

Software testing

Tomáš Krátký

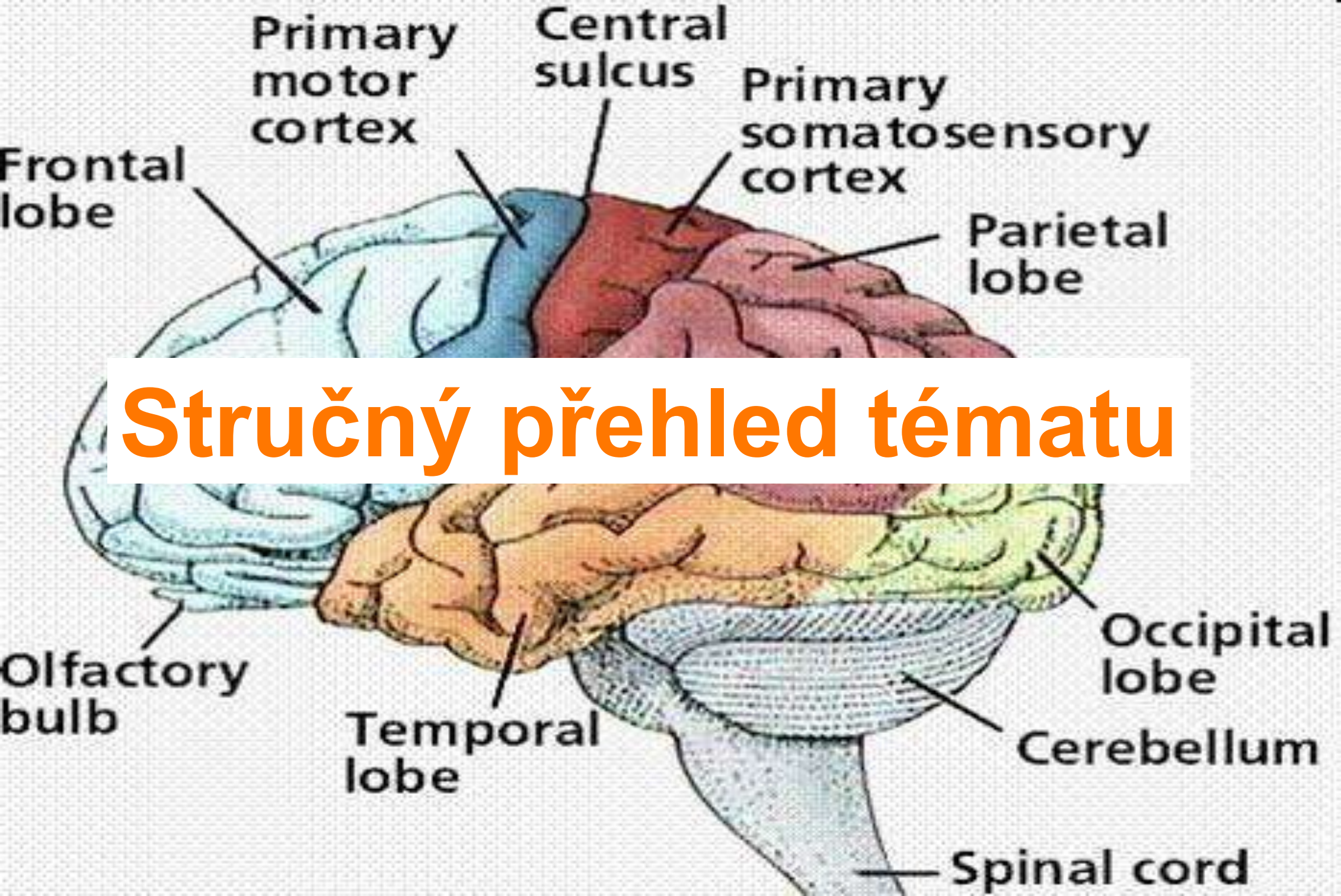
tomas.kratky@profinit.eu





Obsah

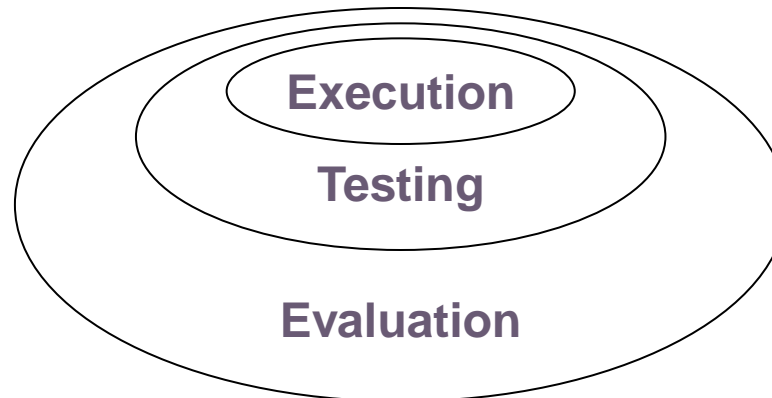
- Stručný přehled tématu
- Začínáme testovat
- Základní techniky
- Plánování testů
- Návrh testovacích případů
- Provádění testů
- Automatizace v kontextu testování
- Goodies





Co je Software testing ?

- Zkoušení / simulace provozu SW
- Ustanovení důvěry v to, že SW dělá co má, a nedělá, co nemá
- Analýza SW s cílem nalézt chyby a problémy
- Měření funkcionality a kvality SW
- Zhodnocení atributů a schopností SW, zda dosahují požadovaných či akceptovatelných výsledků
- Inspekce, stejně jako provádění testů kódu





Trocha historie

Testing objectives

- Pomoci jasně popsat chování systému
- Nalézt defekty v požadavcích, designu, dokumentaci a kódu jak nejdříve je to možné





Základní pojmy

Test plan (plán testů)

- Definuje strategii testů, vždy obsahuje Test coverage (**co** testovat, co netestovat), Test methods & tools (**jak** testovat a pomocí jakých nástrojů), Test responsibilities (**odpovědnosti**)

Test case (testovací případ)

- Množina podmínek, za kterých tester určí, zda aplikace či systém funguje korektně či nikoliv

Test oracle

- Mechanismus pro určení, zda SW prošel nebo neprošel určitým testem (požadavek, regres, heuristika, ...)



Základní pojmy

Test script

- Množina instrukcí (kroků), které budou provedeny na testovaném systému s cílem zjistit, zda systém funguje, jak je očekáváno

Test data (testovací data)

- Data speciálně identifikovaná pro využití v rámci testovacího případu

Test report (výsledky testu)

- Výsledek jednoho či více testů obsahující minimálně identifikaci testu a jednoznačný výsledek společně s komentářem, je-li třeba



Základní principy

- Kompletní testování není možné
- Práce testerů je kreativní a náročná
- Testování je „řízeno“ riziky
- Analýza, plánování a návrh jsou důležité
- Motivace je důležitá
- Čas a zdroje jsou důležité
- Časování přípravy testů hraje velkou roli
- Měření a sledování „pokrytí“ je důležité



Typologie testů

- Unit testy
- Integrovní testy
- Systémové testy
 - Funkční
 - Výkonové
 - Bezpečnostní
 - ...
- Akceptační testy
 - Uživatelské akceptační testy
 - Operační testy
 - ...





