

NÁSKOK
DÍKY
ZNALOSTEM

PROFINIT

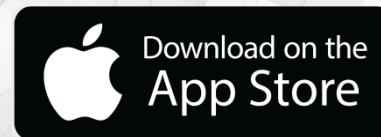
Quality assurance a testování

Bohumír Zoubek, Otakar Ertl

17. ledna 2018



Dotazy na
<https://www.sli.do>
event #W485



Definice pojmů



Definice pojmů

Kvalita

- › „*The totality of features and characteristics of a product or service that bear on its ability to meet stated or implied needs.*“ (ISO 8402-1986)
- › Souhrn vlastností nebo charakteristik produktu či služby, které souvisí s jeho či její schopností splnit explicitně uvedené či implicitně předpokládané potřeby.
- › V podstatě to znamená mít **spokojeného zákazníka/uživatele**

Quality assurance

- › Množina aktivit, jejichž cílem je **zajistit kvalitu** produktu či služby **systematickým** a **věrohodným** způsobem.
- › QA nedokáže na 100% zajistit tvorbu kvalitního software, výrazně však dokáže zvýšit pravděpodobnost, že se tak stane.

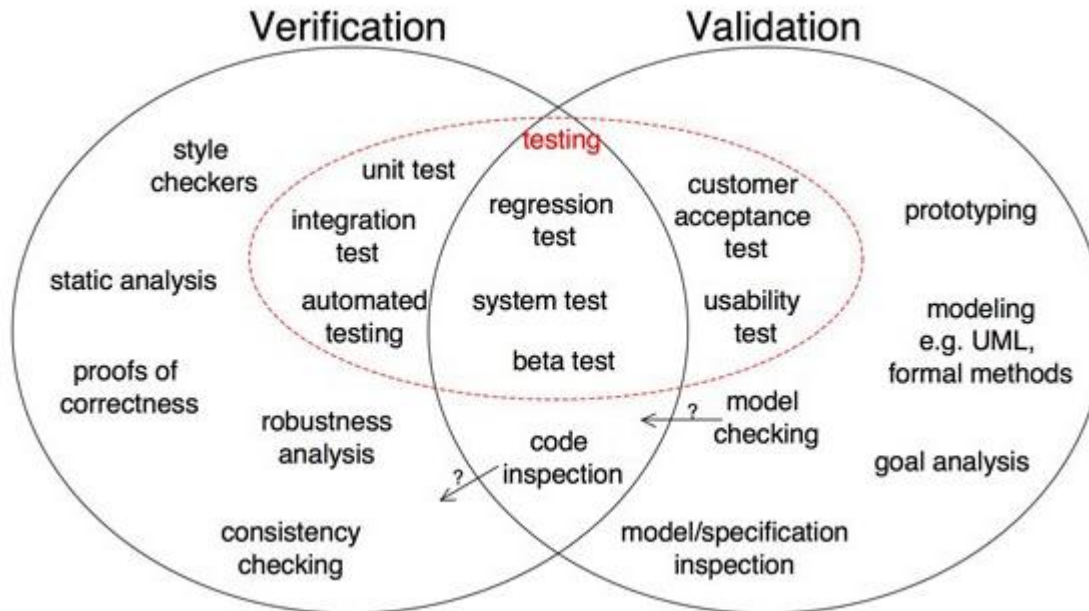
Testování

- › Množina aktivit s cílem **změřit kvalitu** vytvářeného software.

Definice pojmů

Validace, Verifikace

- › Proces/množina aktivit s cílem **zjistit**, zda určitý artefakt **splňuje nároky** na něj kladené
- › Validace → end-to-end verifikace dle specifikace
- › Validace → *building the right thing ...*
- › Verifikace → *building it right ...*



- › <http://www.easterbrook.ca/steve/2010/11/the-difference-between-verification-and-validation/>

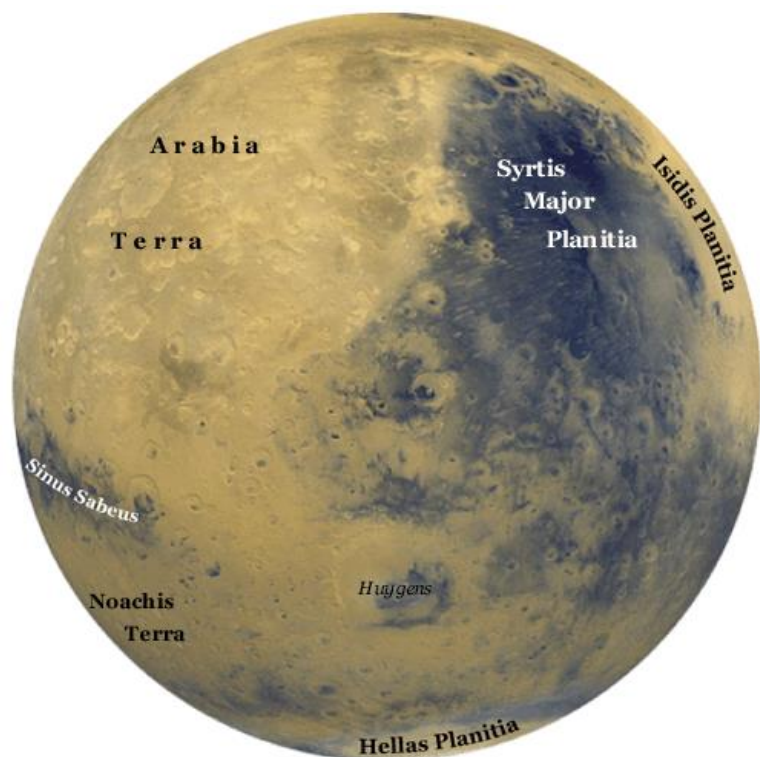
Definice pojmů – shrnutí

QA je o zajištění kvality !

