Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky

Katedra informačních technologií

RUP disciplína Test

- charakteristika, role, činnosti, produkty, návody, vlastní zhodnocení

Bc. Vojtěch Kastner

ROK : 2011

Obsah

1. Úvod

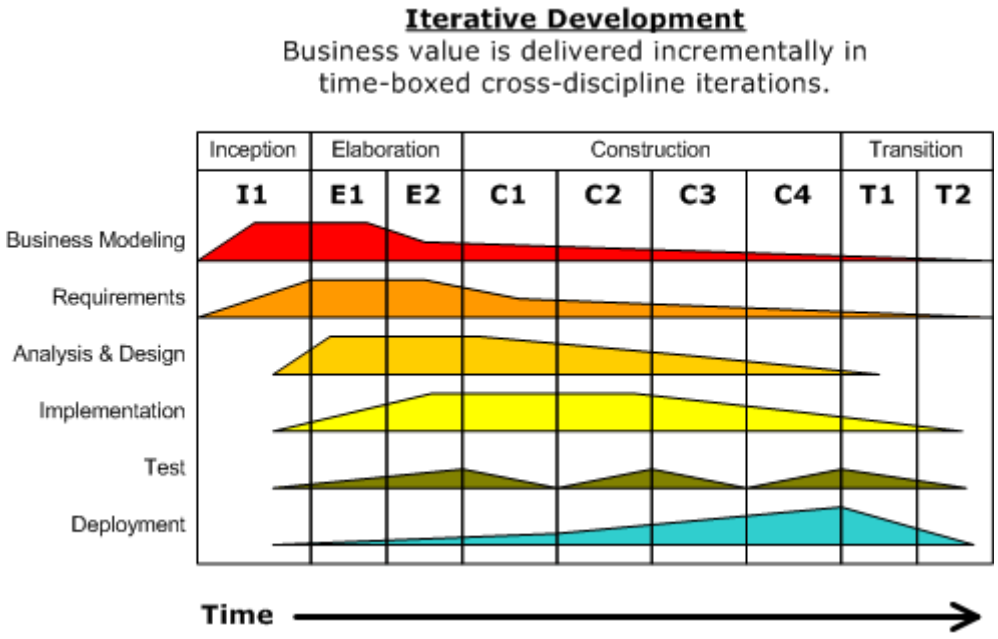
Tato práce pojednává o disciplíně Test medodiky RUP (Ration Unified Process). Téma jsem měl ještě s Bc. Antonínem Kučerou, který se zaměřil a detailně popsal jednotlivé role a s nimi spojené povinnosti. Čerpal převážně ze zdroje [5]. Proto byla má práce zaměřena více na praktické přiblížení disciplíny.

V první části bude disciplína zařazena do projektu, neboli jakými fázemi prochází. Následovat bude stručný návod (postup), jak danou disciplínu aplikovat. Společně s aplikací disciplíny budou představeny jednotlivé role, jejich povinnosti a pravomoce. V práci se také dočtete jaké artefakty (včetně jejich funkcí) jsou používány a stručný přehled nástrojů, které by měly sloužit k vytváření některých artefaktů či jinému ulehčení práce.

Protože vím, že je dost obtížné si představit jak disciplína funguje, napsal jsem na konci práce přesný postup, jak vytvořit testovací případ z případu užití. Vlastně praktická ukázka, jak má reálně takový proces tvorby testů vypadat.

2. Testování v RUP

„Cílem testování je vytvořit a provést soubor testů pro ověření funkčnosti komponent systému a jejich správné integrace.“ Na toto větné spojení jsem narazil na [4]. Nevím, jestli to patří mezi definice testování, rozhodně to ale stručně a jasně vystihuje cíl testování. Testování v RUP prochází všemi fázi životního cyklu viz Obrázek 1.



Obrázek 1: Životní cyklus projektu [2]

Na spodní ose obrázku 1 je vynesen čas, respektive plynutí času celého projektu. V horní části je obrázek rozdělen do 4 částí Zahájení (Inception), Rozpracování (Elaboration), Konstrukce (Construction) a Předání (Transition). Na vodorovné ose jsou pak jednotlivé disciplíny metodiky RUP. Jak můžeme vidět, testování prochází všemi fázemi, jak jsem již zmiňoval a kromě fáze Zahájení prakticky ve všech fázích stejně.

3. Postup při testování

Jedná se o stručný nástin postupu, jak bylo zmíněno v úvodu [2].