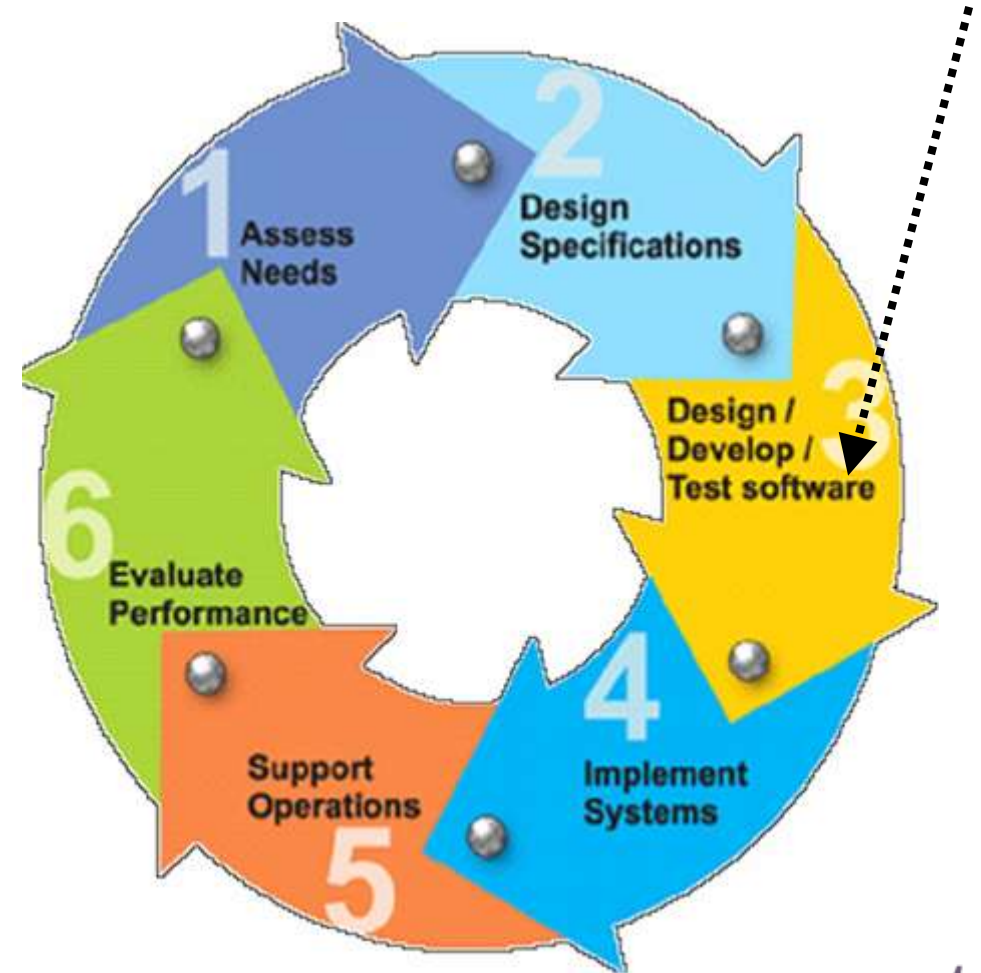
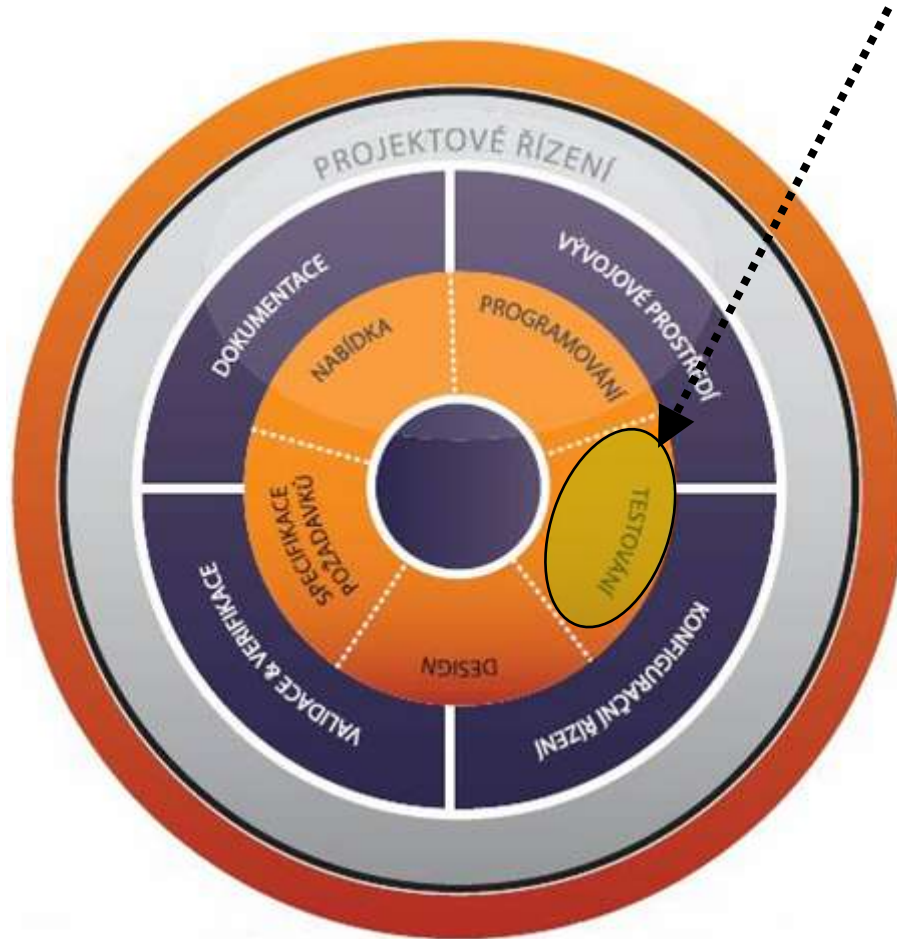
Typologie testů

??? Regresní testy ???

??? Kvalifikační testy ???