**V&V je o zjišťování
(měření) kvality !**

**Testování je jedna
z forem V&V !**

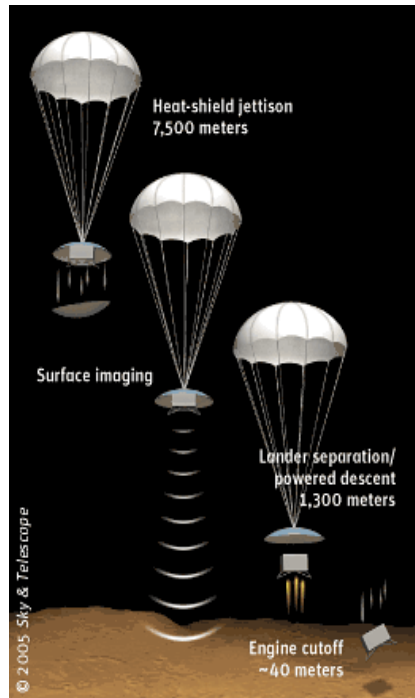
Proč kvalita ? – krátký kvíz



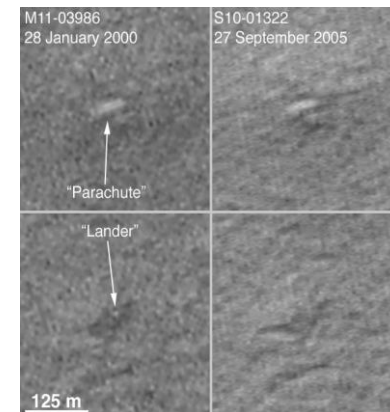
- › Druhá nejmenší
- › Má dva měsíce
- › Čtvrtá planeta sluneční soustavy

Mars Polar Lander

- › 3.1.1999 Mys Canaveral
- › 3.12.1999 vstup do atmosféry



- 40 metrů na povrchem vypnuty motory
- Volný pád
- Víc se neví ...
- Falešný signál od jedné nohy vyhodnocen jako informace o tom, že modul přistál.
- Chyba identifikována na 1 řádku kódu
- **Cena mise 327,6 mil. USD (celý Mars Surveyor '98)**



Víc?

Pokud Vám to nestačí

- › Mars Climate Orbiter (MCO) – metric/imperial (náklady viz MPL)
- › Ariane 5 – 64 floating point → 16 bit signed integer (7 billion USD/10 let vývoje)
- › Procesor Pentium – chybný algoritmus dělení
- › ...

Trocha čtení

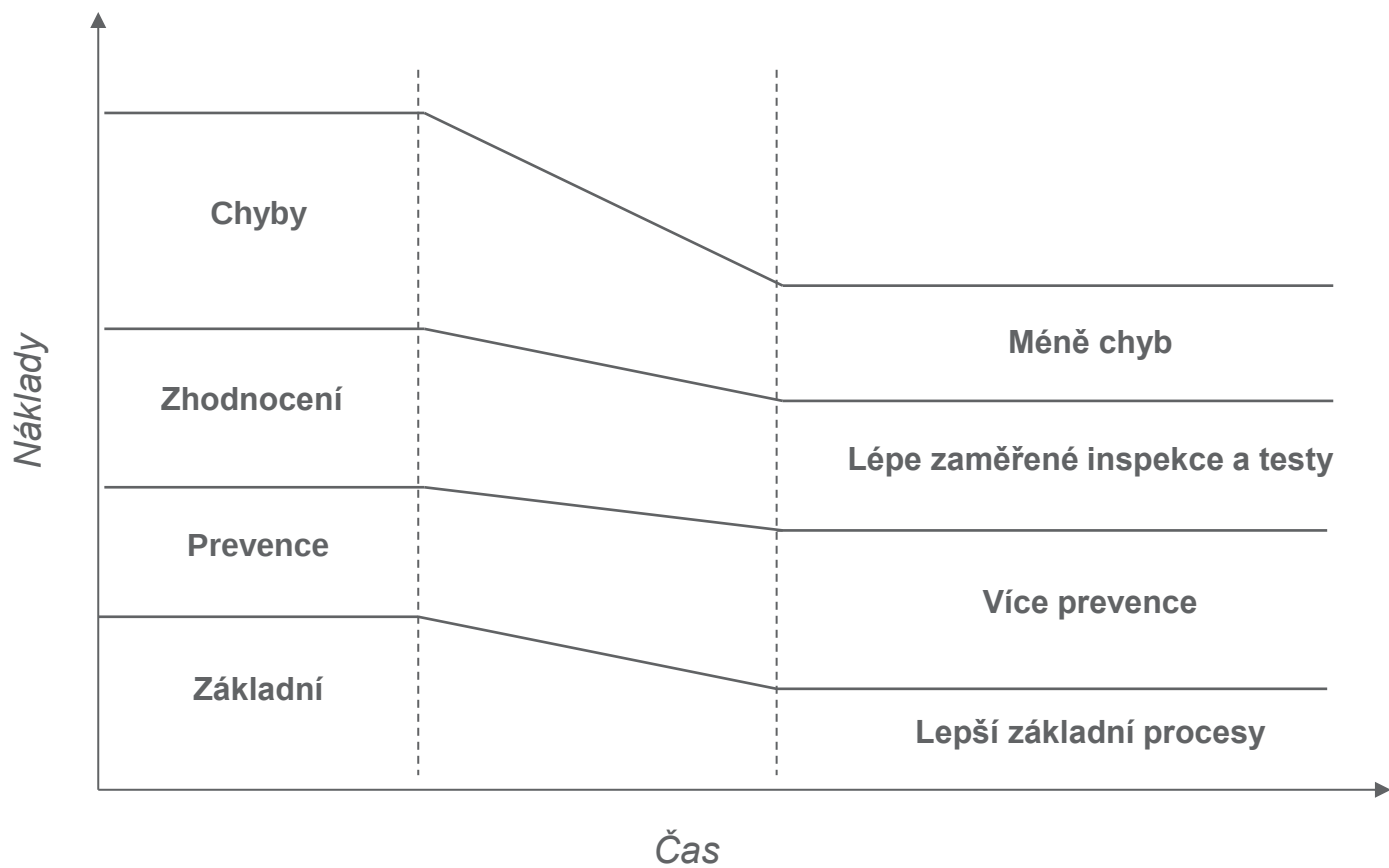
- › <http://nssdc.gsfc.nasa.gov/nmc/spacecraftDisplay.do?id=1998-073A>
- › ftp://ftp.hq.nasa.gov/pub/pao/reports/1999/MCO_report.pdf
- › <http://www.ima.umn.edu/~arnold/disasters/ariane.html>
- › <http://www.around.com/ariane.html>
- › <http://www5.in.tum.de/~huckle/bugse.html>

Proč kvalita ?

Kvalita je finančně efektivní

- › Základní cena (za práci samotnou)
- › Cena za nízkou kvalitu
 - Náklady na prevenci
 - Náklady na posouzení/zhodnocení
 - Náklady na opravu chyb nalezených zákazníkem nebo při posouzení / zhodnocení
- › Často **více než 50 %** nákladů za nízkou kvalitu !