Krok 1:

Nejdříve je určeno co a proč bude testováno neboli cíl a oblast testování a zároveň se určí výstupní kritéria, kterých by mělo být dosaženo (chceme od systému nějakou funkčnost, takže si nadefinujeme jakou a následně zkontrolujeme, že systém opravdu dělá, co chceme). Tento krok se snaží zodpovědět otázku „CO a PROČ testovat?“

Krok 2:

Po určení cílů a oblasti testování následuje stanovení způsobu testů. Tedy jaké konkrétní testy se mají provést, aby byly splněny cíle testování, a jaká je pro ně požadovaná konfigurace testovaného systému. Tento krok se snaží zodpovědět otázku „JAK otestovat stanovené cíle?“

Krok 3:

V kroku 3 dochází k samotnému vytváření testů. Po tomto kroku by měly být připraveny testy včetně testovacích dat. Všechny testy by měly být spojovány do logických celků (aby mohlo docházet k jejich vyhodnocování) a zároveň by měly pokrýt celou oblast nadefinovanou v kroku 1.

Krok 4:

Tento krok obsahuje jak přípravu, tak i „provedení“ testů. Provedení je zde myšleno tak, že se testy aplikují, ale ještě nejsou spuštěny. V tomto kroku dochází k ověření, že systém je vhodný pro testy, respektive je nainstalován, spuštěn a „funkční“. Říká se tomu Smoke testy (viz kap 3 Typy testů).

Krok 5:

Samotné provedení testů. Zdá se, že v tomto kroku dochází ke spuštění jednoho tlačítka, jelikož je vše připraveno z kroku 4, ale zase tak jednoduché to není. Zde se provádí konfigurace systému dle vstupních požadavků (pro různé testy se systém chová různě, proto je potřeba na různé testy systém různě připravit) daného testu a spouští se test. V případě rozdílu mezi skutečnými a očekávanými výsledky z jednotlivých testů se požadovaným formátem dané rozdíly zaznamenají a nahlásí.

Krok 6:

Na závěr je provedena analýza výsledků testů. Nalezené rozdíly jsou v podobě žádostí o změnu předány k další analýze. Výstupem testů je Test report (artefakt), který podává informace o rozsahu provedených testů a jejich výsledků.

4. Testerské role

V RUP se vyskytují 4 testerské role [3]:

Manažer testování (Test Manager)

Jako každý manažer zastřešuje práci celého týmu. Jeho úkolem je řídit proces testování a řešit problémy, které nastanou. Zodpovídá za plánování veškerých zdrojů a kvalitu celého

testování (vystupuje jako tzv. advokát kvality). V postupu viz kapitola 3, vystupuje v kroku 1 a 6.

Test analytik (Test Analytik)

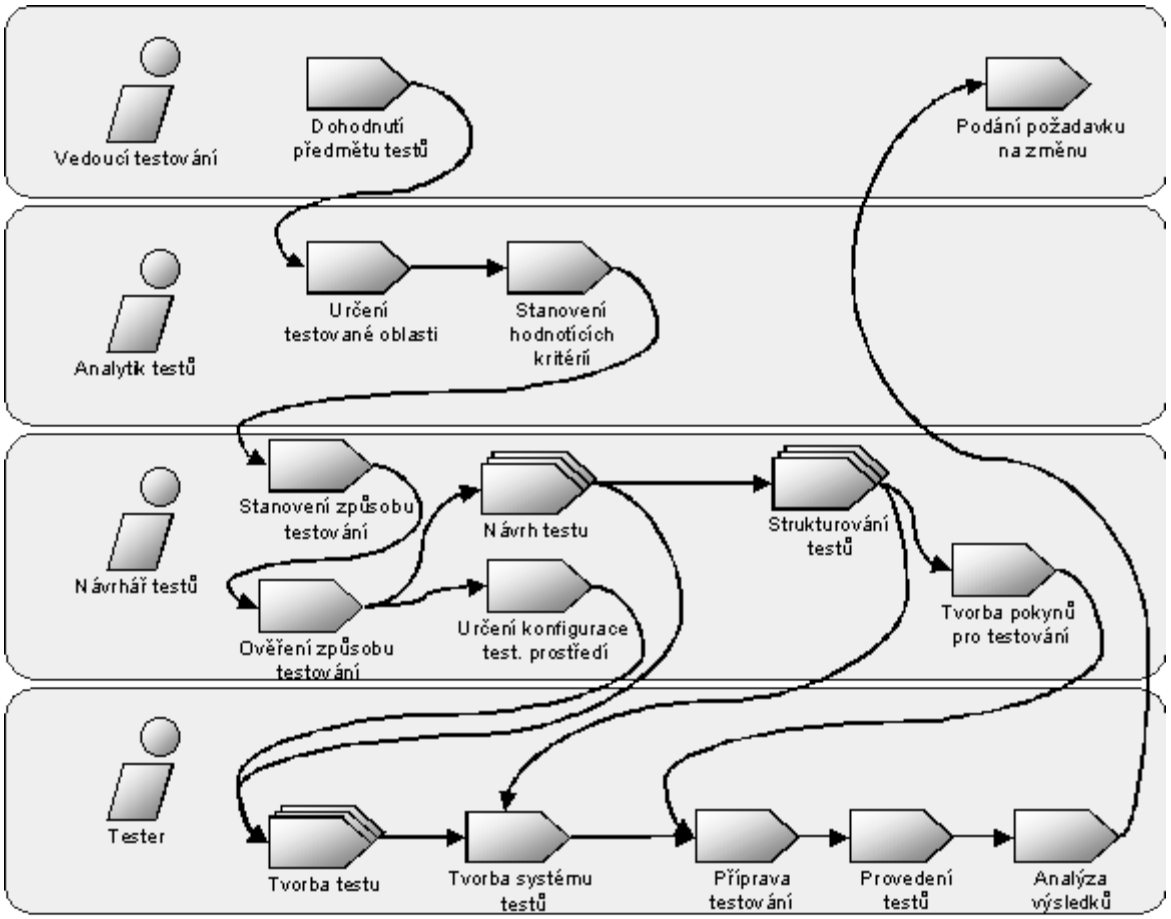
Test analytik má tyto úkoly: identifikace a popsání potřebných testů, sledování průběhu procesu testování, hodnocení dosažené úrovně kvality zjištěné během testování a zodpovídá za vhodnou volbu reprezentace výsledků testování. V postupu viz kapitola 3, vystupuje v kroku 2.

