9 Príloha A (Odovzdaná ponuka)

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava 4

Ponuka na tímový projekt

Viliam Kubis, Radoslav Zachar, Jakub Calík, Marián Hlavenka, Ondrej Danada, Matúš Hitka, Pavol Škvarenina

Tímový projekt 2011 / 2012

Ročník: 1
Ak. rok: 2011 / 2012

Obsah

1 Tím

Náš tím, s poradovým číslom 6, sa skladá z nasledovných členov:

Viliam Kubis, skúsený webový vývojár so šiestimi rokmi praxe pri vývoji dynamických webových stránok, ktorý sa aktivne venuje programovaniu pre mobilné platformy. V súčasnosti programuje aplikácie pre mobilný operačný systém Android. Vyniká schopnosťou učiť sa nové veci a objavovať nepoznané, čo je iste výhodné nielen v obyčajnom živote ale aj vo svete programovania. Ovláda jazyky a technológie PHP, MySQL, XHTML, XHTML, XML, CSS (1-3), JavaScript (+jQuery) na profesionálnej úrovni, medzi jeho obľúbené jazyky taktiež patrí štandardné C, v ktorom má mnoho skúseností. Jeho koníčkom je počítačová bezpečnosť a efektívnosť programov a skriptov, ktoré vytvára, alebo databáz, ktoré navrhuje. Má tiež skúsenosti s jazykmi Java a Lua. Taktiež ho zaujíma nízkourovňové poznanie počítača (assembler, hardware), rôzne komunikačné protokoly a informatika ako celok. Rád presne vie, ako každá vec funguje, aby si bol istý jej spoľahlivosťou a efektívnosťou.

Radoslav Zachar je cieľavedomý, zodpovedný a komunikatívny mladý programátor. Zaujíma sa o moderné informačne technológie a informatiku ako celok. Ovláda programovacie jazyky C, C++, C#, Java, PHP, HTML, CSS, XML, Delphi, databázy MySQL a PostgreSQL. Orientuje sa v operačných systémoch OS WIN a OS LINUX. Jeho programovacie znalosti sú najväčšie z oblasti vytvárania multiplatformových webových aplikácií v jazyku PHP prepojeným s CSS a akoukoľvek databázou. Vyniká schopnosťami v oblasti programovania v jazyku C++ alebo C#. Bez problémov programuje aj v prostedií .NET. Pri svojej práci sa snaží najmä o jednoduché a užívateľovi prívetivé grafické používateľské prostredie.

Jakub Calík, všestranne zameraný programátor, ktorý preferuje hlavne objektovo orientovaný prístup k programovaniu. Ovláda viaceré programovacie jazyky ako napr. C, C++, C#, Java, HTML. Taktiež sa orientuje v jednoduchých databazových systémoch a databázach MySQL a SQLite. Aktívne sa venuje vývoju aplikácií pre mobilné operačné systémy, hlavne iOS a Android. Prínosom sú aj základné poznatky z oblasti počítačovej grafiky a návrhu používateľských rozhraní.

Marián Hlavenka, zodpovedný a ambiciózny programátor, ktorého obľúbeným jazykom je C#. Preferuje objektovo orientovaný prístup k programovaniu a jeho skúsenosti zahŕňajú prácu s Pascalom, Assemblermi, HTML, Javou, C, C++ a Macromedia Flash Playerom. Má základy pri používaní operačných systémov na báze Unix, avšak jeho primárnym operačným systémom je Microsoft Windows, v ktorom je veľmi zbehlý.

Ondrej Danada, ovláda technológie a jazyky HTML, CSS, SQL, XML, JavaScript, AJAX, PHP, Java, UML, C/C++, Visual Basic, ASP, databázové systémy MsSQL a MySQL spolu so softvérovými produktami ako Visual Studio, Eclipse, Tortoise SVN a IBM RSA. Medzi jeho praktické skúsenosti patrí napríklad vývoj webovej JAVA aplikácie – šach na webe, vývoj, údržba, nasadenie a prevádzka informačného systému malej firmy a integrácia s Active Directory, CMS Joomla! a phpBB.

Matúš Hitka, programátor, ktorý si počas práce na rôznych školských zadaniach osvojil základné programovacie techniky jazykov C, C#, Java, HTML, XML, SQL, Pascal. Prednosť však dáva objektovo orientovanému programovaniu. Taktiež zvláda prácu v jednoduchých databázových systémoch. Je mladý, sebavedomý a veľmi prispôsobivý, schopný rýchlo sa naučiť nové veci.

Pavol Škvarenina je softvérový vývojár so štvorročnou praxou vo vývoji webových informačných systémov. Vo firme, v ktorej pracuje, nesie plnú zodpovednosť za vývoj určitých častí IS a taktiež, v prípade potreby, pomáha pri nasadzovaní IS u zákazníka. Do bázy jeho znalostí patria najmä technológie a prostredia .NET, ASP.NET, Silverlight, HTML, JavaScript, Oracle, MS SQL.

2 Motivácia

2.1 Rozvrhový systém novej FIIT (téma číslo 12)

Hlavná motivácia k danej téme spočíva v tom, že ide o vývoj, respektíve dolaďovanie systému, ktorý bude reálne nasadený v praxi. Táto práca by mala uľahčiť tvorbu rozvrhov ako študentom, tak aj pedagógom, teda výsledný produkt úsilia nášho tímu by mal prispieť k celkovému blahu všetkých ľudí na fakulte.

Samotný rozvrhový systém je napísaný v jazyku PHP a jedná sa o webovú aplikáciu, s čím má veľa ľudí v našom tíme enormné skúsenosti. Aplikácia založená na databáze PostgreSQL tvorí značne bezpečný a výkonný systém, ktorý je pre túto úlohu ako stvorený.

Je pre nás veľkou výzvou pracovať na projekte, ktorý neskončí len niekde v archíve, keďže cieľom je systém nasadiť a uviesť do prevádzky čo najskôr. Teší nás, že sa bude nový systém používať pre potreby celej novej FIIT a sme si istí, že uľahčí prácu nie len mnohým profesorom, ale v konečnom dôsledku aj nám, študentom. Jeden člen nášho tímu pracoval na problematike tvorby rozvrhov v rámci predmetu PSI v bakalárskom štúdiu. Nakoľko bola jeho práca veľmi dobre ohodnotená, očakávame, že jeho skúsenosti dopomôžu k vyššej úrovni vypracovávaného projektu. Veríme, že každý študent či pedagóg by bol vďačný za spustenie takéhoto systému a že každému z nich by zlepšil a spríjemnil život na fakulte.

Za dôležité považujeme snahu podporiť proces tvorby rozvrhov na fakulte lepším nástrojom na tvorbu rozvrhov s využitím podporného systému na zber požiadaviek k samotnej tvorbe týchto rozvrhov. Vypracovaním tohto projektu sa každému z nás určite rozšíria naše odborné znalosti a poznatky pri tvorbe moderných dynamických webových systémov na tvorbu rozvrhov a takisto aj zber informácií.

2.2 Textový editor obohatený o grafické prvky (téma číslo 11)

Každý z nás potrebuje pri práci kvalitný textový editor. Jeho výhody nemusíme zdôrazňovať. Textový editor je všeobecne najpoužívanejšia súčasť každého operačného systému.

Okrem kvalitného zvýrazňovania syntaxe pre mnoho rôznych jazykov by mal textový editor podporovať aj viacero pokročilých funkcií. Jednou z týchto funkcií môže byť napríklad presunutie akéhokoľvek bloku kódu (funkcia, podmienka, alebo cyklus) jednoduchým potiahnutím kurzora metódou Drag and Drop. Medzi ďalšie môžme zaradiť zbaľovanie a rozbaľovanie podmienok, funkcií a cyklov, automatickú nápovedu k parametrom rôznych funkcií, alebo samotným funkciám alebo dátovým typom. Tieto nadštandardné funkcie môžu fungovať už len pri podržaní kurzora myši nad potrebným názvom funkcie alebo štruktúry, čo je už len čerešničkou na torte.

Takýto textový editor by isto pomohol každému programátorovi, ale možno aj zdatnejšiemu používateľovi počítača. Sami vieme, koľko času často strávime iba hľadaním poradia parametrov pre funkciu v dokumentácii, alebo zistením významu daného parametra.

Vízia vývoja funkcií, ako sú napr. detekcia nedosiahnuteľného kódu, strom volaní, diagram tried, rôzne štatistické a optimalizačné funkcie, oznámenie nepoužitej premennej, nedefinovaného indexu poľa, možného pretečenia zásobníka a podobne nás ženú vpred k vývoju toho najlepšieho textového editora. Je to vízia textového editora ušitého dokonale pre nás - programátorov. Skutočnosť, že sa jedná o rozšírenie a doladenie už existujúceho editora, robia z tímového projektu príležitosť vytvoriť naozaj plne funkčný a použiteľný program.