Proč kvalita



QA v praxi

Poznatky z praxe

- › QA je nutné **napláňovat**
- › Proces musí být **pragmatický**
- › O kvalitě je nutné uvažovat **na všech úrovních** od organizace až po jedince
- › **Přezkoumání je efektivní** (a mnohdy jediný) způsob zajištění kvality
- › Začínat s QA ve fázi vývoje je pozdě



Softwarový proces

Parameter / Org. level	Definition, responsibility	Supervision review	Enactment	Record
Organization	Minimal practices Policies, standards Goodies	Reviews abstract	Tuesdays	Contracted/utilized MDs MD rate
Customer team	How-tos SAM	Reviews Proposal review	Fiscal tables Contracts Proposal process	Reviews records
Project	Project how-tos Min. practices mapping Project page Plan, risk management Maintenance PM	Reviews	Meetings	Reviews records Projects history
Person	Programming techn. Every person	Code review SAM review PM review	Pre-recruitment Recruitment Career Staffing Q-meetings	Personal plan



Minimální praktiky

- › Požadované praktiky softwarového procesu
- › Checklisty
- › Pokrývají tyto oblasti:
 - Organizace softwarového projektu
 - Konfigurační řízení
 - Prostředí
 - Požadavky
 - Design
 - Programování
 - Testování
 - Dokumentace
 - Validace a Verifikace

Odhady

- Definovaná metodika
- Unifikovaná napříč firmou
- Založená na best practices a vyladěná na naše typy projektů

Celkový přehled

	Min (MD)	Max (MD)	Nej. prav. (MD)	Prům. (MD)	Oček. (MD)	Riziko (%)
Analýza	61,0	102,0	83,6	81,5	82,9	13,7%
Design	32,5	58,6	48,2	45,6	47,3	8,8%
Implementace	90,3	149,3	125,7	119,8	123,7	19,8%
Testování	75,0	130,0	108,0	102,5	106,2	18,4%
PM	45,0	75,0	65,0	60,0	63,3	10,1%
Dodávka	17,0	24,0	21,2	20,5	21,0	2,3%
Ostatní	147,0	213,0	188,6	180,0	185,7	22,1%
Záruka (5%)	23,4	37,6	32,0	30,5	31,5	4,8%
Realizace (MD)	389,8	625,9	535,4	507,8	526,2	
Realizace včetně záruky (MD)	413,1	663,5	567,4	538,3	557,7	
Celkem bez záruky (MD)	467,8	751,9	640,2	609,8	630,1	
Celkem včetně záruky (MD)	491,1	789,5	672,2	640,3	661,6	

Hodnoty zadávám v MD
 Již zadané hodnoty se při změně nepřepočítají!

Úroveň týmu 60%

Podíl kategorií v celku

	Min (MD)	Max (MD)	Nej. prav. (MD)	Prům. (MD)	Oček. (MD)
Analýza	12,42%	12,92%	12,44%	12,73%	12,53%
Design	6,62%	7,43%	7,17%	7,12%	7,15%
Implementace	18,38%	18,91%	18,69%	18,70%	18,69%
Testování	15,27%	16,47%	16,07%	16,01%	16,05%
PM	9,16%	9,50%	9,67%	9,37%	9,57%
Dodávka	3,46%	3,04%	3,15%	3,20%	3,17%
Ostatní	29,93%	26,98%	28,06%	28,11%	28,07%
Podíl realizace	79,36%	79,28%	79,65%	79,31%	79,54%

Podíl kategorií v celku se bere ze Souhrnu včetně záruky

Seznam projektů

› Všechny projekty

<u>ID</u>	<u>Sev</u>	<u>Pri</u>	<u>OS</u>	<u>Assignee</u>	<u>Status</u>	<u>Resolution</u>	<u>Summary</u>
15259	nor	P1	Wind	vit.klугanost@profinit.cz	NEW		WIP::Scope::VJon:CPOJ:Blacklisty
9657	cri	P2	Wind	Michal.Houdek@profinit.cz	NEW		Revize - master bug
15614	nor	P2	Wind	michal.petrik@profinit.cz	NEW		WIP::SCOPE::JTrp:CSOBP:Paleta - Určení RPZ na MP
15935	nor	P2	Wind	petr.tietze@profinit.cz	ASSI		:::JHouska:VF:EDW Testing
15509	nor	P2	Wind	miroslav.piroh@profinit.cz	ASSI		:QA::MPiroh:Profinit:Profis
14574	nor	P2	Wind	oleg.smirskyy@profinit.cz	NEW		WIP::JTrpkos:Cardif:Převzetí WINGS
13984	nor	P2	Wind	radek.bohata@profinit.cz	NEW		WIP::JTrpkos:CSOB:Videobanker
13600	nor	P2	Wind	michal.petrik@profinit.cz	NEW		WIP::SCOPE::JTrp:CSOBP:Domky IV
11633	nor	P2	Wind	michal.petrik@profinit.cz	NEW		WIP::JTrp:CSOBP:Unifikace klienta
10198	nor	P2	Wind	miroslav.piroh@profinit.cz	NEW		WIP::JTrp:CSOB:PAYSEC maintenance
6544	nor	P2	Wind	jan.zamostny@profinit.cz	NEW		WIP::JZamostny:CS:CIC
5766	nor	P2	Wind	vladimir.rytir@profinit.cz	NEW		:::VRytir:CS:Partner 24 Development
15997	nor	P2	Wind	tomas.smolik@profinit.cz	NEW		WIP::JTrp:CSOBP:Baličky - Napojení S-Cube na Baličky
15510	nor	P2	Wind	lukas.hermann@profinit.cz	ASSI		:QA::LHermann:Profinit:TACR
14726	nor	P2	Wind	tomas.smolik@profinit.cz	NEW		:::VRytir:CS:LMS
14170	nor	P2	Wind	vladimir.rytir@profinit.cz	NEW		WIP::VRytir:CS:ECRS
13603	nor	P2	Wind	michal.petrik@profinit.cz	NEW		WIP::JTrp:CSOBP:Úprava taxace s využitím ČKPS
13505	nor	P2	Wind	vit.klугanost@profinit.cz	NEW		WIP::Time::JTrp:CSOBL:Partner
10463	nor	P2	Wind	martin.cerovsky@profinit.cz	NEW		:::JTrp:Cardif:DWH konzultace
8140	nor	P2	Wind	marek.susicky@profinit.cz	NEW		WIP::VRytir:CS:SecODS
16172	nor	P2	Wind	michal.podzimek@profinit.cz	NEW		:::MPodzimek:CSOBL:DWH Basel II
15877	nor	P2	othe	vladimir.rytir@profinit.cz	NEW		:::VRytir:CS:Audit MiddleWare
14181	nor	P2	Wind	vladimir.rytir@profinit.cz	NEW		:::VRytir:CS:EnergyDB
13778	nor	P2	Wind	vladimir.rytir@profinit.cz	NEW		::ScopeTimeQuality::VRytir:CS:PDB
13577	nor	P2	Wind	jan.zamostny@profinit.cz	NEW		WIP::JZamostny:CS:CIC New Delivery Model a release Jaro 2013
12681	nor	P2	Wind	vladimir.rytir@profinit.cz	NEW		WIP::VRytir:CS:PFCS Datamart

Seznam projektů

- › Všechny projekty
- › Priority
 - P1 = kritický, potřeba denního monitoringu
 - P2 = problematický, například riziko nárůstu rozsahu
 - P3 = bezproblémový projekt