Návrhář testů (Test Designer)

Návrhář testů zodpovídá za definici přístupu k testování a s tím i spojenou realizaci. V rámci toho musí identifikovat vhodné postupy a nástroje včetně doporučení pro provádění testů a odhadu zdrojové náročnosti testování. V postupu viz kapitola 3, vystupuje v kroku 3 a 4.

Tester (Tester)

Na roli testera už nezbylo nic jiného než provádění samotných testů. Tester zodpovídá za správné provedení testů a zaznamenání průběhu a závěru (pokud testy neprošly, zaznamenání rozdílů do předpřipravených formulářů atp.). V postupu viz kapitola 3, vystupuje v kroku 4 a 5.



Obrázek 2: Ukázka rolí, jejich úkolů a komunikace mezi sebou [4]

Na obrázku 2 můžeme vidět v grafické podobě komunikaci, pravomoce a zodpovědnosti jednotlivých rolí. Pomocí šipek je pak nastíněno, jakým způsobem mezi sebou komunikují a mezi šipkami jsou pětiúhelníky, které jsou označují činnosti, což je stručně popsáno v kapitole 3.

5. Typy testů

Možná se většině zdá, že disciplína Test v metodice RUP zahrnuje veškeré testování. Přeci jen mít tým o několika lidech pouze na testování je velmi nákladné. Všichni dobře víme, že ve většině případů si testuje každý vývojář sám. To disciplína Test samozřejmě dělá také. V disciplíně Test se nejedná o testy jednotkové (vývojáři provádějí dle potřeby) a integrační (které mají samozřejmě na starosti již zmínění vývojáři). Testerský tým v RUP se zabývá testy systémovými a akceptačními [3].

Systémové testy by měly proběhnout několikrát během iterace, aby se otestovalo, že systém dělá opravdu to, co chceme a co nejdříve se tak mohlo zamezit případným chybám či nesrovnalostem s požadavky.

Akceptační testy se týkají závěrečné fáze projektu, kdy je produkt hotov a dochází k jeho předání zákazníkovi. Testujeme, zda je vše, co jsme chtěli funkční, správné a nic nechybí.

V metodice RUP se mluví o 2 typech testů [3]:

5.1 Smoke testy

Smoke test se využívá ve chvíli, kdy je dokončen vývoj aplikace a lze ji spustit. Jedná se o krátký test sloužící k rychlému ověření, že jsou všechny části implementovány, nainstalovány a spuštěny. Zabývá se pouze hlavní funkcionalitou aplikace. Dá se říci, že je to taková příprava k dalšímu testování. Smoke testy bývají většinou automatizovány.

5.2 Beta testy

Beta testování nastává při dokončení aplikace ještě předtím, než se aplikace nasadí na ostrý provoz. Aplikaci testují uživatelé a dále se testuje zátěž (simulace velkého počtu uživatelů = automatické generování velkého množství vstupních dat do aplikace - pokud je to možné). Pokud se naskytnou chyby v této fázi, tak se aplikace ještě upraví před úplným nasazením (odstranění zbylých chyb).