Existujúci editor je napísaný vo veľmi zaujímavom jazyku C++ spolu s veľmi prenosným, ľahko použiteľným a hlavne platformovo nezávislým grafickým toolkitom Qt, ktorý vytvára ideálne prostredie pre ďalší vývoj aplikácie. Je to príležitosť vytvoriť takmer dokonalý textový editor s ideálnymi grafickými prvkami pre každého používateľa. Bohaté skúsenosti niektorých členov tímu v oblasti lexikálnej a syntaktickej analýzy zdrojového kódu by mohli byť pre projekt taktiež veľmi prínosné.

3 Koncepcia riešenia

3.1 Rozvrhový systém novej FIIT (téma číslo 12)

Aktuálne rozpracovaný rozvrhový systém novej FIIT minuloročným tímom je v štádiu, v ktorom je už mnoho častí systému navrhnutých a napísaných, avšak taktiež je ešte treba mnoho súčastí vytvoriť / upraviť. Keďže je celý systém napísaný v skriptovacom jazyku PHP, pričom konkrétne používa PHP framework YII a databázu PostgreSQL, bude pre náš tím veľmi ľahké na projekt nadviazať a vo vývoji pokračovať.

Našou konkrétnou víziou je dorobenie plne automatizovaného systému na tvorbu rozvrhov, pričom by sme sa chceli zamerať na nasledovné vlastnosti a funkcie:

- automatická tvorba rozvrhov na základe požiadaviek
- burza cvičení (študenti si budú mocť vymienať cvičenia, kus sa kus alebo doplatkom "bodov")
- prepojenie systému s akademickým informačným systémom (AIS) - vygenerované rozvrhy sa do AIS automaticky vložia
- možnosť vytvoriť si záujmové skupiny pri vsádzaní si na rozvrhové akcie - študent si založí skupinu, v ktorej so svojimi kamarátmi spoja svoje kredity a budú si mocť vsádzať na rozvrhové akcie spoločne
- podpora riešenia kolízií v rozvrhu

Náš tím má bohaté skúsenosti s bezpečnosťou web stránok, pred nasadením systému do praxe bude systém dôkladne bezpečnostne otestovaný proti prípadným bezpečnostným chybám, ktoré budú následne opravené.

Je vízia, že sa systém dostane do ostrej prevádzky už tento školský rok a že bude úspešne vytvárať rozvrhy už pre budúce ročníky študentov.

3.2 Textový editor obohatený o grafické prvky (téma číslo 11)

Cieľom je vytvoriť editor s množstvom užitočných funkcií, medzi ktoré patrí napríklad:

- zvýrazňovanie zátvoriek v kóde, zvýrazňovanie blokov kódu
- zabaľovanie a rozbaľovanie blokov kódu
- kontrola syntaxe, ako napríklad upozorňovanie na neuzatvorené zátvorky, chýbajúce bodkočiarky a podobne
- kontextové nápovedy k funkciám, parametrom funkcií a ku premenným v programe
- možnosť pracovať s viacerými súbormi naraz, teda mód projektu (nápoveda by mala pracovať nad všetkými súbormi v projekte)
- podpora pre viacero programovacích jazykov
- pokročilé štatistiky zdrojového kódu (počet funkcií, premenných, počet volaní funkcií, ..)
- možnosť zostrojiť class diagram alebo call diagram
- možnosť prejsť na definíciu funkcie
- možnosť presúvania blokov kódu metódou drag & drop bez nutnosti osvieťovať daný text
- detekcia nedosiahnuteľného kódu, detekcia nepoužitej premennej alebo funkcie

Keďže sa jedná o už existujúci projekt, cieľom bude doplnenie dohodnutej funkcionality a dovedenie projektu do takej úrovne, kedy bude plne použiteľný a veľmi používateľsky prívetivý a nápomocný pre nás - programátorov.

Príloha A – Zoradenie všetkých tém podľa priority

1. Rozvrhový systém novej FIIT (Rozvrhy) [12]
2. Textový editor obohatený o grafické prvky (TextEdit) [11]
3. Inteligentná hra pre mobilné zariadenia (MobHra) [8]
4. Znalosti a zručnosti študentov (Znalosti) [13]
5. Personalizované odporúčanie (Odporúčanie) [5]
6. Digitálne divadlo (Divadlo) [3]
7. Osobný manažment fyzickej aktivity pomocou mobilných zariadení (Aktivita) [2]
8. Imagine Cup 2012: Game Design (ICup2012) [1]
9. Webový editor pre TeX (WebEdit) [10]
10. Tvorba "ľahko" sémantického obsahu pre adaptívny webový (výučbový) portál (ALEF) [6]
11. RoboCup - tretí rozmer (RoboCup) [7]
12. Plagiáty na webe (Plagiáty) [4]
13. Simulácia davu (Dav) [15]
14. Štatistický preklad voľného textu (Preklad) [9]
15. 3D UML (3D UML) [17]
16. Editovanie viacrozmerného grafu prepojenia informácií v dokumentoch (Dokumenty) [16]
17. Virtuálna FIIT (VirtFIIT) [14]

Príloha B - Aktuálny rozvrh všetkých členov tímu

	7:00 - 7:50	8:00 – 8:50	9:00 – 9:50	10:00 – 10:50	11:00 – 11:50	12:00 – 12:50	13:00 - 13:50	14:00 - 14:50	15:00 – 15:50	16:00 – 16:50	17:00 – 17:50	18:00 – 18:50	19:00 – 19:50	20:00 – 20:50
Pondelok														
Viliam Kubis														
Radoslav Zachar														
Jakub Calík														
Marián Hlavenka														
Matúš Hitka														
Ondrej Danada														
Pavol Škvarenina														
Utorok														
Viliam Kubis											?	?	?	?
Radoslav Zachar											?	?	?	?
Jakub Calík											?	?	?	?
Marián Hlavenka											?	?	?	?
Matúš Hitka											?	?	?	?
Ondrej Danada											?	?	?	?
Pavol Škvarenina											?	?	?	?
Streda														
Viliam Kubis					navrhovaný čas stretnutí									
Radoslav Zachar														
Jakub Calík														
Marián Hlavenka														
Matúš Hitka														
Ondrej Danada														
Pavol Škvarenina														
Štvrtok														
Viliam Kubis														
Radoslav Zachar														
Jakub Calík														
Marián Hlavenka														
Matúš Hitka														
Ondrej Danada														
Pavol Škvarenina														

	7:00 - 7:50	8:00 – 8:50	9:00 – 9:50	10:00 – 10:50	11:00 – 11:50	12:00 – 12:50	13:00 - 13:50	14:00 - 14:50	15:00 – 15:50	16:00 – 16:50	17:00 – 17:50	18:00 – 18:50	19:00 – 19:50	20:00 – 20:50
Piatok														
Viliam Kubis														
Radoslav Zachar														
Jakub Calík														
Marián Hlavenka														
Matúš Hitka														
Ondrej Danada														
Pavol Škvarenina														

10 Príloha B (Preberací protokol)

Preberací protokol

Rozvrhový systém novej FIIT

Tímový projekt 2011/2012

Odovzdávajúci tím: Rozvrhári 2.0 (tím č. 6)
Preberajúca osoba: Ing. Miroslav Galbavý (vedúci projektu)

Kontakt na tím: http://vm07.ucebne.fiit.stuba.sk/
tim6@danada.sk

Predmet prebratia:

- odovzdanie dokumentácie k prvému kontrolnému bodu
- odovzdanie dokumentácie obsahujúcej zadanie, opis problémovej oblasti, analýzu systému, špecifikáciu požiadaviek, návrh riešenia a inštalačnú a používateľskú príručku.

V Bratislave, dňa 11.11.2011

....

podpis zástupcu odovzdávajúceho tímu podpis preberajúcej osoby

Preberací protokol

Rozvrhový systém novej FIIT

Tímový projekt 2011/2012

Odovzdávajúci tím: Rozvrhári 2.0 (tím č. 6)
Preberajúca osoba: Ing. Miroslav Galbavý (vedúci projektu)

Kontakt na tím: http://vm07.ucebne.fiit.stuba.sk/
tim6@danada.sk

Predmet prebratia:

- odovzdanie dokumentácie tímového projektu za zimný semester
- odovzdanie dokumentácie obsahujúcej zadanie, opis problémovej oblasti, analýzu systému, špecifikáciu požiadaviek, návrh riešenia a inštalačnú a používateľskú príručku.
- odovzdanie dokumentácie k riadeniu

V Bratislave, dňa 13.12.2011

....

podpis zástupcu odovzdávajúceho tímu podpis preberajúcej osoby

11 Príloha C (Metodiky)

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Bc. Pavol Škvarenina

Metodika manažmentu iterácií

Bratislava, 11.11 2011

1. Úvod

Predkladaná metodika definuje postup manažovania iterácií projektu. Opisuje procesy plánovania, vytvárania, kontroly, ukončenia a revízie iterácie a ich nadväznosti, ktoré musia byť dodržiavané každým vedúcim vývojárskeho tímu.