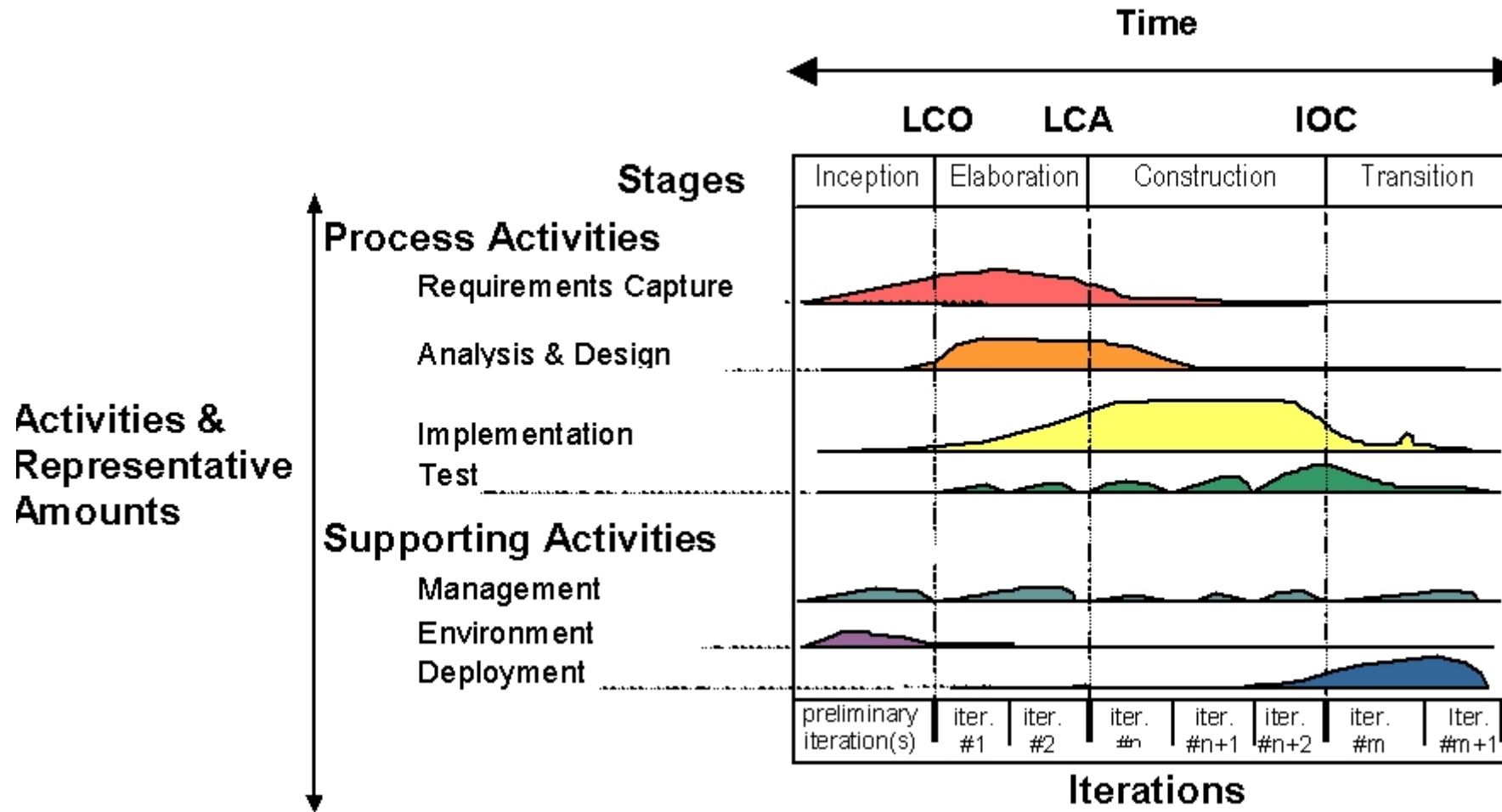


Softwarový proces



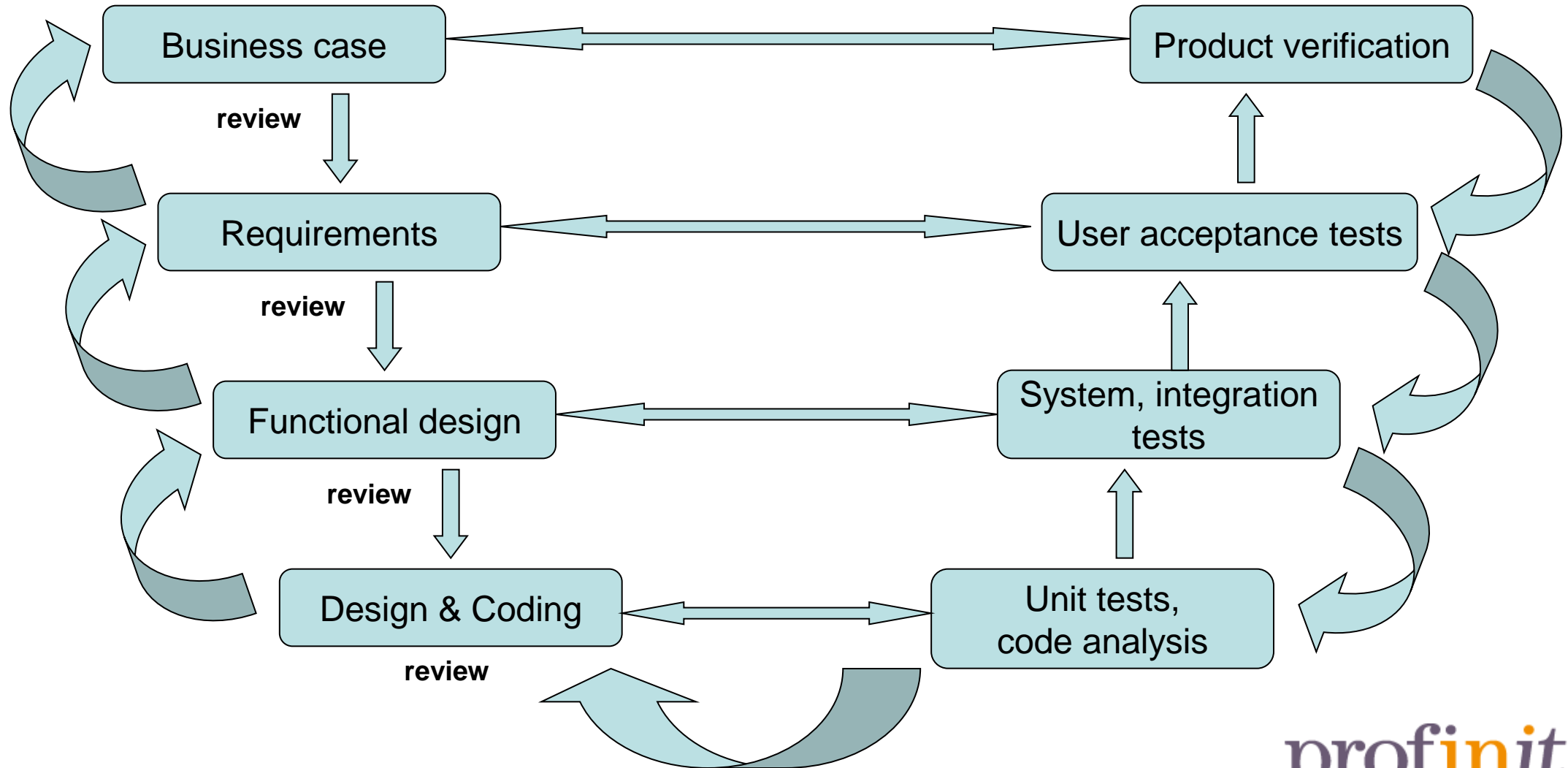


Softwarový proces



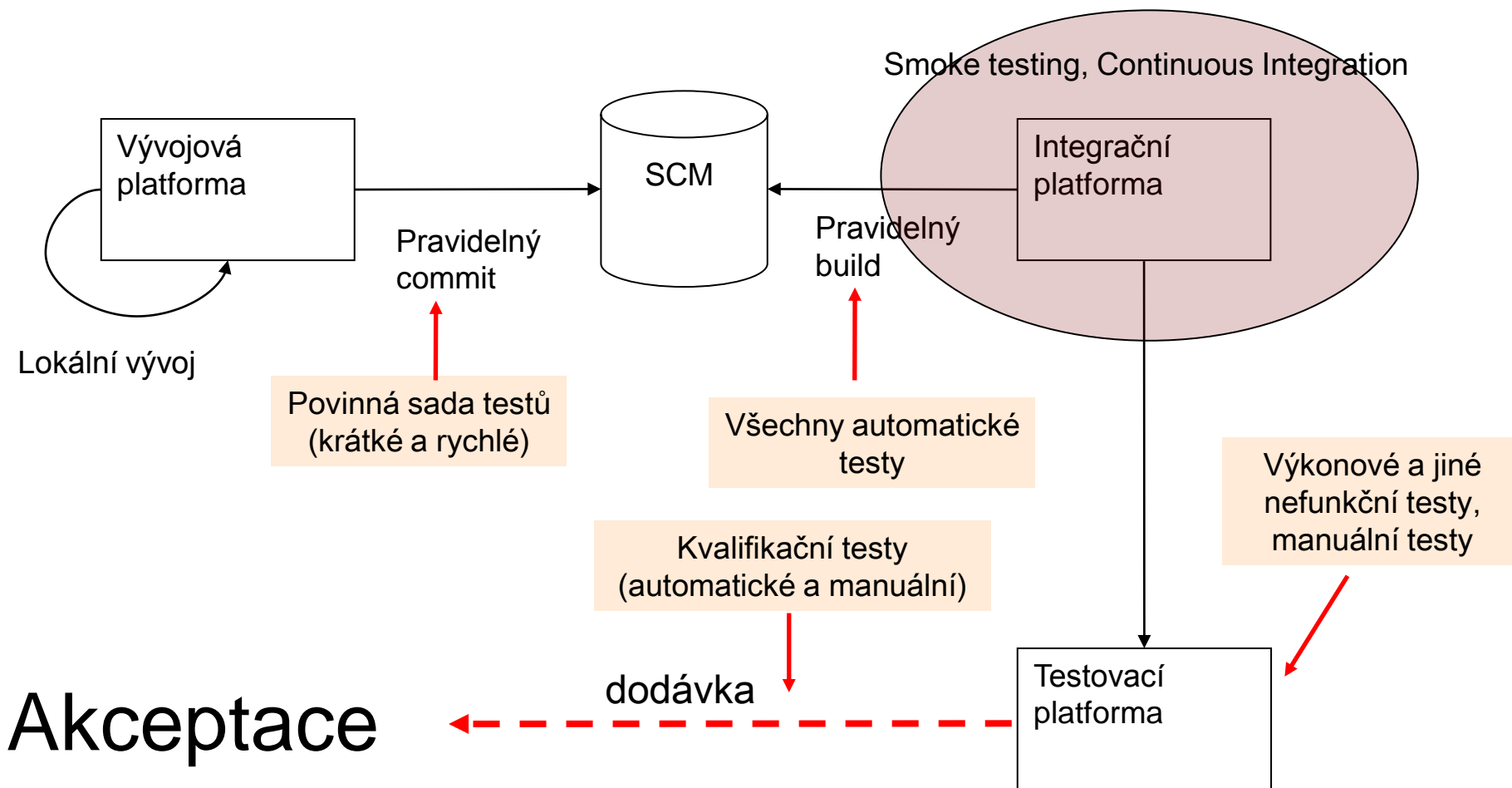


Načasování testů aneb „V – model“





Testy na projektu





SWEBOK

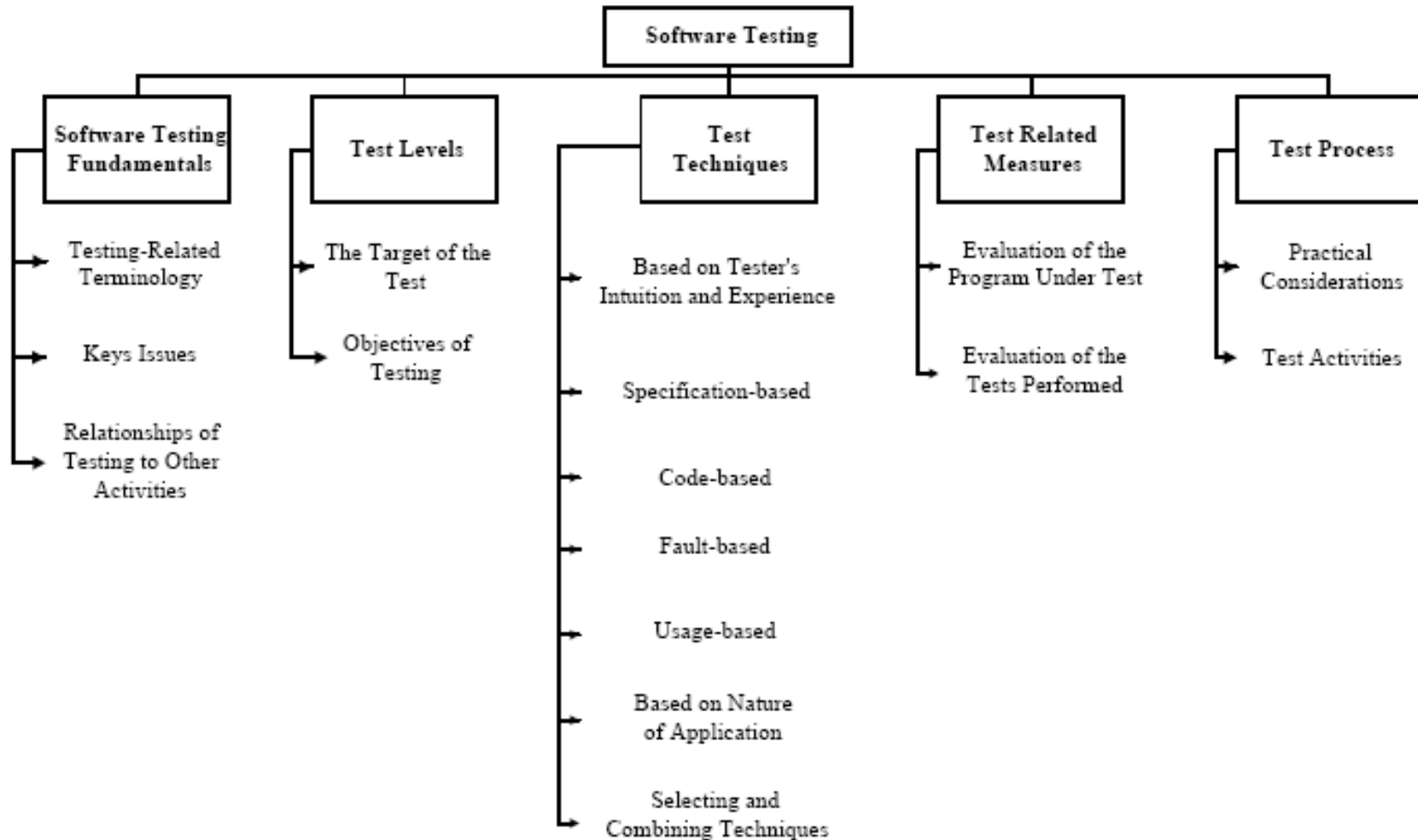


Figure 1 Breakdown of topics for the Software Testing KA

A photograph of a rocket launch at night. The rocket is a tall, slender cylinder with a red nose cone, positioned vertically in the center. A bright, intense white and yellow flame extends from the base, surrounded by a massive, billowing cloud of orange and yellow smoke and fire. The launch is taking place on a launch pad, with various support structures visible on the left and right sides. The background is a dark, clear night sky.

Začínáme testovat ...



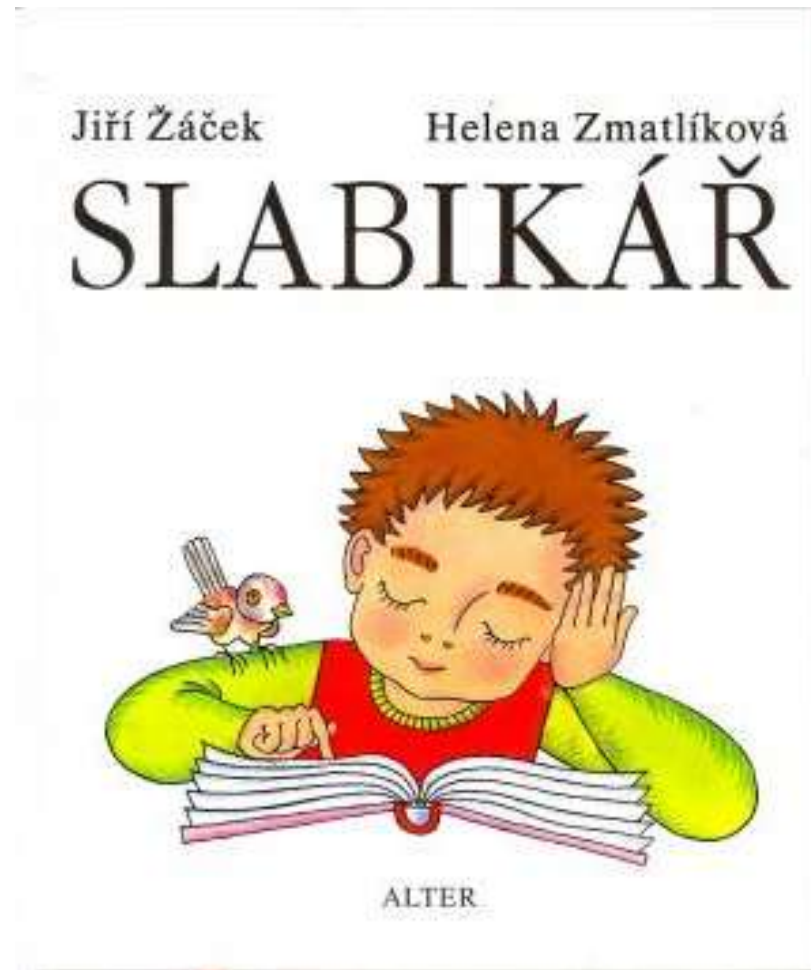
Jak (jednoduše) na testování

Základní strategie

- **Začít se základními testy**
Testovat pomocí hodnot, které by měly projít, jinak to bude vážný problém.
- **Testovat nejdříve do šířky spíše než do hloubky**
Nejdříve zkontrolovat zběžně všechny části než se zaměříme na detaily.