6. Artefakty

Než přistoupíme k samotnému příkladu generace testovacích případů z případů užití, ukážeme si několik nejdůležitějších artefaktů z disciplíny Test metodiky RUP (názvy artefaktů nepřekládány). Na konci popisu každého artefaktu (mimo pár výjimek) je v závorce uvedena role, která zodpovídá za daný artefakt [5].

Test Case

Tento artefakt definuje sadu testovacích vstupů, podmínek a očekávaných výsledků jednotlivých případů užití. (test analyst).

Test Data

Tento artefakt definuje soubor testovacích vstupů a očekávaných výsledků, které jsou využívány během provádění testu (test analyst).

Test Design

Tento artefakt popisuje strukturu testovacích prvků a realizaci testovacích případů (test designer).

Test Environment Configuration

Tento artefakt specifikuje uspořádání hardware, software a související nastavení prostředí, které jsou nutné k tomu, aby se mohli provádět testy (test designer).

Test Evaluation Summary

Tento artefakt obsahuje přehled výsledků testů (test manager)

Test Log

Tento artefakt obsahuje řádky výstupu zachycené během provedení jednoho nebo více jedinečných testů (tester).

Test Plan

Tento artefakt definuje cíle a záměry testování v rámci jedné iterace (nebo projektu) (test manager).

Test Results

Tento artefakt shrnuje analýzu jednoho nebo více Test logů a požadavků na změny, které poskytují poměrně podrobné posouzení kvality jednotlivých položek cíle testu (test analyst).

Test Script

Tento artefakt nám popisuje krok po kroku jakým způsobem realizovat testy (tester).

Test Strategy

Tento artefakt určuje strategický plán, jakým bude proveden test jednoho nebo více aspektu cílového systému (test designer).

Test Suite

Tento artefakt definuje kolekci souvisejících testů (test designer).

Test-Ideas List

Tento artefakt obsahuje výčet myšlenek (ne detailně zpracovaných), které identifikují potenciálně užitečné testy (test analyst).

Další artefakty: Workload Analysis Model, Test Automation Architecture, Test Interface Specification

Troufám si tvrdit, že není nutné, aby výše zmíněné artefakty byly všechny, některé mohou být sloučeny nebo vynechány. To může být závislé na velikosti a potřebách projektu.

Pro správnou tvorbu artefaktů a realizaci disciplíny Test v metodice RUP existuje spoustu nástrojů.

7. Přehled základních nástrojů

Rational ClearCase

Rational ClearQuest

Rational PurifyPlus

Rational Unified Process

Rational Robot

Rational TestFactory

Rational TestManager

Rational Test RealTime

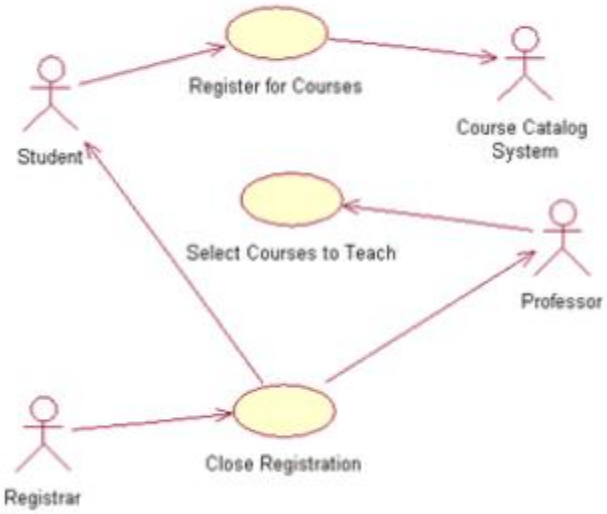
Tyto nástroje jsou uvedeny na oficiálních stránkách firmy IBM [5], kde jsou I popsány. Další nástroje můžete dohledat ve zdroji [3], kde se nachází více nástrojů se stručným popisem (nalezme tam i stejné nástroje jako na stránkách IBM).

8. Příklad generování test cases z use cases

Testovací případy a scénáře jsou v metodice RUP odvozené od případů užití systému. Takový model slouží k zachycení požadavků na systém a je odsouhlasen zákazníkem. Další informace o tvorbě případů užití by se mělo vyskytovat v disciplíně Pažadavky (Requirements) popřípadě Analýza a návrh (Analysis and Design). Zdroje pro kapitolu 8 jsem čerpal z [1].

8.1 Úvod do případů užití

Případy užití jsou založeny na Unified Modeling Language (UML) a můžou být vizuálně reprezentovány v use-case diagramech. Obrázek 3 ukazuje use-case diagram znázorňující požadavky na systém registrace kursů univerzity.