2. Pojmy

Backlog - úložisko artefaktov určitého typy

Iterácia - jeden cyklus vývoja s jasným cieľom a so stanoveným začiatkom a koncom

Používateľský príbeh - funkčná požiadavka na produkt, ktorá môže byť dokomponovaná na úlohy, ktoré budú implementované v rámci iterácie

3. Manažment iterácií projektu

Manažment iterácií projektu opisovaný v tejto metodike je primárne založený na inkrementálnej a iteratívnej metodike vývoja, ale obsahuje aj prvky agilnej metodiky Scrum (pojem Backlog, rola Produktového vlastníka).

3.1. Role a zodpovednosti

Rola	Zodpovednosti
Vedúci tímu	• Plánovanie iterácie a vytvorenie plánu iterácie. • Vytvorenie iterácie a naplnenie Backlogu iterácie. • Kontrola priebehu iterácie a vykonávanie potrebných zmien v iterácii. • Ukončenie iterácie. • Revízia iterácie.
Produktový vlastník	• Priorizácia a výber používateľských príbehov pre implementáciu v danej iterácii.

3.2. Artefakty manažmentu iterácií

Artefakt	Definícia
Backlog iterácie	Predstavuje úplný zoznam úloh, ktoré budú vývojárskym tímom implementované v danej iterácii.
Backlog produktu	Predstavuje zoznam používateľských príbehov, ktoré môžu byť v rámci vývoja produktu implementované.

Graf priebehu iterácie	Graf priebehu iterácie zobrazuje súčet zostávajúcich odhadnutých hodín na implementáciu úloh v danom dni. V prvý deň iterácie dosahuje graf maximum a v nasledujúcich dňoch postupne klesá, až v ideálnom prípade dosiahne v deň konca iterácie nulovú hodnotu.
Plán novej verzie	Výstup procesu plánovania novej verzie, ktorý pre potreby manažmentu iterácii obsahuje nasledovné časti: • zoznam plánovaných iterácií, • ciele každej iterácie, • dátum začiatku a konca každej iterácie.
Plán iterácie	Plán iterácie obsahuje: • dátum začiatku a konca iterácie, • cieľ iterácie, • množstvo alokovaných hodín na vývoj, • zoznam používateľských príbehov dekomponovaných na úlohy, ktoré sa budú implementovať, • zoznam chýb, ktoré je nutné v iterácií opraviť, • zoznam úloh, ktoré treba dokončiť z predchádzajúcej iterácie.
Revízia iterácie	Revízia iterácie obsahuje: • zoznam úloh, ktoré nemohli byť dokončené v danej iterácii spolu s dôvodmi nedokončenia, • zoznam prekážok, ohrození alebo iných negatívnych vplyvov, ktoré iteráciu ovplyvnili, • zoznam pozitívnych skúseností nadobudnutých počas iterácie, • zoznam návrhov na vylepšenie procesu vývoja.

3.3. Procesy manažmentu iterácií

Číslo	**Proces**	**Kapitola**
1.	Plánovanie iterácie	3.3.1
2.	Vytvorenie iterácie	3.3.2
3.	Kontrola priebehu iterácie	3.3.3
4.	Ukončenie iterácie	3.3.4
5.	Revízia iterácie	3.3.5

3.3.1. Plánovanie iterácie

Vstup: Plán novej verzie, Backlog produktu, Revízia predchádzajúcej iterácie
Výstup: Plán iterácie
Zodpovedný: Vedúci tímu, Produktový vlastník

Vedúci tímu s produktovým vlastníkom a ostatnými členmi tímu, vyberú na základe Plánu novej verzie, obsahu Backlogu produktu a Revízie predchádzajúcej iterácie používateľské príbehy, ktoré budú implementované v plánovanej iterácii. Tím za pomoci vedúceho tímu dekomponuje používateľské príbehy na úlohy a pridáva ich do Plánu iterácie. Do plánu sú tiež pridané chyby a nedokončené úlohy z predchádzajúcej iterácie.

3.3.2. Vytvorenie iterácie

Vstup: Plán iterácie
Výstup: Iterácia, Backlog iterácie
Zodpovedný: Vedúci tímu

Vedúci tímu vytvorí novú iteráciu a založí nový Backlog iterácie. V Backlogu iterácie vytvorí úlohy pre jednotlivých členov tímu na základe vybraných dekomponovaných používateľských príbehov z plánu iterácie. Do Backlogu iterácie sú tiež pridané nedokončené úlohy z predchádzajúcej iterácie a úlohy opravujúce chyby.

3.3.3. Kontrola priebehu iterácie

Vstup: Iterácia, Graf priebehu iterácie
Výstup: Zmena iterácie
Zodpovedný: Vedúci tímu

Vedúci tímu pravidelne kontroluje priebeh iterácie na základe grafu jej priebehu. V prípade problémov vykonáva zmeny v iterácii, a to zmenu priority úlohy, presunutie úlohy do nasledujúcej iterácie a zrušenie implementácie úlohy.

3.3.4. Ukončenie iterácie

Vstup: Iterácia, Plán iterácie
Výstup: Ukončená iterácia
Zodpovedný: Vedúci tímu

Vedúci tímu v deň naplánovaného ukončenia iterácie, najskôr s jednotlivými členmi tímu preberie zostávajúce úlohy jednotlivých členov tímu. Ak nie je možné niektorú úlohu dokončiť v deň ukončenia iterácie, je označená na presun do nasledujúcej iterácie. Následne v dohodnutom časovom momente vedúci tímu iteráciu ukončí.

3.3.5. Revízia iterácie

Vstup: Ukončená iterácia
Výstup: Revízia iterácie
Zodpovedný: Vedúci tímu

Vedúci tímu spolu s ostatnými členmi tímu vykonáva revíziu iterácie v rámci diskusie, v ktorej sa zaznamenáva výsledok iterácie, splnené a nesplnené ciele spolu s dôvodmi nesplnenia, pozitívne a negatívne skúsenosti jednotlivých členov tímu a návrhy na zlepšenie samotného procesu vývoja.

4. Metodika ukončenia iterácie

Účelom tejto metodiky je definícia postupu ukončenia iterácie v systéme Redmine. Opisuje jednotlivé procesy a kroky, ktoré je nutné vykonať pred ukončením iterácie a proces ukončenia iterácie.

4.1. Pojmy

Backlog - úložisko artefaktov určitého typy
Iterácia - jeden cyklus vývoja s jasným cieľom a so stanoveným začiatkom a koncom
Redmine - open-source systém na manažovanie projektov a sledovanie chýb

4.2. Role a zodpovednosti

Rola	Zodpovednosti
Vedúci tímu	• Kontrola grafu priebehu iterácie. • Kontrola stavu backlogu iterácie. • Naplánovanie kontroly nedokončených úloh s ich vlastníkmi. • Kontrola nedokončených úloh s ich vlastníkmi. • Ukončenie iterácie v systéme Redmine.

4.3. Artefakty procesu ukončenia iterácie

Artefakt	Definícia
Backlog iterácie	Predstavuje úplný zoznam úloh, ktoré budú vývojárskym tímom implementované v danej iterácii.
Graf priebehu iterácie	Graf priebehu iterácie zobrazuje súčet zostávajúcich odhadnutých hodín na implementáciu úloh v danom dni. V prvý deň iterácie dosahuje graf maximum a v nasledujúcich dňoch postupne klesá, až dosiahne nulovú hodnotu.
Plán kontroly nedokončených úloh	Plán postupnosti kontrol s jednotlivými členmi tímu.