15259	nor	P1	Wind	vit.kludanost@profinit.cz	NEW	WIP:Scope::VJon:CPOJ:Blacklisty
9657	cri	P2	Wind	Michal.Houdek@profinit.cz	NEW	Revize - master bug
15614	nor	P2	Wind	michal.petrik@profinit.cz	NEW	WIP::SCOPE::JTrp:CSOBP:Paleta - Určení RPZ na MP
15935	nor	P2	Wind	petr.tietze@profinit.cz	ASSI	:::JHouska:VF:EDW Testing

› Hlavička

- WIP existuje
- QA – quality assurance na vyšší úrovni
- Problém – rozsah, čas, kvalita, jiný
- Obchodník, zákazník
- Jméno projektu

WIP::SCOPE::JTrp:CSOBP:Paleta - Určení RPZ na MP
 :::JHouska:VF:EDW Testing
 :QA::MPiroh:Profinit:Profis
 WIP::JTrpkos:Cardif:Převzetí WINGS
 WIP::JTrpkos:CSOB:Videobanker
 WIP::SCOPE::JTrp:CSOBP:Domky IV

Projektová stránka v Confluence

- › Všechny základní informace
- › Provázaný na seznam projektů

Přehled projektu

Aplikace Partner slouží jako CMD (Customer Management Database) systém, pro potřeby ČSOB Leasingu. Slučuje informace o všech právnických a fyzických osobách (subjektech), se kterými ČSOB Leasing spolupracuje. Cílem projektu, je centralizovat a zvýšit datovou kvalitu udržovaných informací o spolupracujících subjektech.

Stav projektu	Finalizace poslední betaverze aplikace V0.3														
Stručný popis	Viz text výše														
Zákazník	ČSOB Leasing a.s.														
KAM/SAM	Rott/Trpkoš														
PM/Team leader	Vít Kluganost														
Obchodní bug	Bug 7577														
Revizní bug	Bug 13505														
Platná verze nabídky	http://svn-interni/viewwc/viewwc.cgi/csobl/obchod/nabidky/partner/														
Aktuální smlouvy	http://bugzilla-obchodni.profinet.lan/bugzilla_obchod/show_bug.cgi?id=7577														
Skutečná pracnost projektu a pracnost dle schváleného odhadu	Původní rozsah: Analýza 135 čd, Implementace 485 čd, Celkem 620 čd Navýšení: <table> <tr> <td>OZ</td><td>25,00</td></tr> <tr> <td>CODA</td><td>0,50</td></tr> <tr> <td>Grafika</td><td>8,00</td></tr> <tr> <td>Předběžné odhady</td><td>2,08</td></tr> <tr> <td>Xsell</td><td>6,5</td></tr> <tr> <td>Navýšení</td><td>42,08</td></tr> <tr> <td>Celkem</td><td>662,08</td></tr> </table>	OZ	25,00	CODA	0,50	Grafika	8,00	Předběžné odhady	2,08	Xsell	6,5	Navýšení	42,08	Celkem	662,08
OZ	25,00														
CODA	0,50														
Grafika	8,00														
Předběžné odhady	2,08														
Xsell	6,5														
Navýšení	42,08														
Celkem	662,08														

Projektové revize - PM

- › Dedikovaná osoba
- › Systémový přístup
- › Soulad s minimálními praktikami

Projekt/člověk revize - technické

- › Softwarový proces
- › Revize architektury
- › Revize designu
- › Revize kódu
- › ...

```
*****
* zhodnocení *****
*****
```

Plusy:

- 1) Příkladná znalost základů Javy a schopnost je jasně vysvětlit.
- 2) znáš i jinou viditelnost metod a polí než public a private.
- 3) obvykle vcelku rozumně volená jména proměnných a metod.

Doporučení:

- 1) Dbej na dodržování DRY (Don't Repeat Yourself) principu - výhradu k porušování DRY principu jsem musel v poznámkách níže velmi často opakovat. Toto je naprosto zásadní pro dlouhodobou údržbu kódu! Dobré zdůvodnění najdeš v knížce Pragmatic Programmer (viz doporučená literatura níže), včetně názorné ilustrace toho, na co všechno se tento princip dá (a měl by se) aplikovat.
- 2) Dodržuj zásady pro psaní srozumitelného kódu - používej konstanty místo čísel a řetězců přímo k kódu, rozděľuj metody na kratší funkční jednotky, důsledně piš dokumentační komentáře.
- 3) Snaž se nezneužívat side-efekty. Tento problém jsem našel ve validacích (viz poznámky 5, 7 níže) - pokud tvým záměrem je něco kontrolovat, kontroluj to přímo, a ne pozorováním side-efektu jiného příkazu (ke kontrole typu slouží v Javě instanceof, ne přetypování hodnoty při přiřazení do nikdy dále nečtené proměnné; ke kontrole objektu na null slouží porovnání s null, ne dereferencování objektu a čekání, zda vypadne NullPointerException). Pokud to opravdu nejde a side-effect musíš použít (tuším, že to je např. případ kontroly, zda lze řetězec převést na Integer), aspoň se ujisti, že z tvého kódu a komentářů je tento záměr jasně patrný. Využívání side-efektů tam, kde se tomu lze vyhnout, je jednak zbytečně neintuitivní a nesrozumitelné, druhak může být i nebezpečné - kompilátor (optimalizér) o tvém záměru využít side-efekt nemusí vědět a ty se pak můžeš dost divit (viz poznámka 5).