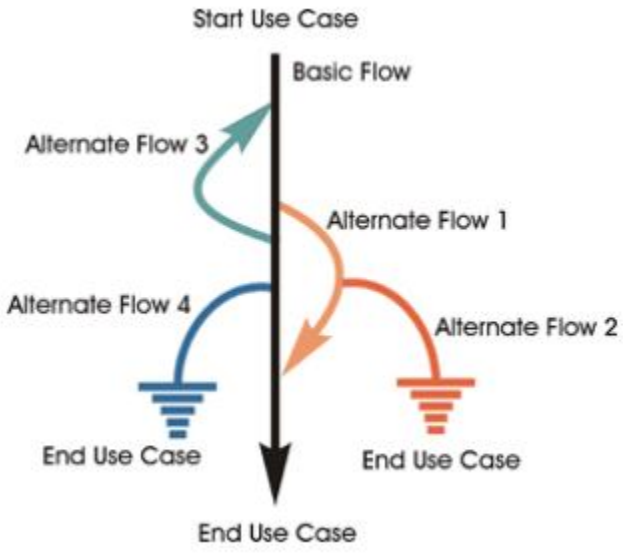
Obrázek 3: Diagram případů užití systému registrace kurzů [1]

Ovály reprezentují případy užití a postavičky představují aktéry, kterými mohou být buď lidé, nebo jiné systémy. Čáry se šipkami představují komunikaci mezi aktéry a případy užití. Jak je vidět na obrázku 1. Každý případ užití představuje značnou část funkcionality, které budou implementovány a každý aktér představuje někoho nebo něco mimo systém, který s ním komunikuje. Určitě je to dobrý krok k nastínění, jak má systém vypadat, ale samotný diagram většinou nestačí. Proto je potřeba udělat ještě slovní popis, který může vypadat následovně:

Table 1: Příklad textového popisu případů užití [1]

Sekce případů užití	Popis
Jméno	Jméno vhodné pro použití
Stručný popis	Stručný popis rolí a účelu (funkčnosti) systému
Tok událostí	Textový popis, co systém dělá samozřejmě s ohledem na případy užití (ne jak systém řeší problémy). Mělo by být srozumitelné pro zákazníka.
Zvláštní požadavky	Textový popis, který shromažďuje všechny požadavky, jako například nefunkční požadavky, které nejsou brány v úvahu na diagramu případu užití, ale systém je musí akceptovat (musí o nich být uvažováno v průběhu návrhu či implementace)
Předpoklady	Textový popis, který definuje každé omezení systému spadající do případu užití
Omezení	Textový popis, který definuje každé omezení systému, které nespadají do případů užití

Nejdůležitější součástí případu užití pro generování testovacích případů je tok událostí. Existují dvě hlavní části toku událostí a tím jsou základní tok událostí a alternativní tok událostí. Základní tok událostí by se měl týkat toho, co se „normálně“ (v ideálním případě) stane, když se provádí případ užití. Alternativní toky událostí se vztahují na chování volitelného nebo mimořádného charakteru ve vztahu k „normálnímu“ chování, ale mohou být i jeho součástí (například se vrátí k základnímu toku událostí – objížďka na silniční komunikaci).



Obrázek 4: Základní tok událostí s alternativními toky událostí pro případy užití (pracovní tok) [1]

Obrázek 4 představuje typickou strukturu toků událostí. Rovná svislá černá šipka směřující dolů představuje základní tok událostí a křivky (ostatní čáry) představují alternativní toky. Zde si můžeme všimnout, že některé alternativní toky událostí se vrací zpět na základní tok událostí, zatímco jiné alternativní toky ukončují případy užití. Základní i alternativní toky by měly být dále členěny do dalších kroků či podtoků (v našem případě to není potřeba, ale pro velké projekty nám jistě 5 toků nebude stačit).

8.2 Následují popisy jednotlivých toků registrace kurzů

8.2.1 Základní tok

1. **Přihlášení**
   Tento případ začíná, když se chce student přihlásit na web univerzity. Systém ho požádá o vstupní údaje, ID a heslo.

2. **Vytvoření studentova plánu**
   Systém zobrazí funkce, které má student k dispozici. Student si vybere „vytvořit plán“

3. **Získání informací o předmětu**
   Systém načte seznam dostupných (samozřejmě z katalogu nabídky) kurzů a zobrazí studentovi nabídku

4. **Vybrání kurzu**
   Student vybere jeden povinný kurz a dva volitelné kurzy ze seznamu nabídky kurzů

5. **Předložení plánu**
   Pro každý z vybraných kurzů se nabízí rozvrhový plán, který si vybírá student samozřejmě za předpokladu, že má student potřebné předpoklady (v podstatě nějaké podmínky)