4.4. Procesy prípravy ukončenia iterácie

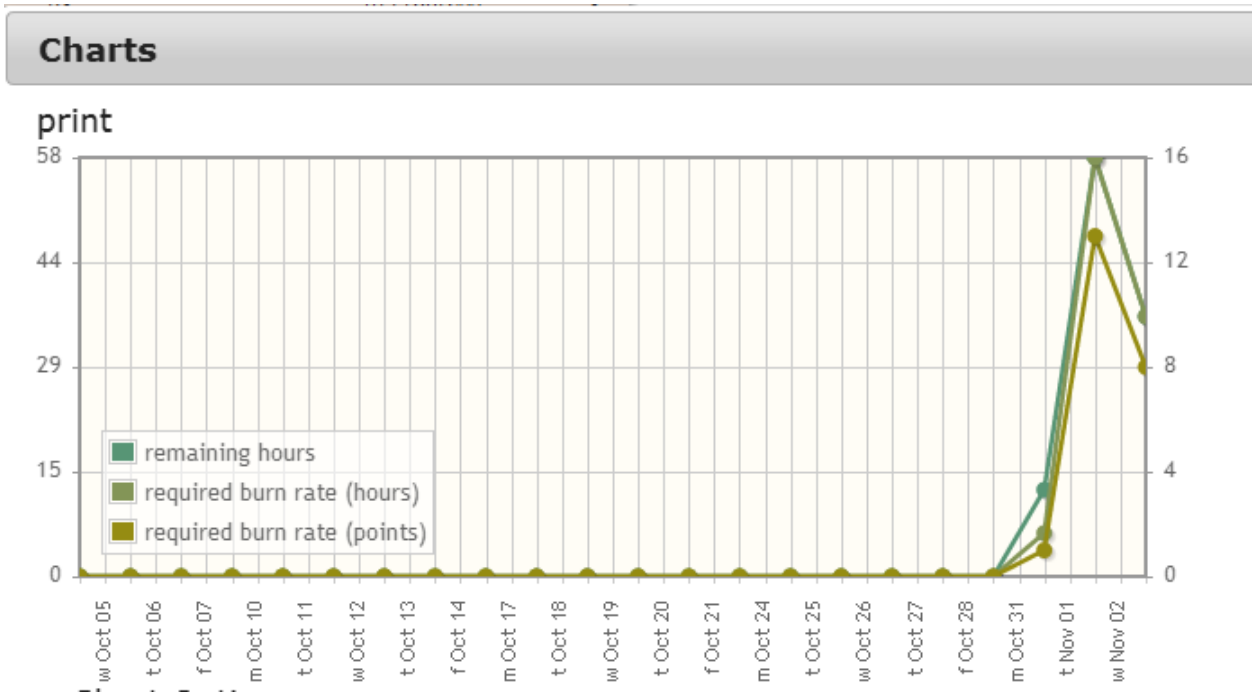
Procesy číslo 1. až 3. nasledujú lineárne za sebou. Proces číslo 4. prebieha medzi vedúcim tímu a každým vlastníkom nedokončenej úlohy samostatne.

Číslo	Proces	Kapitola
1.	Kontrola grafu priebehu iterácie	4.4.1
2.	Kontrola stavu backlogu iterácie	4.4.2
3.	Naplánovanie kontroly nedokončených úloh s ich vlastníkmi	4.4.3
4.	Kontrola nedokončených úloh s ich vlastníkmi	4.4.4
5.	Ukončenie iterácie v systéme Redmine	4.4.5

4.4.1. Kontrola grafu priebehu iterácie

Vstup: Graf priebehu iterácie
Výstup: Zostatok nedokončených hodín iterácie
Zodpovedný: Vedúci tímu

Vedúci tímu v systéme Redmine v karte *Backlogs* vyberie z kontextového menu pre aktuálnu iteráciu položku *Burndown*, ktorá zobrazí graf priebehu iterácie. Vedúci tímu vizuálne skontroluje graf a určí zostatok nedokončených hodín práce na úlohách a získa tak prvý pohľad na stav úloh pred ukončením iterácie.

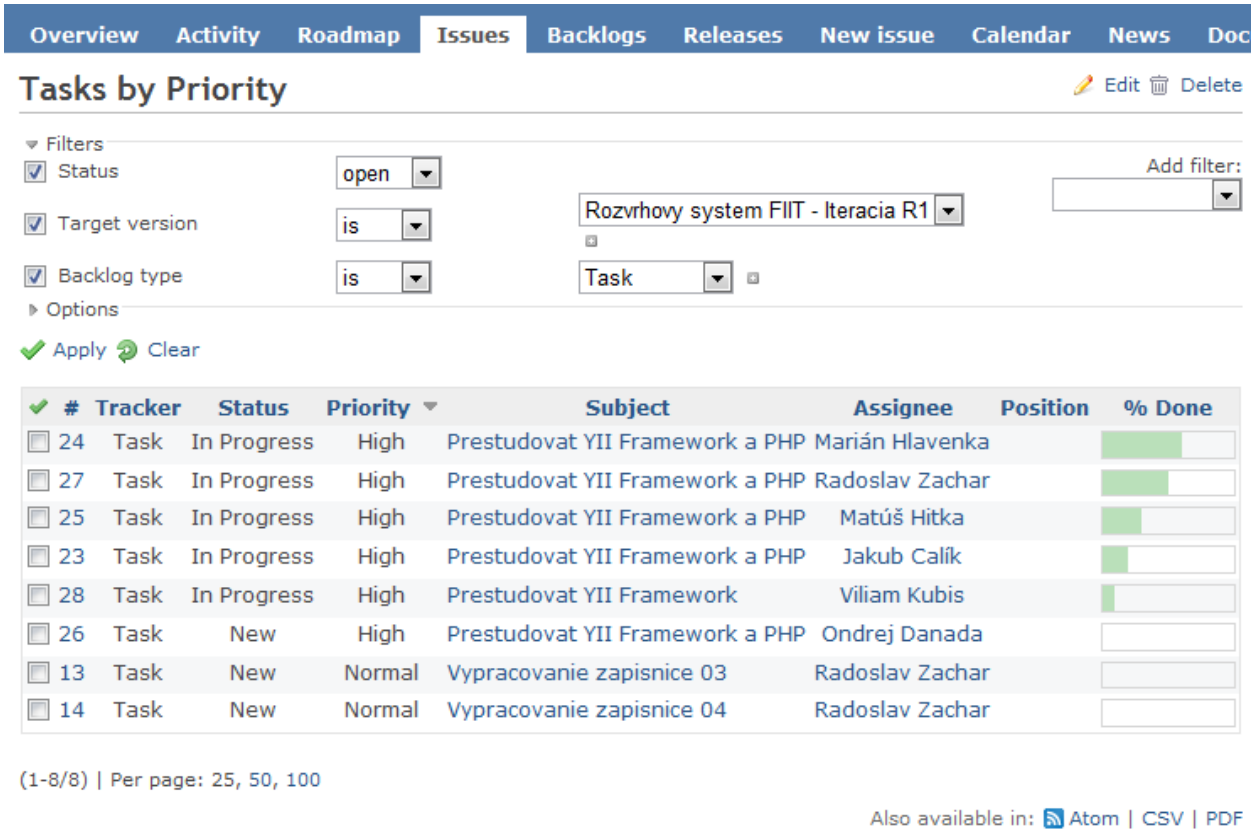


Obrázok 1. Príklad grafu priebehu iterácie v systéme Redmine

4.4.2. Kontrola stavu backlogu iterácie

Vstup: Zostatok nedokončených hodín iterácie, Backlog iterácie
Výstup: Zoznam nedokončených úloh
Zodpovedný: Vedúci tímu

Vedúci tímu v systéme Redmine v karte *Issues*, výberom dotazu na nedokončené úlohy v rámci aktuálnej iterácie zobrazí zoznam nedokončených úloh. Zoznam je automaticky usporiadaný primárne podľa priority a sekundárne podľa stupňa ukončenia (% Done). Vedúci tímu získaný zoznam exportuje do PDF súboru.



Obrázok 2. Dotaz na nedokončené úlohy v systéme Redmine

4.4.3. Naplánovanie kontroly nedokončených úloh s ich vlastníkmi

Vstup: Zoznam nedokončených úloh
Výstup: Plán kontroly nedokončených úloh
Zodpovedný: Vedúci tímu

Vedúci tímu kontaktuje vlastníkov nedokončených úloh podľa zoznamu a oznamuje im čas kontroly im pridelenej nedokončenej úlohy. Podľa zoznamu sú ako prvé v zozname úlohy s najvyššou prioritou a s najvyšším stupňom dokončenia, aby čo najvyšší počet úloh mohol byť dokončený do konca iterácie.