- **Pak se zaměřit na mocnější, komplexnější testy**
Jak program přežije základní otestování, je možné „přitvrdit“.
- **Vybrat vhodné hraniční podmínky**
Testů je příliš mnoho, je nutné vybrat vhodnou strategii pro jejich výběr.
- **Dělat exploratory testing**
Každý týden se snažit provést nějaké nové testy.



Základy základů





Boundary and Equivalence Analysis

- Testů je příliš mnoho → **Partitioning** (rozdělení testů do tříd ekvivalence)
- Provádění více testů ze stejné skupiny je **redundantní**
- Volba **nejvhodnějšího reprezentanta** skupiny (ten s max. pravděpodobností odhalení chyby)
- **Hraniční testovací případy** jsou obvykle velmi mocné a identifikují často chyby





Black box vs. White box

Black box

- Testujeme oproti „rozhraní“
- Nezajímá nás implementace
- Robustnější (není nutné často upravovat)



White box

- Strukturální testy
- Přihlížíme k implementaci
- Křehčí (změna implementace je rozbije)
- „path“ testing





Pozitivní vs. negativní

- Vždy testovat, že
 - **funguje, co fungovat má**
 - **nefunguje, co fungovat nemá**



- Neomezovat se na „přípustné“ hodnoty, operace, ...