6. **Zobrazení dokončeného rozvrhu**
   Systém zobrazí rozvrh, který obsahuje vybrané kurzy a potvrzovací číslo o rozvrhu

8.2.2 Alternativní toky (ukázka jen několika)

1. **Student neidentifikován**
   V kroku 1 základního toku, přihlášení, dojde k chybě, pokud systém zjistí, že studentovo ID s heslem není platné a zobrazí se chybová zpráva

2. **Ukončení**
   Systém umožňuje studentovi kdykoliv během případů užití ukončit zápis předmětů. Student si může zvolit pro uložení částečný rozvrh ještě před ukončením. Všechny kurzy, které nejsou označeny jako „zapsal“, jsou označeny jako „vybrané“ v rozvrhu. Rozvrh je samozřejmě uložen a případy užití končí.

3. **Nesplněné předpoklady, kurz je plný nebo došlo k rozvrhovému konfliktu**
   V kroku 5 základního toku neprovede systém zápis kurzu studenta do rozvrhu, pokud student nesplňuje předpoklady, kapacita kurzu je vyčerpána nebo dochází k rozvrhovému konfliktu. V takovém případě se studentovi zobrazí správa s odůvodněním a případnou další nabídkou ke zvolení jiného kurzu. Případ užití pokračuje v kroku 4 vybrání kurzu na základním toku

4. **Systém není k dispozici**
   Pokud je systém nedostupný, pak se zobrazí zpráva, že je systém nedostupný a případ užití okamžitě končí

5. **Registrace kurzů uzavřena**
   Pokud systém během případu užití zjistí, že registrace byla zavřena, zobrazí se zpráva a případ užití končí.

Každý krok by měl objasnit, co aktér dělá s reakcí systému na danou událost. Měl by být také číslovaný a popsaný. U alternativních toků by mělo být vždy určeno, kde začínají v základním toku a co udělají, když skončí.

8.3 Scénář případů užití

Je tu ještě jedna věc, kterou musíme popsat předtím, než se pustíme do toho, jak vygenerovat test z případů užití. Koncový uživatel hotového systému může projít pracovním tokem několika cestami. Nemá předem danou cestu, kterou musí jít, a proto se musí vytvořit scénář případů užití. Ukázka v tabulce 2.

Tabulka 2: scénáře k případům užití zobrazené v tabulce [1]

Scénář 1	Základní tok			
Scénář 2	Základní tok	Alternativní tok 1		
Scénář 3	Základní tok	Alternativní tok 1	Alternativní tok 2	
Scénář 4	Základní tok	Alternativní tok 3		
Scénář 5	Základní tok	Alternativní tok 3	Alternativní tok 1	
Scénář 6	Základní tok	Alternativní tok 3	Alternativní tok 1	Alternativní tok 2
Scénář 7	Základní tok	Alternativní tok 4		
Scénář 8	Základní tok	Alternativní tok 3	Alternativní tok 4	

Tyto scénáře budou použity pro tvorbu testů případů užití.

8.4 Generování testovacích případů

Testovací případy jsou sada testovacích vstupů, prováděných podmínek a očekávaných výsledků vyvinutých pro konkrétní cíl. Například vykonání určité cesty v programy nebo ověření souladu s konkrétním požadavkem.

Účelem testovacích případů je identifikovat a sdělit podmínky, které se budou provádět v testu. Testovací případy jsou nezbytné k ověření úspěšnosti a akceptační implementace projektových požadavků (případů užití).

Popíšeme si tři kroky procesu pro generování testovacích případů

1. Pro každý případ užití vytvoříme úplnou sadu scénářů daného případu užití

2. Pro každý scénář identifikujeme minimálně jeden testovací případ a podmínky, které je umožní provést.
3. Pro každý testovací případ identifikujeme data, se kterými budeme testovat.

8.4.1 Krok 1: generování scénářů

Přečíst si textový popis případů užití a identifikovat každou kombinaci základního toku s alternativními toky. Na základě toho vytvořit matici scénářů. Tabulka č. 3 ukazuje dílčí scénář matice pro případy užití registrace kurzů. Jedná se o jednoduchý příklad bez vnořených alternativních toků.