Rozvrhovy system FIIT - Tasks by Priority

#	Tracker	Status	Priority	Subject	Assignee	% Done
24	Task	In Progress	High	Prestudovat YII Framework a PHP	Marián Hlavenka	60
27	Task	In Progress	High	Prestudovat YII Framework a PHP	Radoslav Zachar	50
25	Task	In Progress	High	Prestudovat YII Framework a PHP	Matúš Hitka	30
23	Task	In Progress	High	Prestudovat YII Framework a PHP	Jakub Calík	20
28	Task	In Progress	High	Prestudovat YII Framework	Viliam Kubis	10
26	Task	New	High	Prestudovat YII Framework a PHP	Ondrej Danada	0
13	Task	New	Normal	Vypracovanie zapisnice 03	Radoslav Zachar	0
14	Task	New	Normal	Vypracovanie zapisnice 04	Radoslav Zachar	0

Obrázok 3. Príklad exportu zoznamu nedokončených úloh v PDF formáte

4.4.4. Kontrola nedokončených úloh s ich vlastníkmi

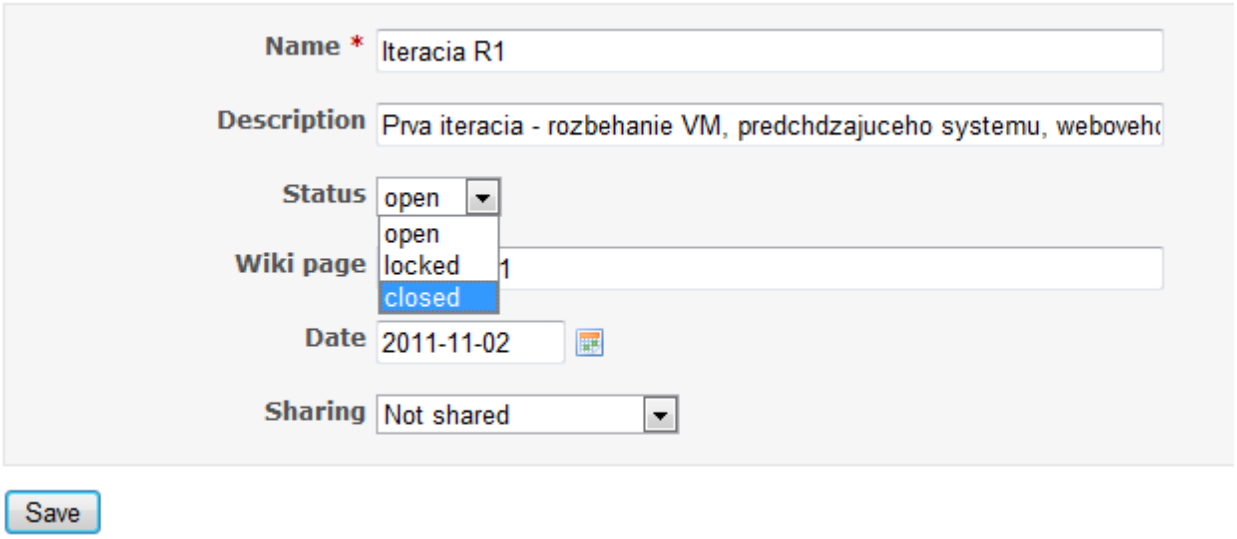
Vstup: Plán kontroly nedokončených úloh
Výstup: Upravený zoznam nedokončených úloh
Zodpovedný: Vedúci tímu

Vedúci tímu spolu s vlastníkom úlohy diskutujú o možnosti dokončenia nedokončenej úlohy v rámci aktuálnej iterácie, teda priamo v daný deň. Ak je táto možnosť z pohľadu oboch aktérov dostatočne prijateľná, vlastník úlohy sa pokúsi úlohu dokončiť do vedúcim stanoveného momentu ukončenia iterácie. Ak je z diskusie jasné, že úloha nemôže byť dokončená v daný deň, vedúci tímu si úlohu označí v zozname nedokončených úloh, aby bola presunutá do nasledujúcej iterácie. Vlastník úlohy naďalej pokračuje v riešení úlohy, aby bola v rámci danej iterácie dokončená čo najväčšia časť úlohy.

4.5. Ukončenie iterácie v systéme Redmine

Vstup: Upravený zoznam nedokončených úloh
Výstup: Ukončená iterácia, Upravený zoznam nedokončených úloh
Zodpovedný: Vedúci tímu

Vedúci tímu ukončuje iteráciu v stanovený deň ukončenia iterácie v čase po skončení práce všetkých členov tímu, ak neexistujú špeciálne okolnosti, v čase okolo 20:00. Všetci vlastníci nedokončených úloh dostali dostatočný čas na dokončenie svojich úloh. Vedúci tímu v tomto momente označí všetky nedokončené úlohy v zozname nedokončených úloh, aby boli presunuté do nasledujúcej iterácie. Vedúci tímu následne v systéme Redmine v karte *Settings* pre daný projekt vyberie možnosť editovať aktuálnu iteráciu, v ktorej zmení jej *Status* z hodnoty *open* na *closed* a ukončí tak aktuálnu iteráciu.



Obrázok 4. Ukončenie iterácie v systéme Redmine

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
Fakulta informatiky a informačných technológií

Tvorba dokumentácie a proces dokumentácie zdrojových kódov

(Metodika)

Autor: **Bc. Radoslav Zachar**

Predmet: Manažment softvérového inžinierstva
Študijný odbor: Softvérové inžinierstvo
Ročník: 1. Ing.
Akademicky rok: 2011/2012

Obsah

1. Úvod

Metodika je formálny, technický a normatívny dokument. Určuje poradie vykonávaných procesov a role a zodpovednosti pracovníkov v každom z týchto procesov. Pre každý proces presne definuje vstupy a výstupy. Stanovuje pravidlá a určuje podmienky a faktory pre rozhodovanie. Metodika je vecná, používa stručné a pomerne jednoduché vety.

Zachytáva skúsenosti a znalosti odborníka danej oblasti a napomáha tak realizácii činnosti a tvorbe produktu aj menej skúseným pracovníkom.

1.1. Účel metodiky

Účelom tejto metodiky je stanoviť jednotný štýl zápisu všetkých komentárov v zdrojových kódoch vyvíjaného softvérového produktu.

Tento dokument je určený pre členov vývojového tímu, ktorí píšu zdrojové kódy. Zmyslom metodiky je povinnosť každého člena tímu dodržiavať nižšie uvedené pravidlá.

2. Role a zodpovednosti v procese dokumentovania

Nevyhnutnou súčasťou každej metodiky je definícia rolí v opisovanom procese. Každá rola je dôležitá a má určitú zodpovednosť. Role sa určujú členom tímu.

V procese dokumentovania definujeme tieto základné role:

1. Dokumentátor
    - Vytvára dokument
    - Určuje štruktúru a štýl písania dokumentu
2. Analytik
    - Analyzuje riešený problém
    - Opisuje problémovú oblasť
    - Dodáva ostatným zúčastnením základné informácie k produktu
3. Korektor
    - Kontroluje a opravuje dokumentáciu
4. Zákazník
    - Nepriamo ovplyvňuje dokument
    - Dáva podnet k vytvoreniu produktu

- Dodáva zakladaciu listinu
- Podieľa sa na špecifikácii požiadaviek

5. Manažér kvality
    - zodpovedná za správne komentovanie zdrojového kódu
    - kontroluje zdrojový kód aj dokument
    - napomína a dáva impulzy ostatným zúčastnením k správnemu plneniu plánu a vytváraniu produktu
6. Hlavný programátor
    - Nesie najväčšiu zodpovednosť za spôsob komentovania zdrojového kódu
    - Vytvára základnú logiku systému
    - Vytvára a hlavne komentuje zdrojový kód
7. Programátor
    - Vytvára a komentuje zdrojový kód

3. Artefakt

Za artefakt v našom procese považujeme samotnú dokumentáciu. Celý výsledný dokument tvorí artefakt, ktorého vývoj predstavoval ukončenie niekoľkých procesov.

Základné procesy pre správne vytvorenie artefaktu sú *vytvorenie dokumentu*, *naplnenie dokumentu dátami*, *kontrola vytvoreného dokumentu*, *zmeny v dokumente* a neposlednom rade aj *udržovanie tohto dokumentu*.

4. Dokumentácia zdrojových kódov

Dôležitou súčasťou dokumentácie je aj dokumentácia k zdrojovému kódu vytváraného softvérového produktu.

Ešte pred začatím programovania je dôležité mať presne stanovený spôsob komentovania a vytvárania zdrojového kódu. Tento spôsob je jednotný pre každého vývojára a je povinnosťou ho dodržovať!

5. Komentovanie zdrojového kódu

Proces komentovania zdrojového kódu je dôležitý proces, ktorý je možné rozdeliť do viacerých podprocesov zoradených striktným poradím vykonávania.

Komentáre je potrebné písať členmi vývojového tímu ešte pred samotným začatím implementácie. Jednotlivé komentáre opisujúce funkcionality a význam prvkov v súbore by nemali byť dlhšie ako jeden riadok. Naopak komentáre opisujúce samotnú logiku a správanie jednotlivých metód alebo tried by mali byť dlhšie.

Proces komentovania zdrojového kódu delíme na nasledujúce podprocesy:

1. Vytvorenie bloku pre komentár
2. Jednoduché opísanie súboru vo vytvorenom bloku
3. Opísanie vlastností tried
4. Zapísanie značiek pre triedy
5. Opísanie metód
6. Zapísanie značiek pre metódy

5.1. Vytvorenie bloku pre komentár

Prvý podproces predstavuje vytvorenie bloku, do ktorého sa vloží základný komentár. Tento blok je dôležitý preto, aby v ňom vpísaný komentár bolo možné zobraziť vo výslednej dokumentácii.