- Vždy zkoušet, jak se SW chová v případě „nepřípustných“ hodnot, operací, ...



Základní techniky





Techniky testování

Testovací techniky / paradigmata

- Definuje typy testů, které jsou relevantní a zajímavé
 - Vytváří určitý způsob myšlení a přístup k testování
 - Implicitně určuje limity co je relevantní, zajímavé nebo možné
- Existuje velké množství technik, cca 150
- Překrývají se



Techniky testování

Jak je využíváme ke tvorbě testů ?

- Analýza situace
- Modelování testovacího prostoru
- Volba pokrytí
- Konfigurace testovacího systému
- Provoz testovacího systému
- Pozorování testovacího systému
- Zhodnocení výsledků testu

Testovací
technika
je **recept** pro
provádění těchto
činností s cílem
objevit něco,
co stojí
za reporting.



Jak vybrat vhodnou techniku

Záleží na několika aspektech

- Požadavky na testy
- Atributy testů
- Kontext vývoje
 - Elementy produktu
 - Kritéria kvality
 - Rizika
 - Omezení projektu





Jak vybrat vhodnou techniku

Požadavky na testy

- Funkce stejně důležité
- Chyby stejně závažné

Varianta I

	Nalezeno před releasem
Funkce A	100
Funkce B	0
Funkce C	0
Funkce D	0
Celkem	100

Která varianta je lepší ?