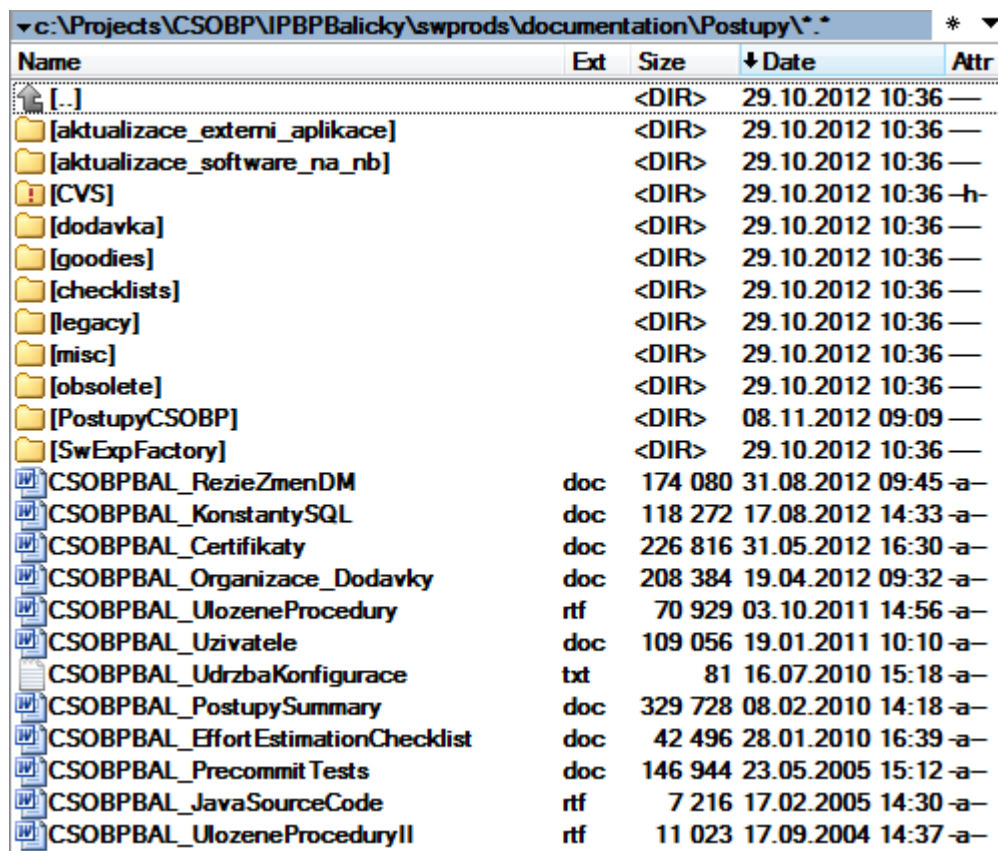
Literatura doporučená:

- The Pragmatic Programmer: From Journeyman to Master
- Effective Java (máme ji i v elektronické verzi)
- Článek OO in One Sentence - http://media.pragprog.com/articles/may_04_oo1.pdf
- Spring in Action, cokoli od Johnsona, POJO in Action a další podobné knihy (cílem je pořádně se naučit Spring a vidět, jak hezky skládat aplikaci)
- Rod Johnson - Expert One-on-One J2EE Design and Development
- Chapter 4 -

<http://media.techtarget.com/tss/static/articles/content/RodJohnsonInterview/JohnsonChapter4.pdf>

Procedury, checklisty, ...

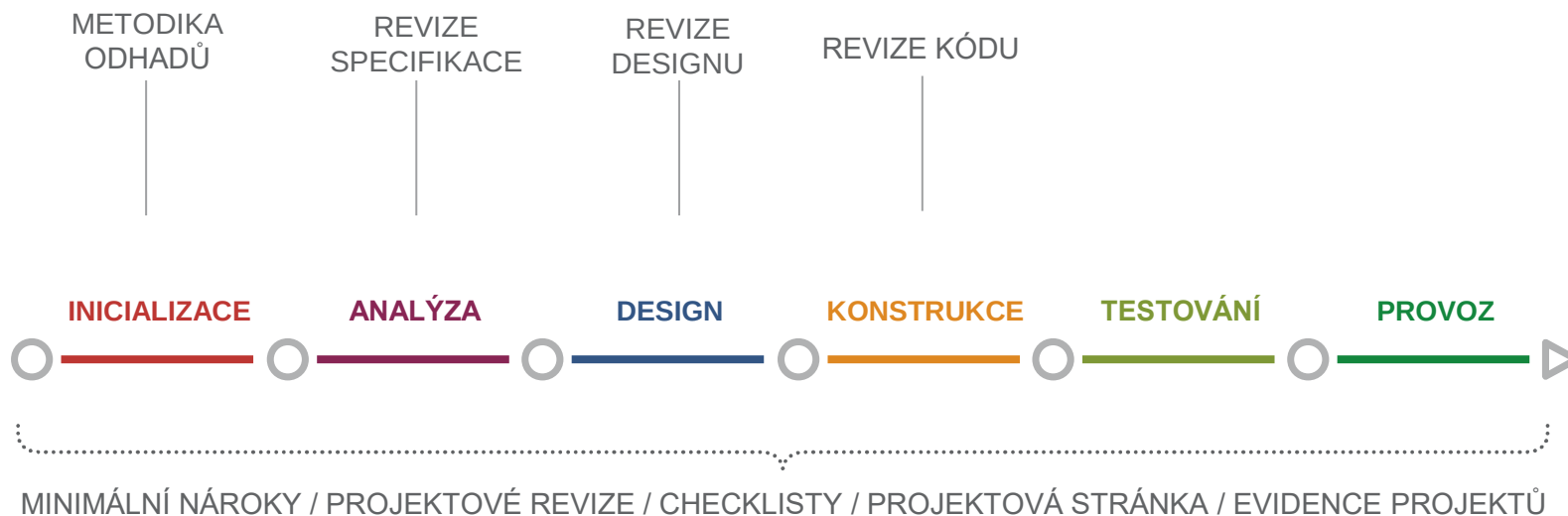
- › Typicky projektově specifické



The screenshot shows a Windows Explorer window with the address bar set to `c:\Projects\CSOBP\IPBPBalicky\swprods\documentation\Postupy\*.*`. The window displays a list of files and folders. The columns are Name, Ext, Size, Date, and Attr. The files are organized into a hierarchy of folders, including [aktualizace_externi_aplikace], [aktualizace_software_na_nb], [CVS], [dodavka], [goodies], [checklists], [legacy], [misc], [obsolete], [PostupyCSOBP], and [SwExpFactory]. Below these folders, there is a list of files with various extensions (doc, rtf, txt) and dates.

Name	Ext	Size	Date	Attr
[..]	<DIR>		29.10.2012 10:36	—
[aktualizace_externi_aplikace]	<DIR>		29.10.2012 10:36	—
[aktualizace_software_na_nb]	<DIR>		29.10.2012 10:36	—
[CVS]	<DIR>		29.10.2012 10:36	-h-
[dodavka]	<DIR>		29.10.2012 10:36	—
[goodies]	<DIR>		29.10.2012 10:36	—
[checklists]	<DIR>		29.10.2012 10:36	—
[legacy]	<DIR>		29.10.2012 10:36	—
[misc]	<DIR>		29.10.2012 10:36	—
[obsolete]	<DIR>		29.10.2012 10:36	—
[PostupyCSOBP]	<DIR>		08.11.2012 09:09	—
[SwExpFactory]	<DIR>		29.10.2012 10:36	—
CSOBPBAL_RezieZmenDM	doc	174 080	31.08.2012 09:45	-a-
CSOBPBAL_KonstantySQL	doc	118 272	17.08.2012 14:33	-a-
CSOBPBAL_Certifikaty	doc	226 816	31.05.2012 16:30	-a-
CSOBPBAL_Organizace_Dodavky	doc	208 384	19.04.2012 09:32	-a-
CSOBPBAL_UlozeneProcedury	rtf	70 929	03.10.2011 14:56	-a-
CSOBPBAL_Uzivatele	doc	109 056	19.01.2011 10:10	-a-
CSOBPBAL_UdrzbaKonfigurace	txt	81	16.07.2010 15:18	-a-
CSOBPBAL_PostupySummary	doc	329 728	08.02.2010 14:18	-a-
CSOBPBAL_EffortEstimationChecklist	doc	42 496	28.01.2010 16:39	-a-
CSOBPBAL_Precommit Tests	doc	146 944	23.05.2005 15:12	-a-
CSOBPBAL_JavaSourceCode	rtf	7 216	17.02.2005 14:30	-a-
CSOBPBAL_UlozeneProceduryII	rtf	11 023	17.09.2004 14:37	-a-