Vstupy do procesu: akýkoľvek súbor pripravený pre zdrojový kód.
Výstup z procesu: súbor s blokom textu pripraveným na komentovanie.
Zodpovedná osoba: hlavný programátor / programátor.
Popis: programátor vytvorí súbor a následne do prvých riadkov tohto súboru vloží potrebný blok textu

5.2. Jednoduché opísanie súboru vo vytvorenom bloku

Opis súboru sa relizuje vložením krátkeho a výstižného komentára vysvetľujúceho význam daného súboru. Tento komentár sa vkladá do predpripraveného bloku, ktorý už súbor musí obsahovať. Komentár opisujúci súbor začína vždy veľkým písmenom a je dostatočne stručný.

Vstupy do procesu: súbor s predpripraveným blokom textu pripraveným na komentovanie
Výstup z procesu: vytvorený jednoduchý opis významu súboru vložený do predpripraveného bloku
Zodpovedná osoba: programátor
Popis: programátor do existujúceho elementu vpíše komentár, ktorý je výstižný, zrozumiteľný a dostatočne opisuje úlohu súboru v projekte

5.3. Opísanie vlastností tried

Každá trieda musí obsahovať popis, v ktorom sú jasne opísané všetky jej vlastnosti. Opis funkcionality, ktorú trieda zabezpečuje by mal byť pochopiteľný každému prípadnému čitateľovi.

Vstupy do procesu: súbor s predpripraveným blokom textu pripraveným na komentovanie
Výstup z procesu: vytvorený jednoduchý opis funkcionality, ktorú zabezpečuje trieda obsiahnutá v súbore
Zodpovedná osoba: programátor
Popis: programátor v skratke opíše použitú triedu, jej význam, dôležitosť a funkcionalitu

5.4. Zapísanie značiek pre triedy

Vloženie povinných značiek pre každú triedu je jeden z najdôležitejších procesov. Vytvorením týchto značiek určite zjednotíme výslednú dokumentáciu. Značky obsahujú údaje ako meno autora triedy, kategóriu zaraďujúcu opisovanú triedy a názvy balíkov, v ktorých sa trieda nachádza.

Vstupy do procesu: súbor s blokom textu pripraveným na komentovanie
Výstup z procesu: vytvorený podrobný opis triedy
Zodpovedná osoba: programátor
Popis: programátor pripíše k základnému opisu triedy povinné vlastnosti, ktoré sú: názov balíka a podbalíka, v ktorom sa trieda nachádza, kategóriu do ktorej patrí a názov autora triedy (nie súboru!)

5.5. Opísanie metód

Pre každú definovanú metódu v triede je potrebné vytvoriť jednoduchý komentár opisujúci jej význam.

Vstupy do procesu: súbor s blokom textu pripraveným na komentovanie
Výstup z procesu: vytvorený opis všetkých metód opisovanej triedy
Zodpovedná osoba: programátor
Popis: programátor do existujúceho bloku postupne opíše funkciu každej metódy

5.6. Zapísanie značiek pre metódy

Okrem opísania významu metódy je nutné pridať ku komentárom aj povinné značky pre metódu.

Vstupy do procesu: súbor s blokom textu pripraveným na komentovanie

Výstup z procesu: vytvorený podrobný opis metód, ktoré obsahuje trieda v súbore
Zodpovedná osoba: programátor
Popis: programátor pripíše k opisu metódy povinné informácie, ktoré sú: typ prístupu, vstupné parametre a návratovú hodnotu metódy

6. Dokumentovanie zdrojových kódov v PHPDoc

PHPDoc je formálny štandard pre komentovanie zdrojového kódu napísaného v programovacom jazyku PHP.

Tento štandard je definovaný presne určenými značkami, pričom každá má svoj špecifický význam. Predpísaný spôsob komentovania dovoľuje externým generátorom vytvoriť technickú dokumentáciu. Externé programy na tvorbu dokumentácie čítajú špeciálne komentáre umiestnené v základnom bloku nazývanom DocBlock.

Pre proces vytvárania komentárov v jazyku PHPDoc definujeme nasledujúce procesy:

1. Vytvorenie predpísaného komentárového bloku
2. Opísanie významu a obsahu súboru
3. Opísanie vlastností súboru
4. Dokumentácia tried
5. Dokumentácia metód

6.1. Predpísané značky

Značky v štandarde PHPDoc začínajú znakom " @ ". Všetky značky predstavujú jednoduché anglické slová, ktoré informujú generátor technickej dokumentácie ako naformátovať a interpretovať komentár. Používanie značiek vo všeobecnosti nie je povinné, ale ich použitie výrazne prispeje k prehľadnosti výslednej dokumentácie.

Medzi základné značky patria:

- @abstract – abstraktná trieda, premenná alebo metóda
- @access – typ prístupu (public/private/protected)
- @author – meno autora
- @copyright – copyright informácie
- @example – cesta k súboru s príkladom
- @global – globálne premenné
- @internal – špeciálne poznámky pre vývojárov
- @package – názov balíka

- @subpackage – naázov podbalíka
- @param – opis vstupných paramentrov
- @return – typ a opis výstupu
- @since – datum vloženia elementu do projektu
- @todo – úlohy pre programátorov na neskorší vývoj
- @version – číslo verzie elementu

6.2. Vytvorenie predpísaného komentárového bloku

Pri písaní každého komentára musí programátor dodržovať presne daný formát. Každý komentár musí byť v prvom rade umiestnený v špeciálnom bloku, ktorý začína samostantým riadkom obsahujúcim“ /** ” a končí samostatným riadkom obsahujúcim “ **/ ”. Každý komentár vo vnútri tohto bloku musí začínať znakom “ * ”. Riadok komentáru, ktorý nezačína znakom “ * ” je vo výsledne vygenerovanej dokumentácii ignorovaný. Celý opísaný komentárový blok musí byť umiestnený na začiatku súboru (v prvých jeho riadkoch).

Príklad povinne vytvoreného bloku pre jednotlivé komentáre:

```
/**
  * komentar 1
  * komentar 2
  * ...
 **/
```

6.3. Opísanie významu a obsahu súboru

Je povinnosťou každého programátora vytvoriť komentáre pri vytváraní nového php súboru patriaceho k projektu.

Prvý komentár zobrazuje meno súboru. Druhým komentárom opisujeme význam a obsah súboru v jednej jednoduchej vete. Nasledujúci komentár obsahuje detailnejší opis súboru v niekoľkých vetách.

Príklad komentárov opisujúcich súbor:

```
...
* meno_suboru.php
* jednoduchy opis suboru
* podrobny opis suboru
...
```

6.4. Opísanie vlastností súboru

Okrem opisu súboru je treba vložiť aj informácie opisujúce vlastnosti vytvoreného súboru. Všetky komentáre obsahujúce vlastnosti súboru musia začínať

značkou. Tieto vlastnosti zahŕňajú minimálne kategóriu, do ktorej súbor patrí, meno autora, číslo verzie súboru a dátum jeho vytvorenia.

Príklad komentárov opisujúcich súbor:

```
...
* @category      kategoria
* @package       balik
* @copyright     meno autora
* @version       verzia suboru
* @since datum   vytvorenia
...
```

6.5. Dokumentácia tried

Každá trieda musí obsahovať jednoduchý a povinne aj podrobný komentár vystihujúci jej význam. Ďalej by mali byť opísané aj jej vlastnosti značkami.

Príklad komentárov opisujúcich triedy:

```
...
* jednoduchy opis triedy
* podrobny opis triedy
* @category
* @package
* @author
...
```

6.6. Dokumentácia metód

Podobne ako pri komentovaní tried, aj metóda musí obsahovať jednoduchý a povinne aj podrobný komentár vystihujúci jej činnosť alebo logiku. Ďalej by mali byť opísané aj jej vlastnosti značkami.

Príklad komentárov opisujúcich metódy:

```
...
* jednoduchy opis metody
* podrobny opis metody
* @author
* @param
* @return
...
```

Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológií

Manažment testovania a tvorba unit testov

(Metodika)

Autor: **Jakub Calík**

Predmet: Manažment projektov softvérových a informačných systémov
Odbor: Softvérové inžinierstvo
Ročník: 1. Ing.
Akademický rok: 2011/2012

1 Obsah

2 Úvod

Účelom tejto metodiky je najskôr na vyššej úrovni definovať celkový postup testovania produktu a následne definovať postup vytvárania unit testov v PHP frameworku Yii. Metodika je určená hlavne pre malé tímy vyvíjajúce softvér v PHP frameworku Yii.