Tabulka 3: Částečný scénář matice pro případy užití registrace kurzů [1]

Název scénáře	Počáteční tok	Alternativní tok
Scénář 1 – úspěšná registrace	Základní tok	
Scénář 2 – neidentifikovaný student	Základní tok	Alternativní tok 1
Scénář 3 – ukončení	Základní tok	Alternativní tok 2
Scénář 4 – systém katalog kurzů není k dispozici	Základní tok	Alternativní tok 4
Scénář 5 – registrace kurzů uzavřena	Základní tok	Alternativní tok 5
Scénář 6 – Nesplněné předpoklady	Základní tok	Alternativní tok 3

8.4.2 Krok 2: Identifikace testovacích případů

Jakmile je identifikován celý soubor scénářů následuje krok identifikace testovacích případů. Můžeme to udělat například pomocí analýzy scénářů a přezkoumáním případů užití textovým popisem. Měl by tam být alespoň jeden testovací případ pro každý scénář, ale pravděpodobně jich tam bude víc. Například, je-li textový popis alternativního toku napsán velmi povrchně jako například:

Alternativní krok 3: Nesplněné předpoklady, kurs plný nebo rozvrhové konflikty

Pak mohou být požadovány další testovací případy, aby se vyzkoušely všechny možnosti. Kromě toho můžeme chtít přidat testovací případy pro testování vázaných podmínky.

Dalším krokem k vytvoření testovacích případů je znovu načtení případů užití textovým popisem a nalézání dalších podmínek nebo datových prvků nutných k provádění různých

scénářů. Pro případy užití registrace kurzů budou podmínkami student ID, heslo, vybrané kurzy atd.

Jasně dokumentovat testovací případy v matici je užitečné, jako v tabulce 4. Můžeme si všimnout v prvním řádku. První sloupec obsahuje id testovacího případu, druhý sloupec obsahuje stručný popis testovacího případu, včetně scénáře, který je testován a všechny další sloupce s výjimkou posledního obsahuje hodnoty, které budou použity při implementaci testů. V posledním sloupci je obsažen očekávaný výstup.

Tabulka 4: testovací případy v matici pro případy užití registrace kurzů [1]

Testovací případ	Scénář/podmínka	Studentovo ID	Heslo	Vybrané kurzy	Splněné předpoklady	Kurz otevřen	Rozvrh povolen	Očekávaný výsledek
TP 1	Scénář 1 – úspěšná registrace	V	V	V	V	V	V	Zobrazen rozvrh
TP 2	Scénář 2 – neidentifikovaný student	I	N/A	N/A	N/A	N/A	N/A	Chybná zpráva: vrácení obrazovky do přihlášení
TP 3	Scénář 3 – registrovaný studen ukončil	V	V	N/A	N/A	N/A	N/A	Zobrazení přihlašovacího okna
TP 4	Scénář 4 – systém registrace kurzů nedostupný	V	V	N/A	N/A	N/A	N/A	Chybná zpráva: zpět na krok 2
TP 5	Scénář 5 – registrace ukončena	V	V	N/A	N/A	N/A	N/A	Chybná zpráva: zpět na krok 2
TP 6	Scénář 6 – není možné se zapsat, kurz je plný	V	V	V	V	I	V	Chybná zpráva: zpět na krok 3
TP 7	Scénář 6 – není možné se zapsat, nesplněné předpoklady	V	V	V	I	V	V	Chybná zpráva: zpět na krok 4
TP 8	Scénář 6 – není možné se zapsat, rozvrhový konflikt	V	V	V	V	V	I	Chybná zpráva: zpět na krok 4

Všimněte si, že v této tabulce žádná data nejsou. Buňky tabulky obsahují V, I nebo N/A. „V“ označuje platnou vstupní hodnotu (valid), „I“ neplatnou (invalid) a „N/A“ znamená, že není nutné zadávat hodnotu (not available). Tato matice je dobrý mezistupeň. Ukazuje, jaké podmínky jsou testovány pro jaký testovací případ. To se dá jednoduše zjistit, při pohledu na V v tabulce a zda-li je určen dostatečný počet testovacích případů. Kromě prvního řádku,

který říká „vše funguje“, by měl každý řádek v matici na obrázku 4 obsahovat I (invalid) v případě testování podmínky. V TP3, TP4 a TP5 samozřejmě I (invalid) chybí, jelikož se netestuje podmínka, ale jedná o automatické ukončení registrace.

8.4.3 Krok 3: Identifikace hodnot testu do matice

Jakmile byly všechny testovací případy zjištěny, měly by být testovány a validovány, aby byla zajištěna přesnost identifikace nadbytečných nebo chybějících testovacích případů. Poté, co jsou schváleny, nastává poslední krok a to nahrazení V a I skutečnými hodnotami. Tabulka 5 ukazuje testovací případ matice s hodnotami, které nahrazují V a I v předchozí matici v tabulce 4. Existuje několik příkladů jak vytvořit skutečné hodnoty, ale to je už individuální. Jako příklad je tu tabulka 5.