Varianta II

Funkce A	50
Funkce B	8
Funkce C	8
Funkce D	8
Celkem	74



Jak vybrat vhodnou techniku

Varianta I

	Nalezeno před releasem	Nalezeno později	Celkem
Funkce A	100	0	100
Funkce B	0	16	16
Funkce C	0	16	16
Funkce D	0	16	16
Celkem	100	48	148

Varianta II

Funkce A	50	50	100
Funkce B	8	8	16
Funkce C	8	8	16
Funkce D	8	8	16
Celkem	74	74	148



Jak vybrat vhodnou techniku

Požadavky na testy

- Najít důležité bugy, aby byly odstraněny
- Pomoci udělat ship / no-ship rozhodnutí
- Ověřit interoperabilitu s jiným produktem
- Minimalizovat náklady na technickou podporu
- Ověřit shodu se specifikací
- Změřit kvalitu

...



Jak vybrat vhodnou techniku

Atributy testů

- **Power** – vysoká pravděpodobnost nalezení problému, pokud existuje
- **Valid** – odhalí skutečné chyby
- **Value** – odhalí chyby důležité pro uživatele
- **Credible** – odpovídá očekávanému chování uživatele
- **Representative** – odpovídá tomu, čeho si uživatel nejpravděpodobněji všimne
- **Non-redundant** – reprezentuje větší skupinu testů, které se zaměřují na stejné riziko



Jak vybrat vhodnou techniku

Atributy testů

- **Motivating** – „klient“ bude chtít chyby nalezené testem opravit
- **Performable** – proveditelný v souladu s návrhem
- **Maintainable** – udržitelný při změnách systému
- **Repeatable** – snadno a levně znovupoužitelný
- **Pop** (Karl Popper) – odhalí věci týkající se našich základních či kritických předpokladů
- **Coverage** – vyzkouší systém způsobem, kterým to nečiní jiné testy



Jak vybrat vhodnou techniku

Atributy testů

- **Easy to evaluate** – snadné a jasné vyhodnocení
- **Supports troubleshooting** – poskytuje užitečné informace pro vývojáře, který ladí nalezené problémy
- **Appropriately complex** – dostatečná komplexnost
- **Accountable** – obhajitelnost, prokazatelnost testu
- **Cost** – přímé náklady, čas a pracnost
- **Opportunity cost** – náklady, které se ušetří provedením testu



Dominantní „techniky“

- Function
- Specification-based
- Domain
- Risk-based
- Scenario
- Stress
- User
- High volume automated
- Exploratory
- Regression

Regresní testování není technika sama o sobě, jde o využití testů vytvořených dle jiných technik, zde explicitně vytaženo pro svou důležitost ...



Plánování testů





Plánování testů

Jak neuspět

- Zapomenout, že testují lidé
- Předstírat, že testeré jsou odpovědní za kvalitu, nikoli management
- Diktovat datum spuštění bez ohledu na reálná omezení projektu
- Hodnotit testerů podle počtu nelezených chyb
- Nedostatek vzdělávání pro testery
- Oddělit vývoj a testování



Plánování testů

Plán testů

- Kolik času vyhradit na testování ?
- Co všechno má být v plánu testů ?
 - Test coverage (co testovat)
 - Test methods and tools (jak testovat)
 - Test responsibilities (odpovědnosti)

The background of the slide is a composite image. It features architectural blueprints with various lines and grids. In the lower-left corner, a portion of a laptop keyboard is visible. A black pen lies diagonally across the lower-right section of the image. The overall color palette is cool, with blues and greys dominating the scene.

Návrh testovacích případů



Návrh testovacích případů

Jak neuspět

- Malá diverzita použitých technik
 - Pouze specification based testing
 - Pouze function testing
- Příliš detailní testovací skripty
 - Malá volnost pro kreativitu testera
 - Malý prostor pro „náhodu“
 - Obtížná udržitelnost
- Exploratory testing bez patřičného vzdělávání
- Oddělení návrhu a provádění testů
- Ignorování existujících rizik

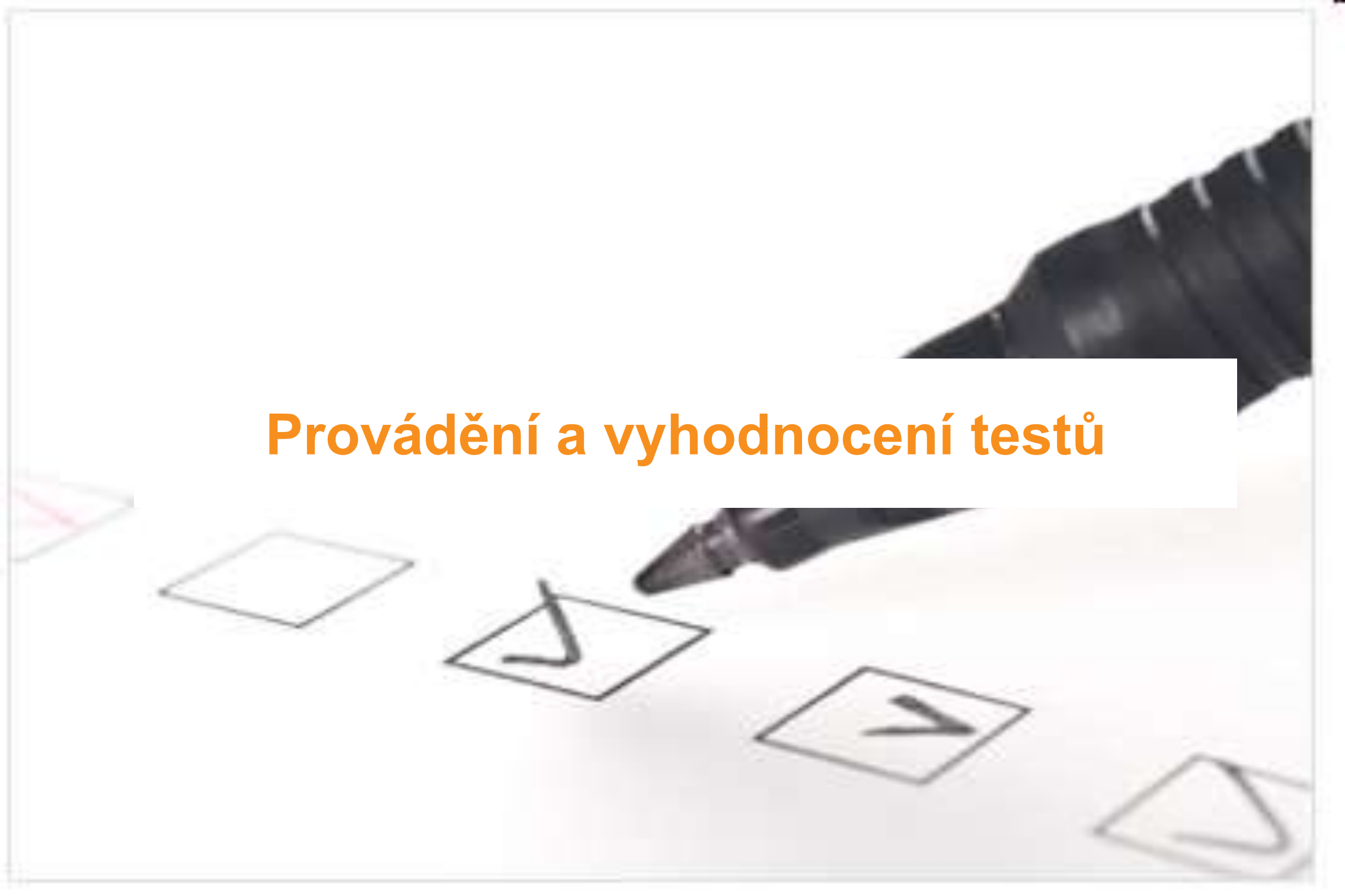


Návrh testovacích případů

Jak má vypadat testovací případ ?