3 Kľúčové slová

Yii – framework pre tvorbu aplikácií v jazyku PHP

PHPUnit – framework pre písanie jednotkových (unit) testov pre programovací jazyk PHP, poskytujúci možnosť spúšťať unit testy a analyzovať získané výsledky

Selenium RC (Remote Control) – testovací nástroj, ktorý umožňuje písať automatizované unit testy pre webové aplikácie napísané v ľubovoľnom programovacom jazyku cez HTTP, využívajúc ľubovoľný webový prehliadač s povoleným JavaScriptom.

4 Role a zodpovednosti

Softvérový návrhár

- návrh architektúry systému
- definovanie technických požiadaviek na systém
- návrh akceptačných testov pre navrhovanú logiku

Vývojár

- programovanie samotnej funkcionality systému

Tvorca unit testov

- Tvorba jednotkových (unit) testov (často samotný vývojár)
- Otestovanie funkcionality voči akceptačným testom
- Hlásenie nezhôd naprogramovanej funkcionality s príslušnými akceptačným testom

Tester

- Integrácia jednotkových testov do skriptu na dávkové spúšťanie testov
- Zbehnutie jednotkových testov
- Zdokumentovanie výsledkov

Manažér kvality

- Analýza výsledkov správ z testov vykonávaných testermi
- Hlásenie nezhôd naprogramovanej funkcionality s požiadavkami

Systémový administrátor

- Inštalácia a konfigurácia vyžadovaných nástrojov pre testovanie

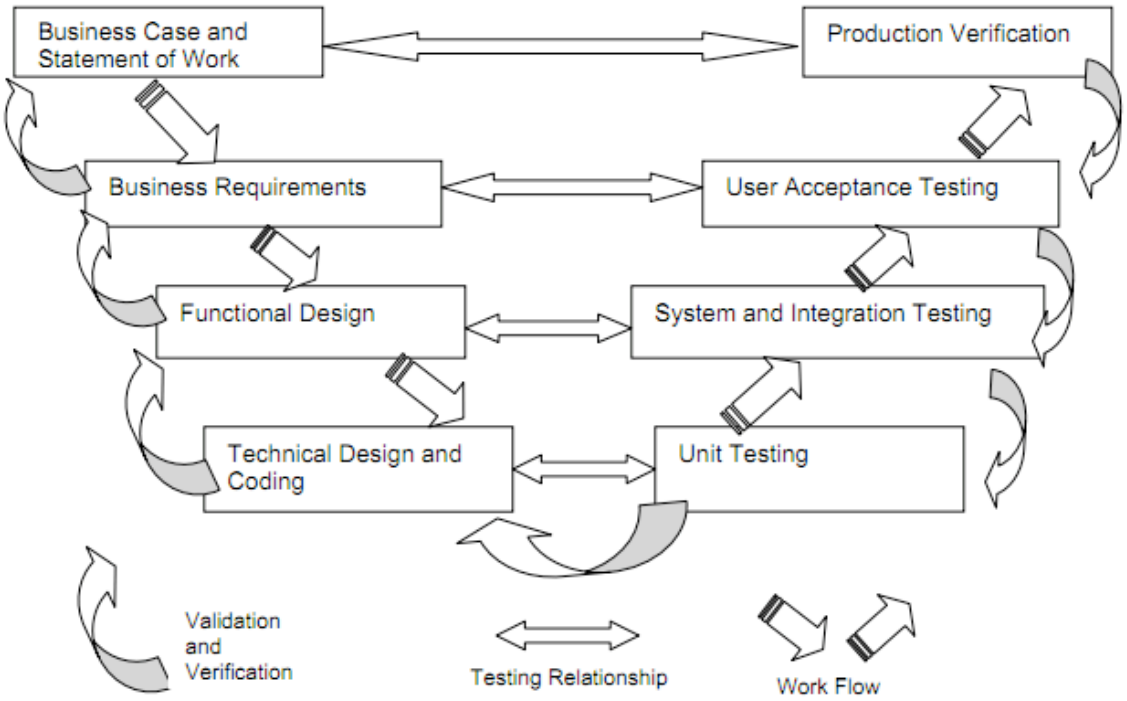
- Zaisťuje dostupnosť všetkých testovacích zariadení v súlade s harmonogramom projektu
- Monitoruje výkon systému počas záťažových a pamäťových testov

Jedna osoba môže zastávať aj viacero rolí, ale býva pravidlom, že tvorca jednotkových testov, ktorý vykonáva aj akceptačné testy nesmie byť tá istá osoba ako vývojár príslušnej testovanej funkcionality.

5 Procesy pri testovaní

5.1 Testovanie v procese vývoja softvéru

Najjednoduchší spôsob testovania softvéru je jeho opakované spustenie a zadávanie rôznych vstupných dát s cieľom preveriť čo najväčší počet možných kombinácií a používateľských situácií. Tento prístup je však pri testovaní nedostačujúci. Testovanie je nutné chápať ako skupinu činností vzťahujúcich sa k jednotlivým etapám procesu vývoja softvéru.



Obrázok 1: *V-model testovania softvéru*

Štandardný proces vývoja softvéru postupuje najčastejšie podľa V-modelu (*Obrázok 1*). Základnými aktivitami realizovanými v procese testovania softvéru sú revízia a inšpekcia požiadaviek a špecifikácií. Ďalej to sú identifikácia testovacích požiadaviek, návrh, vykonanie a vyhodnotenie testov.

5.2 Revízia a inšpekcia požiadaviek a špecifikácií softvéru

Revíziám podlieha všetka dokumentácia, ktorá popisuje vyvíjaný systém. Cieľom revízie je odhaliť chyby v návrhu softvéru a doplniť prípadné nejasnosti. Zároveň je nutné už v počiatočných fázach vývoja vniesť do návrhu požiadavky na budúcu možnosť testovať softvér. Taktiež je nevyhnutné skontrolovať, či je možné existujúce požiadavky podrobiť testu. Formulácie typu napr. „hodnota parametra je pravidelne aktualizovaná“ je príliš nejednoznačná a pod pojmom „pravidelná aktualizácia“ sa môže skrývať veľké množstvo definícií.

Pri návrhu sa tiež musí uvažovať aj o budúcej automatizácii testovania a navrhnúť príslušné rozhranie pre testovacie nástroje. Prispôsobenie softvéru je z hľadiska vývoja časovo náročná práca, s ktorou je nutné počítať pri plánovaní celkovej doby trvania projektu.

5.3 Identifikácia testovacích požiadaviek

Proces identifikácie testovacích požiadaviek spočíva vo výbere všetkých požiadaviek, ktoré súvisia s predmetom testu. Ide predovšetkým o výber podmnožiny z vyšpecifikovaných systémových a softvérových požiadaviek a o definovanie nových požiadaviek na základe analýzy dokumentácie návrhu.

5.4 Návrh testov

Návrh testov zahŕňa špecifikáciu testovacieho zariadenia a definíciu jednotlivých testovacích postupov. Navrhnuté testovacie postupy musia overiť všetky identifikované testovacie požiadavky. Kontrolu použitých postupov a kontrolu testovaných požiadaviek vykonáva manažér kvality.

5.5 Vykonanie testov a regresné testovanie

Vykonávanie testov uskutočňuje spravidla tester. Pri vykonávaní testov je predovšetkým nevyhnutné podrobné dokumentovanie vykonania každého jednotlivého kroku každého z postupov. Dokumentovanie sa musí spraviť tak, aby bolo možné testy v prípade potreby presne zreprodukovať.

Regresné testovanie predstavuje opakovanie všetkých testov, ktorých predmet bol upravený alebo zmenený za účelom odstránenia chýb detekovaných behom predchádzajúceho testovania. Práve vzhľadom k časovej náročnosti regresných testov, a to hlavne v záverečných etapách projektu, je potrebné zautomatizovať testovanie softvéru.

5.6 Vyhodnotenie testov

Dôležitou súčasťou testovania je vyhodnotenie testov. Testy sa vyhodnocujú na základe rôznych meraní, ktoré sú vždy definované počas plánovacej fázy projektu. Často sa využíva kombinácia súhrnných kombinovaných meraní (napr. podiel chýb na celkovom počte vykonávaných testov a podiel chýb v rámci jednotlivých komponent systému). Z hľadiska zhodnotenia úspešnosti testov je dôležité sledovať pomer chýb nájdených počas vykonávania naplánovaných testov a chýb nájdených pri náhodných neplánovaných testoch. Vyhodnocovanie testov taktiež vykonáva tester. Následne odovzdáva správu manažérovi kvality o úspešnosti/neúspešnosti testov. Manažér kvality dodatočne vyhodnocuje správy od testerov a navrhuje postup, ako vyriešiť objavené nedostatky systému.