QA příklady



Shrnutí

- › Systematický přístup
- › Začít brzy – od iniciální fáze projektu
- › Pokračovat v rámci projektu
- › Snadno přístupné informace



Testování

Agenda

- › Co je testování
- › Vývojový proces a testování
- › Teorie testování – základní pojmy

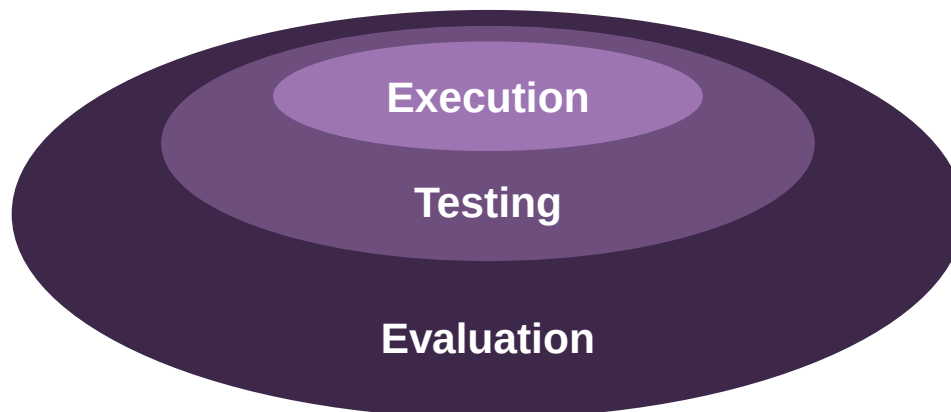
Cíl přednášky:

Cílem je porozumět principům, pojmům, závislostem a problémům testování.

Schematický pohled

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



Softwarový proces



Základní principy

- › Neexistuje software bez chyb
- › 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