- Viz techniky testování
- Viz atributy testu
- Míra detailu
- Testovací data
- Standardní struktura

Provádění a vyhodnocení testů





Provádění testů

Kdy začít testovat ?

- Plánovaný harmonogram vs. realita
 - zpoždění dodavatele
 - čekat nebo začít dříve ?
- Dobré časování je zásadní
 - příliš **pozdě** → problém se splněním termínů, málo času na testování
 - příliš **brzo** → nestabilní SW, zbytečně vynaložený čas a práce testerů

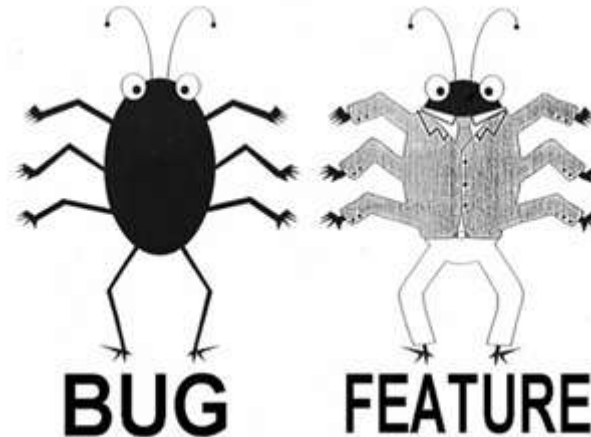




Trouble ticketing, bug reporting

Základní pravidla

- Evidence **všech** nalezených **issues**
- Jediné místo pravdy
- Nejde jen o to nahlásit issue, důležité je udělat to tak, aby bylo možné jej nasimulovat a **opravit**
- Schopnost odlišit chyby které „proklouznou“
- Trouble ticketing → Bug tracking ?





Reportování výsledků testů

- Standardizovaný test report
 - Celkové zhodnocení testovaného SW
 - Dopad nedostatků na projekt, systém, ...
 - Detailní výsledky
 - Nalezené problémy
 - Odchyłky od testovacích případů
 - Log testů (provedené testy, průběh testů, ...)



Řízení průběhu testů



Řízení průběhu testů

- Alokace zdrojů
- Dynamické přidělování a plánování práce
- Reakce na problémy
- Zlepšování procesu testování
- Snaha optimalizovat

...

→ Musíme vědět „**jak na tom jsme**“ !



Řízení průběhu testů

Kdo to ví ?

Jak byste odpověděli na otázku
„Kolik testování jste na projektu už udělali ?“

Jak a podle čeho měřit rozsah testování ?

Co je to rozsah testování ?



Řízení průběhu testů

Co je to rozsah testování ?

- Typicky odpovědi založené na
 - **Produkt:** „Otestovali jsme 70% řádek kódu“
 - **Plán:** „Provedli jsme 65% testovacích případů“
 - **Výsledky:** „Našli jsme 753 chyb“
 - **Pracnost:** „Pracovali jsme na tom po 3 měsíce 60 hodin týdně, provedli jsme 8956 testů“
 - **Kvalita testování:** „Beta testéři našli 28 chyb, které nám unikly, naše regresní testy se zdají neefektivní“



Řízení průběhu testů

Co je to rozsah testování ?

- Typicky odpovědi založené na
 - **Rizika:** „Dostáváme spoustu stížností od Beta testerů, stále máme otevřených přes 500 problémů, produkt nebude do 3 dnů připraven ke spuštění“
 - **Projektová historie:** „V tento moment jsme na předchozích projektech měli 12% nalezených problémů stále otevřených, stejné by to mělo být i teď.“



Řízení průběhu testů

Měření rozsahu testování

- Žádná metrika dokonalá
- Řešením z praxe je ... ?
 - ... kombinace více různých metrik ...
 - pokrytí, pracnost, výsledky, rizika, potíže, ...

A large crowd of white, featureless humanoid figures in a dark, industrial setting. The figures are arranged in rows, with one figure prominently in the foreground, looking directly at the camera. The background is dark and industrial, with a grid-like pattern of light and shadow.

Automatizace v kontextu testování



Automatizace exekuce testů

- Snaha automatizovat testy již mnoho desítek let
- Proč ?
 - Opakovatelnost a konzistence testů → stejné vstupy a podmínky nezávisle na počtu opakování, odpadá problém s motivací lidí k opakování stejných testů
 - Praktická znovupoužitelnost testů → lze opakovat stejný test v různých prostředích, v různých konfiguracích, s mírně modifikovanými vstupními daty, ... a znovuspuštění testu je levné
 - Praktické baseline testy → automatizace umožňuje spustit velmi „hutnou“ sadu testů, umožňují efektivně provádět regresní testování



Automatizace regresních testů

- Velice častý scénář
- Typický průběh automatizace
 - Vytvořit testovací případ
 - Manuálně jej spustit a ověřit výstup
 - V případě selhání nahlásit chybu
 - V případě úspěchu „uložit“ výsledek
 - Opakovaně spouštět test a výsledky porovnávat s uloženými, hlásit chybové situace
 - Udržovat automatický test



Automatizace regresních testů

Je to skutečně automatizace ?

- Analýza programu
- Design testu
- První spuštění testu
- Uložení výsledků
- Dokumentace testu
- Znovuspuštění testu
- Vyhodnocení výsledků
- Údržba testu

Co z toho vlastně
dělá stroj ?



Automatizace regresních testů

Ne pro automatizaci ...

- Tvorba testovacích případů je drahá
- Vyžaduje velmi technicky zkušené členy týmu
- Vyžaduje dobře definované a stabilní rozhraní
- Vyplácí se pozdě, výhody automatizace v release N se vrací až v release N+1
- Regresní testy mají často menší Power než nové testy



Automatizace v kontextu testování

Kdy může mít smysl ? Vždy otázka ROI !

- Smoke testing (**Continuous Integration**)
- Configuration testing (HW SW compatibility)
- **Variace** (viz. debata o Stochastic testing)
- Stress testing
- Load testing
- **Příprava testovacího prostředí** (data, ...)
- ...



Nástroje pro automatizaci testů

Unit a integrační testy

- [jUnit](#), [TestNG](#), [jMock](#), [EasyMock](#), [DbUnit](#), ...

Statická analýza kódu

- [Findbugs](#), [PMD](#), [JDepend](#), FoxCop, ...

Funkční testy – tlustý klient

- [Selenium](#), HP QuickTest, IBM Functional Tester, ...

Funkční testy – tenký klient

- HP QuickTest, IBM Functional Tester, White, Autolt, ..

Výkonové testy

- [JMeter](#), [Dieseltest](#), [QALoad](#), ...

