco od Shakespeara.

- Black Box testing s náhodným generováním uživatelského vstupu

- Monkey
  - nástroj pro generování náhodných vstupů z klávesnice a myši

Implementace „opice“

- Monkey
  - pracuje přímo s frontou zpráv operačního systému
  - XWindows - XSendEvent
  - Windows API - SendMessage

Náhodné testování software

- Dumb monkeys
  - zcela náhodný vstup
- Smart monkeys
  - stavové automaty
  - statistické modely

Užitečnost „dumb monkeys“

- Zdánlivě nejsou příliš zajímavé
- „Rachmaninoff testing“
- … ale ve skutečném nasazení na velkém projektu našly 10-20% procent chyb

Zlepšení

- Dumb monkey která rozumí základním událostem
  - pád aplikace
  - otevření nového okna
  - zavření okna

- Dumb monkey ovládá Smart monkey
  - náhodně kombinuje posloupnost rozumných testů
  - Hledá stavy které nejsou pokryté „Smart monkey“

Apple, 1983

- speciální flag MonkeyLives
  - na adrese $100

- pokud byl tento flag nastaven, aplikace schovala volbu Exit

Zajímavá literatura

NEJZBYTEČNĚJŠÍ CHYBY V PROGRAMÁTORSKÉ HISTORII - ROOT.CZ
http://www.root.cz/clanky/nejzbytecnejsi-chyby-v-programatorske-historii/
[cit. 28-10-2010]

BLACK BOX TESTING: TYPES AND TECHNIQUES OF BBT - SOFTWARE TESTING HELP
http://www.softwaretestinghelp.com/black-box-testing/ [cit. 28-10-2010]

G. D. Everett, R. McLeod: *SOFTWARE TESTING: TESTING ACROSS THE ENTIRE SOFTWARE DEVELOPMENT LIFE CYCLE.* John Wiley & Sons, Inc., Hoboken, New Jersey. 2007