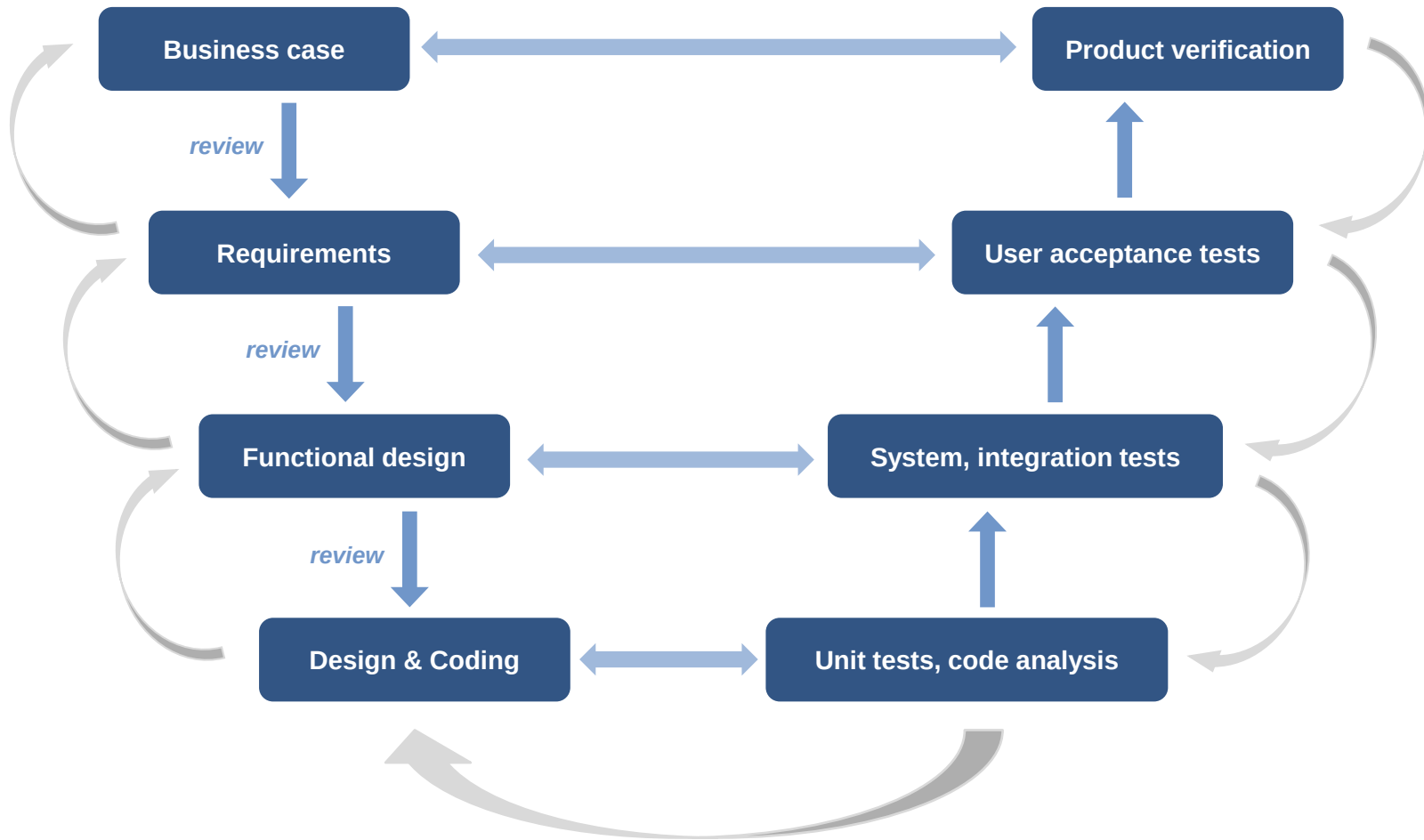
Trocha historie

Cíle testování

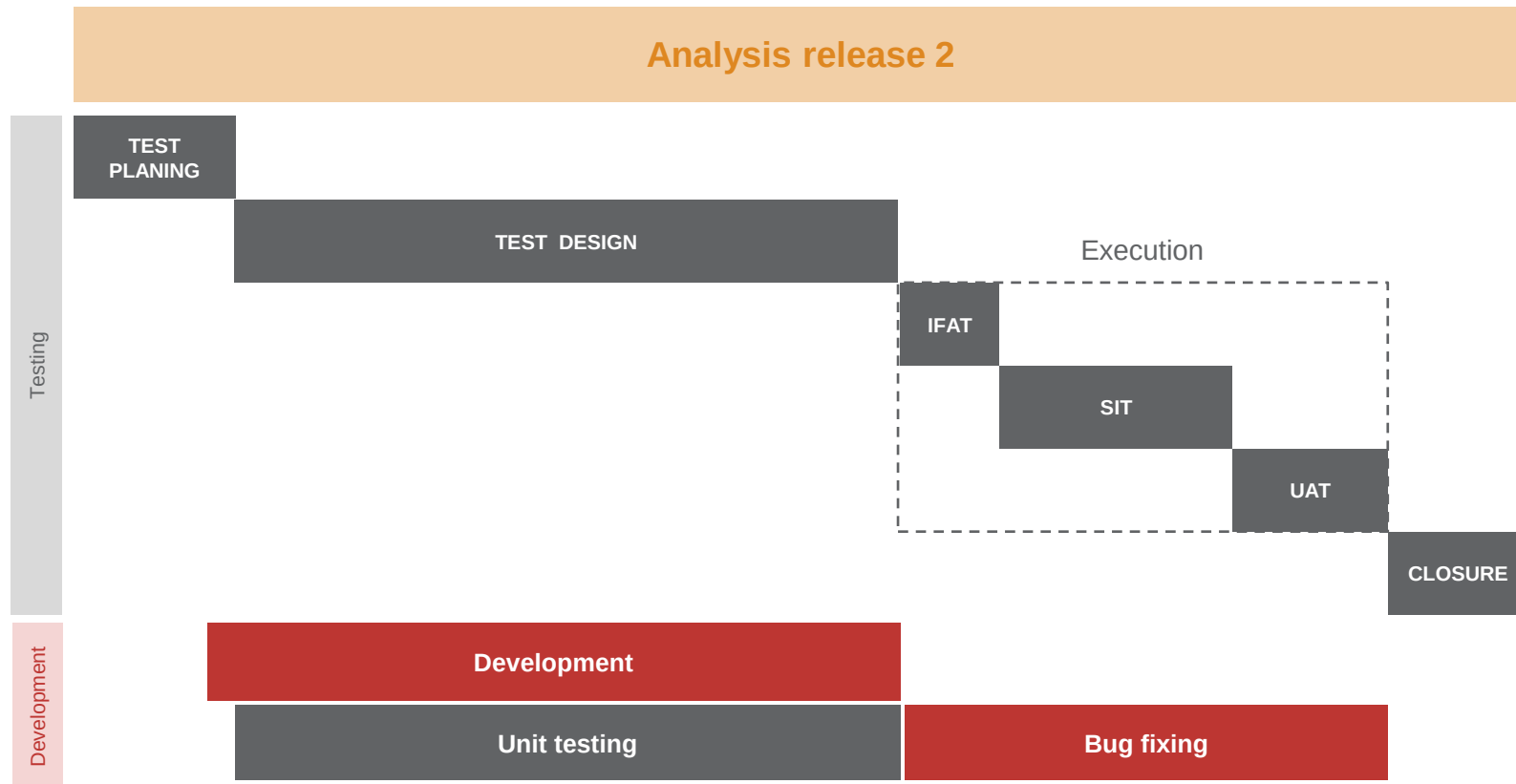
- › Pomoci jasně popsat chování systému
- › Nalézt defekty v
 - požadavcích,
 - designu,
 - dokumentaci a
 - kódu,jak nejdříve je to možné



Načasování testů aneb „V – model“



Časování v projektu



Trocha teorie testování

Typologie testů

› Tři různé dimenze

- co testujeme za konfigurační jednotku
- jaký aspekt konfigurační jednotky testujeme
- s jakým cílem testujeme

- Regresní testy
- Kvalifikační testy
- ...

- Funkční testy
- Výkonové testy
- Bezpečnostní testy
- Testy dostupnosti
- Testy spolehlivosti
- ...

- Unit testy
- Integrační testy
- Systémové testy
- Akceptační testy
 - Uživatelské
 - Operační
 - ...

Základy základů

Pozitivní vs. negativní testy

- › 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, ...

Black box

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

vs.

White box

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

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

Testovací techniky

Testovací techniky / paradigmatata

- › 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

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.

Výběr vhodné techniky

Na základě cíle testů

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

Na základě atributů

Atributy techniky

- › **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 skupinu testů, které se zaměřují na stejné riziko
- › **Motivating** – „klient“ bude chtít chyby nalezené testem opravit
- › **Performable** – proveditelný v souladu s návrhem
- › **Maintainable** – udržovatelný při změnách systému
- › **Repeatable** – snadno a levně znovupoužitelný
- › **Pop** (Karl Popper) – odhalí věci týkající se základních či kritických předpokladů
- › **Coverage** – vyzkouší systém způsobem, kterým to nečiní jiné testy
- › **Easy to evaluate** – snadné a jasné vyhodnocení
- › **Appropriately complex** – dostatečná komplexnost
- › **Accountable** – obhajitelnost, prokazatelnost testu
- › **Supports troubleshooting** – poskytuje užitečné informace k ladění nalezených problémů
- › **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 ...