Tabulka 5: Matice testovacích případů s hodnotami [1]

Testovací případ	Scénář/podmínka	Studentovo ID	Heslo	Vybrané kurzy	Splněné předpoklady	Kurz otevřen	Rozvrh povolen	Očekávaný výsledek
RC 1	Scénář 1 – úspěšná registrace	xkasv10	xyz321	4IT421 4IT444 4IT415	Ano	Ano	Ano	Zobrazen rozvrh
RC 2	Scénář 2 – neidentifikovaný student	xkasv10x	N/A	N/A	N/A	N/A	N/A	Chybná zpráva: vrácení obrazovky do přihlášení
RC 3	Scénář 3 – registrovaný studen ukončil	xkasv10	xyz321	N/A	N/A	N/A	N/A	Zobrazení přihlašovacího okna
RC 4	Scénář 4 – systém registrace kurzů nedostupný	xkasv10	xyz321	N/A	N/A	N/A	N/A	Chybná zpráva: zpět na krok 2
RC 5	Scénář 5 – registrace ukončena	xkasv10	xyz321	N/A	N/A	N/A	N/A	Chybná zpráva: zpět na krok 2
RC 6	Scénář 6 – není možné se zapsat, kurz je plný	xkasv10	xyz321	4IT421 4IT444 4IT415	Ano	Kurz 4IT444 je plný	Ano	Chybná zpráva: zpět na krok 3
RC 7	Scénář 6 – není možné se zapsat, nesplněné předpoklady	xkasv10	xyz321	4IT421 4IT444 4IT415	Nesplněné předpoklady pro kurz 4IT421	Ano	Ano	Chybná zpráva: zpět na krok 4
RC 8	Scénář 6 – není možné se	xkasv10	xyz321	4IT421 4IT444	Ano	Ano	4IT415 – rozvrhový	Chybná zpráva: zpět na

	zapsat, rozvrhový konflikt			4IT415			konflikt	krok 4

8.5 Spojení všeho dohromady

V současné praxi případy užití souvisí s první fází životního cyklu vývoje softwaru a testovací případy jsou obvykle spojeny s druhou částí životního cyklu. Díky využití případů použití pro generování testovacích případů však může testovací tým začít mnohem dříve v životním cyklu, což jim umožňuje identifikovat a opravit nedostatky už na začátku a tím se může předejít následným pozdějším opravám, což by bylo mnohem nákladnější a zajistí to spolehlivost systémů již v raném stádiu.

Díky použití jasně vymezené metodiky generování testovacích případů jsem nastínil, že je možné, aby vývojáři zjednodušili testovací proces, zvýšili tím efektivitu a došlo k úplnému pokrytí testů.

9. Závěr

O metodice RUP jsem do doby, než jsem začal psát tuto práci, věděl pouze základní údaje. Po napsání této práce jsem si rozšířil povědomí o RUP, neboť jsem se i částečně při zjišťování informací dopracoval alespoň okraje k ostatním disciplínám. Metodika RUP je pro mě velmi sympatická. Líbí se mi, že klade velký důraz na testování již od začátku projektu, čímž si může ušetřit nemalý obnos peněž.

Na druhou stranu jsem se s danou metodikou v praxi ještě nesetkal a tak nevím, jak opravdu reálně může fungovat. Nevýhodu metodiky vidím, v tom, že je velmi nákladná. Přecijen disciplína test může mít celou řadu osob, kteří v podstatě nic nevytváří, pouze kontrolují, jestli všechen vývoj spěje správným směrem. I samotná dnešní hektická doba, nepřispívá k používání metodiky RUP. Všude se šetří a nikomu se nechce utrácet další finanční prostředky na testování, přestože tuší, že by jim to nakonec prospělo.

Troufám si tvrdit, že se mi povedlo přiblížit disciplínu Test metodiky RUP natolik, aby po přečtení mé práce si udělal každý čtenář částečnou skutečnou představu, jak celý proces testování funguje.

10. Citovaná literatura

1. Rational Software. (2001). *Generating Test Cases From Use Cases.* Získáno 12 2011, z IBM: http://www.ibm.com/developerworks/rational/library/content/RationalEdge/jun01/GeneratingTestCasesFromUseCasesJune01.pdf

2. *Testování software*. (nedatováno). Získáno 2011, z Testování software: http://www.swtestovani.cz/index.php?option=com_content&task=view&id=43&Itemid=11

3. Alena Buchalcevová, J. K. (nedatováno). *Hodnocení metodik vývoje informačních systémů z pohledu testován.* Získáno 12 2011, z http://nb.vse.cz/~buchalc/clanky/testovani.pdf

4. Aldorf, F. (18. 6 2005). *RUP - disciplíny spojené s projektem*. Získáno 12 2011, z http://objekty.vse.cz/Objekty/Rup4

5. IBM Corporation. (2006). *Rational Unified Process*. Získáno 12 2011, z kitscm.vse.cz/RUP