Komplexní řešení

- HP Test Suite, IBM/Rational Test Suite, ...

Příprava testovacího prostředí

- IBM Optim, Grid-Tools DataMaker, Oracle Datamasking, ...





Goodies





Templates, checklists, literatura

Podpora univerzit

Univerzitní výuka

Materiály SwEng

Předmět SW1129

Předmět 1-INF-312

Předmět A4M33SEP


Předmět 1-INF-314

Předmět PDBT

Invited lectures

Odborné konference

ACM SPY



TESTING MATERIÁLY

Články

- **What Is Software Testing? And Why Is It So Hard?**
- **Little Book of Testing - Volume I** - jasná a srozumitelná, obsahuje mnoho nahuštěných informací a dobrých rad pro každého, kdo přichází s testováním do kontaktu
- **Little Book of Testing - Volume II** - pokračování prvního dílu, obsahuje další pokročilá témata
- **Software Test Planning and Management Guide** - průvodce tvorbou a managementem plánu testů vytvořený ve Space and Naval Warfare Systems Center San Diego

Checklists

- **CxCheck TestPlan.pdf** - checklist firmy Construx vhodný pro vytváření plánu testů

Templates

- **CxTemp TestPlan.doc** - template firmy Construx pro plán testů
- **IEEE TestPlanTemplate.pdf**
- **MIL-STD-498 SwTestDescriptionTemplate.doc**
- **MIL-STD-498 SwTestPlanTemplate.doc**
- **MIL-STD-498 SwTestReportTemplate.doc**
- **SPAWAR SwTestPlanTemplate.doc**
- **SPAWAR SwTestReportTemplate.doc**

Všechny odkazované materiály jsou poskytnuty výhradně za účelem výuky softwarového inženýrství.
© Of Respective Parties 2007-2009

Diskuse





Backup slides — dominantní techniky



Dominantní techniky

Function testing

- Test každé funkce / feature v izolaci
- Použití „opatrných“ (middle-of-the-road) hodnot
- Neočekáváme selhání
- Později „přitvrdíme“
- *Highly credible a Easy to evaluate* testy
- Opomíjí interakce a průzkum přínosů programu



Dominantní techniky

Specification-based testing

- Test SW proti každému tvrzení v referenční dokumentaci (specifikace, uživatelský manuál, ...)
- Závisí na kvalitě referenční dokumentace
 - Nutná revize
 - **Neúplnost, nejednoznačnost, netestovatelnost**
- *Traceability matrix* (pokrytí „tvrzení“ testy)
- *Highly significant (motivating)* testy
- Hledá špatně specifikované oblasti



Dominantní techniky

Domain testing

- Testy proměnných (vstupy, konfigur. hodnoty, ...)
- Proměnné v izolaci i ve skupinách
- Všechny přípustné i nepřípustné hodnoty
- Partitioning, třídy ekvivalence, „nejlepší zástupce“
- Pozor na chyby mimo hraniční případy
- *High power, Lot of Information value* testy
- Nižší *Credible a Motivating* (corner cases)



Dominantní techniky

Risk-based testing

- Představit si možné způsoby, jak může program selhat, a navrhnout jeden či více testů, které ověří, zda skutečně daným způsobem selže.
- Snaha najít „**velké problémy**“ co nejdřív
- **Optimalizace** priorit dle míry rizika
- „Kompletní“ množina risk-based testů založena na úplném seznamu rizik (věcí, které mohou jít špatně)
- Nebezpečí **špatné identifikace** rizik
- *Highly credible, highly motivating* testy
- Potenciál mít *high information value*



Dominantní techniky

Risk-based testing – kde hledat problémy

- Nesplnění některých atributů kvality
- Nové funkce, technologie
- Věci dělané na poslední chvíli
- Unavení, problémoví programátoři
- Nejasnosti (ve specifikaci, ...)
- Komplexní funkce
- Historicky chybové funkce
- ...



Dominantní techniky

Risk-based testing – kde hledat problémy

- Hlášené problémy ostatních
 - <http://www.bugnet.com>
 - <http://www.cnet.com>
 - odkazy na <http://www.winfiles.com>
 - některé mailing listy a další zdroje ...



Dominantní techniky

Scenario testing

- Testy jako komplexní „příběhy“, které odpovídají způsobu použití programu v reálných situacích
- Nutno dodržet následující atributy
 - *Complex* (mnoho funkcí)
 - *Credible, Motivating* (odpovídá reálnému použití)
 - *Easy to evaluate* (žádná nejasnost, zda test proběhl či selhal)
- *High power* testy
- Jedná chybná funkce zhatí vše
- Nebezpečí nedostatečného pokrytí
- Inspirace
 - dle produktů konkurence
 - způsoby chování zákazníka
- Varianta „*harsher*“ testy (*killer soap opera* 😊)



Dominantní techniky

Regression testing

- Vytváření testů s důrazem na jejich pravidelné opakování po provedení změn v programu
- Libovolný test lze použít jako regresní
- Zpočátku *Power* a *Credible* testy, jejich síla klesá s časem a počtem jejich opakování (pokud se nedějí v systému velké změny)
- Je nutné se naučit navrhovat testovací případy s důrazem na znovupoužitelnost



Dominantní techniky

Stress testing

- Několik definic
 - Zatížit program extrémní aktivitou a sledovat, zda selže
 - Testování za hranicemi specifikovaných požadavků na komponentu či program s cílem způsobit selhání systému
 - Zahnání aplikace nestandardními podmínkami do stavu selhání s cílem sledovat **JAK** program selže a jaká slabost se tímto odhalí
- *High power* testy
- Někdy ne tolik *Credible* a *Motivating* testy (netypické pro uživatele)
- Problémy s **diagnostikou** v případě selhání



Dominantní techniky

User testing

- Testování, které dělají **skuteční** uživatelé
 - Ne testeři předstírající uživatele
 - Ne někdo předstírající, že je tester, předstírající uživatele
- Navrhovat testy mohou testeři
- Libovolný test lze použít jako User test
- Důležité je nechat uživateli dostatek „prostoru“
- *Credible a Motivating* testy



Dominantní techniky

High volume automated testing

- Random / Stochastic testing
- Strukturu testu navrhuje člověk, jednotlivé testovací případy vyvíjí, spouští a interpretuje stroj, který následně upozorňuje na selhání
- Nutná maximální (kompletní) míra automatizace
- Jednotlivé testy někdy slabé, málo *Credible* a *Motivating*, není to „řemeslná práce“
- Nutno zabudovat troubleshooting



Dominantní techniky

Exploratory testing

- Libovolné testování takové, že tester **aktivně** kontroluje návrh testu **během** jejich **provádění** a využívá informace nabyté během testování k návrhu nových a lepších testů
- Očekávané výsledky ani konkrétní podoba testů není definovaná, záleží na uvážení testera
- Používání mnoha stylů, podle toho, který je zrovna nejvhodnější pro splnění cíle
- Nutný velmi **zkušený tester** se znalostí produktu, domény, testovacích technik, ...
- Vhodné pro **párové** testování