Technika „Specification based“ - atributy

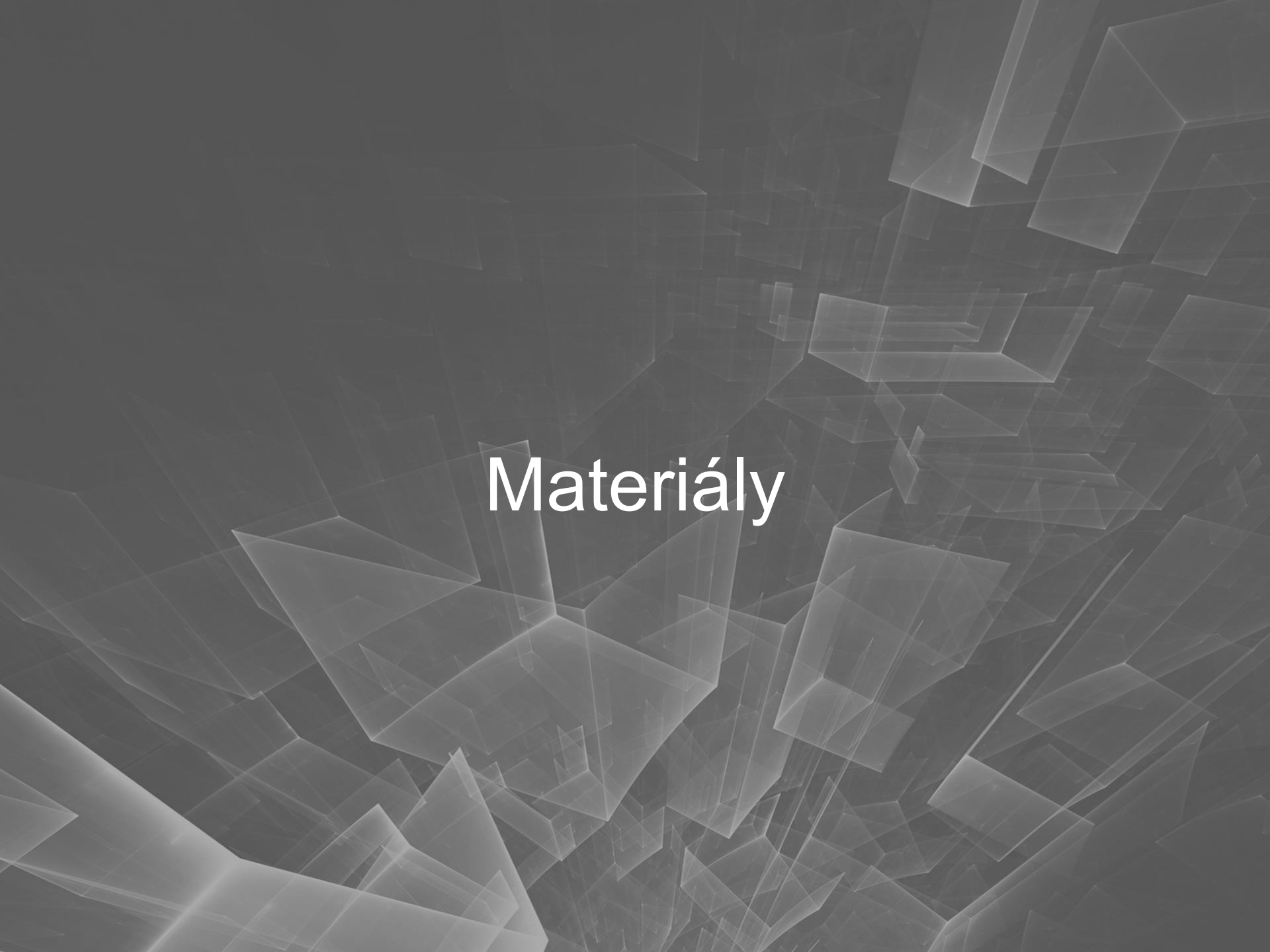
Atributy techniky

- › **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 skupinu testů, které se zaměřují na stejné riziko
- › **Motivating** – „klient“ bude chtít chyby nalezené testem opravit
- › **Performable** – proveditelný v souladu s návrhem
- › **Maintainable** – udržovatelný při změnách systému
- › **Repeatable** – snadno a levně znovupoužitelný
- › **Pop** (Karl Popper) – odhalí věci týkající se základních či kritických předpokladů
- › **Coverage** – vyzkouší systém způsobem, kterým to nečiní jiné testy
- › **Easy to evaluate** – snadné a jasné vyhodnocení
- › **Appropriately complex** – dostatečná komplexnost
- › **Accountable** – obhajitelnost, prokazatelnost testu
- › **Supports troubleshooting** – poskytuje užitečné informace k ladění nalezených problémů
- › **Cost** – přímé náklady, čas a pracnost
- › **Opportunity cost** – náklady, které se ušetří provedením testu

Technika „Stress“ - atributy

Atributy techniky

- › **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 skupinu testů, které se zaměřují na stejné riziko
- › **Motivating** – „klient“ bude chtít chyby nalezené testem opravit
- › **Performable** – proveditelný v souladu s návrhem
- › **Maintainable** – udržovatelný při změnách systému
- › **Repeatable** – snadno a levně znovupoužitelný
- › **Pop** (Karl Popper) – odhalí věci týkající se základních či kritických předpokladů
- › **Coverage** – vyzkouší systém způsobem, kterým to nečiní jiné testy
- › **Easy to evaluate** – snadné a jasné vyhodnocení
- › **Appropriately complex** – dostatečná komplexnost
- › **Accountable** – obhajitelnost, prokazatelnost testu
- › **Supports troubleshooting** – poskytuje užitečné informace k ladění nalezených problémů
- › **Cost** – přímé náklady, čas a pracnost
- › **Opportunity cost** – náklady, které se ušetří provedením testu



Materiály

MATERIÁLY SWENG

Rectangular Snip

QUALITY ASSURANCE

ČLÁNKY

- [ESA Software Engineering Standards – Issue 2](#) – standardy European Space Agency, součástí těchto standardů jsou i nároky na Quality assurance (viz Part 2, Chapter 5)
- [NASA Software Assurance Guidebook](#) – průvodce od NASA, který se systematicky věnuje základním aspektům tématu, velmi přehledně také vysvětluje vazby a souvislosti mezi QA a dalšími činnostmi na projektu (konfigurační řízení, ...)
- [Software Quality Assurance Process](#) – průvodce světem QA vytvořený ve Space and Naval Warfare Systems Center San Diego
- [NASA Software Formal Inspections Guidebook](#) – průvodce od NASA pro každého, kdo uvažuje o zavedení formálních inspekcí či reviews
- [Overcoming the NAH Syndrome for Inspection Deployment](#) – článek snažící se vyvrátit názor řady organizací a firem, že formální inspekce nejsou v jejich prostředí aplikovatelné (tzv. „Not applicable here“ – NAH syndrom)

TEMPLATES

- [Software Quality Assurance Plan Template](#) – template pro SQA plán od Space and Naval Warfare Systems Center San Diego, má těsou vazbu na výše uvedený [Software Quality Assurance Process](#)

Všechny odkazované materiály jsou poskytnuty výhradně za účelem výuky softwarového inženýrství.

OBLASTI SWENG

› Project and process

› Project management

› Documentation

› **Quality assurance**

› Configuration management

› Testing

› Construction

› Architecture and design

› Requirements

› Obecné, přehledové

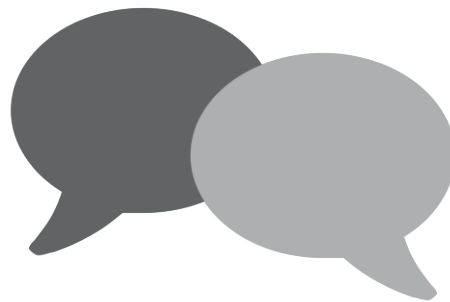


Hodnocení přednášky

<https://www.surveymonkey.com/r/2FQ876S>

nebo

<https://goo.gl/8G5Eue>



Diskuze

Děkujeme za pozornost

PROFINIT

NÁSKOK DÍKY ZNALOSTEM

Profinit EU, s.r.o.

Tychonova 2, 160 00 Praha 6 | Telefon + 420 224 316 016



Web
www.profinet.eu



LinkedIn
linkedin.com/company/profinet



Twitter
twitter.com/Profinit_EU



Facebook
facebook.com/Profinit.EU



Youtube
Profinit EU