5.7 Stratégie testovania

Testovacia stratégia je systematická metóda použitá na výber alebo generovanie testu, ktorý bude zahrnutý do skupiny testov. Testovacia stratégia definuje pravidlá, podľa ktorých je možné objektívne posúdiť, či daný test vyhovuje vybranej stratégii. Stratégiu ide taktiež zautomatizovať.
Testovacie stratégie sa v zásade delia na:

- Funkčné (black-box) testy: návrh testov vychádza z požiadaviek na softvér a popis jeho funkcií, testy na základe tejto stratégie môžu byť navrhnuté bez znalosti testovaného objektu
- Štrukturálne (white-box) testy: sú odvodené priamo zo znalosti štruktúry testovaného objektu, pod štruktúrou je potrebné rozumieť architektúru softvéru a spôsob jeho implementácie
- Hybridné testy: kombinácia funkčných a štrukturálnych testov

5.8 Typy testov

Testy softvéru sú rozdelené do 4 základných etáp podľa V-modelu (*Obrázok1*):

- **Unit testy**, ktoré bývajú často zahrňované do fázy vývoja. Uskutočňujú ich priamo vývojári v rámci ladenia kódu. Cieľom týchto testov je overiť funkčnú schopnosť izolovaných modulov programu na základe overenia všetkých funkcií vstupno-výstupného rozhrania. Podrobný popis tvorby a vykonávania unit testov pomocou PHPUnit v Yii frameworku je v kapitole 6.
- **Integračné testy**, určené k overeniu správnosti spolupráce integrovaných modulov na základe dokumentácie návrhu. Obvykle vyžadujú simuláciu činnosti nehodových modulov, prípadne vytváranie špeciálnych ovládačov pre overenie komunikácie medzi modulmi. Jednotlivé moduly sa pri týchto testoch považujú za uzavreté systémy. Z hľadiska modulov teda ide skôr o funkčné testy, zatiaľ čo z hľadiska celého programu ide skôr o hybridné testy.
- **Systémové testy**, ktoré overujú činnosť systému ako celku. Typicky to sú

predovšetkým záťažové testy, testy kontrolujúce prideľovanie operačnej pamäte a funkčné testy pri rôznych konfiguráciách systému.

- **Akceptačné testy**, ktoré zaisťujú kompletné overenie všetkých systémových funkcií garantovaných zákazníkovi. Slúžia ako výstupná kontrola výsledného produktu.

5.9 Automatizácia testov

Proces návrhu a vykonávania testov softvéru je väčšinou časovo obmedzený v rámci časového harmonogramu projektu. Opakovateľnosť testov a dôkladnosť ich vykonávania taktiež hrajú dôležitú úlohu, no z hľadiska termínu dokončenia projektu je rozhodujúci hlavne počet a časová náročnosť testov. Pre otestovanie väčších softvérových projektov je niekedy nevyhnutné vykonať aj niekoľko tisíc testov. Ručné otestovanie celej aplikácie tak zaberie aj niekoľko mesiacov. Keďže testy je, po každej zmene kódu, potrebné zopakovať, automatizácia testov je za týchto podmienok nevyhnutná.

Automatizáciou sa rozumie použiť vhodný testovací nástroj, ktorý je schopný ovládať testovanú aplikáciu tak, ako by to pri jej testovaní robil človek. Testovací nástroj musí umožniť vytvárať testovacie aplikácie. Preto je najčastejšie napísaný v skriptovacom jazyku (napr. Basic).

Bežným typom funkčných testov sú testu uskutočňované pomocou štandardného používateľského rozhrania testovanej aplikácie, najčastejšie teda pomocou grafického používateľského rozhrania. Pre tento typ testov je nevyhnutné použiť niektorý zo špecializovaných, komerčne dostupných nástrojov, ako napr. ATF (Softbridge), WinRunner (Mercury Interactive), Rational Robot (Rational).

Pre neštandardnejšie rozhrania je nutné vytvoriť vlastný testovací nástroj, ktorý musí byť vyvíjaný spoločne s testovanou aplikáciou. Testovací nástroj sa tak stáva súčasťou vyvíjanej aplikácie a musí byť navrhnutý už v začiatočnej fáze návrhu testovaného softvéru.

6 Proces tvorby a realizácie unit testov v PHPUnit

Testovací framework v Yii je postavený na PHPUnit frameworku. Preto budú v tejto kapitole popísané postupy vytvárania unit testov v PHPUnit.

Cieľom unit testov je izolovať všetky časti programu a preukázať, že samostatné časti programu sú správne a bez chýb. Unit testy navrhujú a píšu testeri. V niektorých prípadoch testerom môže byť aj samotný vývojár, no potom nesmie testovať modul, ktorý sám napísal.

Postup krokov pri unit testovaní softvéru:

1. Vytvorenie testovacieho plánu
2. Vytvorenie prípadu testovania a testovacích dát
3. Ak je prípad použiteľný, vytvorenie testovacieho skriptu na vykonanie testu
4. Vykonanie navrhnutého testu
5. Oprava chýb, ktoré sa počas testu vyskytli a znovuvykonanie testu
6. Opakovanie cyklu, až kým nie je testovaný modul zbavený všetkých chýb

6.1 Vytvorenie prípadu testovania

Prípad testovania (Test Case) presne popisuje ako by mal byť navrhovaný test vykonaný.

Prípad testovania by mal obsahovať nasledujúce prvky:

1. ID testovacieho prípadu – identifikátor pre správne spracovanie a zaradenie testovacieho prípadu do plánu testovania
2. Popis – popis samotného testovacieho prípadu (čo sa bude testovať)
3. Vstupné dáta – vstupné parametre testu (testované triedy, atribúty)
4. Očakávaný výsledok – popis výsledku (ako by mal test dopadnúť, aké hodnoty sa očakávajú)
5. Skutočný výsledok – popis výsledku vykonania testu (skutočné výsledné hodnoty)
6. Zhodnotenie – označenie, či testovaný modul splnil požiadavky (prešiel/neprešiel)
7. Dodatočné informácie – dátum testu, iterácia testu, verzia testovaného modulu

6.2 Vytvorenie unit testu

Unit test sa v PHPUnit frameworku píše v rámci triedy *MenoTriedyTest*, ktorá rozširuje *CTestCase* alebo *CDbTestCase*. *MenoTriedy* je testovaná trieda. *CtestCase* sa používa na otestovanie štandardných tried, zatiaľ čo *CDbTestCase* sa používa na otestovanie tried, ktoré pracujú s databázou (obsahujú štandardné SQL príkazy, ako *create, update, insert, delete*).

Vytvorená testovacia trieda sa uloží ako PHP súbor *MenoTriedyTest.php*, štandardne do priečinka *protected/tests/unit*.

Testovacia trieda môže (nie nevyhnutne) obsahovať sadu testovaných metód. Tieto metódy sa štandardne nazývajú *testMenoMetody*, kde *MenoMetody* je zodpovedajúca metóda v triede, ktorá sa bude testovať. V testovacej metóde sa môžu nachádzať príkazy *assertTrue, assertEquals,* ktoré slúžia ako záchytné body pri validácií a vyhodnocovaní.

Príklad:

Máme triedu *PageController*, ktorá dve jednoduché metódy *repeat* a *concat*:

```
class PageController {
        public function repeat($inputString){
                return $inputString;
        }
        public function concat($string1,$string2){
                return $concat=$string1.''.$string2;
        }
}
```

Potrebujeme otestovať, či sa funkcie repeate a concat správajú správne. Vytvoríme triedu *PageController*:

```
class PageControllerTest extends CTestCase {
        public function testRepeat() {
                $mypage = new PageController();
                $yell = "hello any body out there";
                $returnedMessage=$mypage->repeat($yell);
                $this->assertEquals($returnedMessage, $yell);
        }
        public function testConcat() {
                $mypage = new PageController();
                $answer = $mypage->concat('joe','blogs');
                $this->assertEquals($answer, 'joe blogs');
        }
}
```

6.3 Vykonanie navrhnutého testu

Samotné vykonanie unit testu, vytvoreného v PHPUnit, spočíva v spustení vytvorených testovacích metód a pozorovania výsledkov.

Postup krokov:

1. Vytvorenie inštancie testovacej triedy (*PageControllerTest*)
2. Postupné zavolanie všetkých testovaných metód
3. Pozorovanie a podrobné zdokumentovanie výsledkov testu

12 Príloha D (Zápisnice zo stretnutí)

Zápisnica z tímového stretnutia č. 1

Dátum:	5.10.2011	
Čas:	11:00 – 12:00	
Miesto:	Softvérové štúdio FIIT STU v BA	
Účastníci:	Vedúci pedagóg:	Ing. Miroslav Galbavý
	Členovia tímu:	Bc. Jakub Calík
		Bc. Ondrej Danada
		~~Bc. Matúš Hitka~~
		Bc. Marián Hlavenka
		~~Bc. Viliam Kubis~~
		~~Bc. Pavol Škvarenina~~
		Bc. Radoslav Zachar
Zapisnicu	Vypracoval:	Bc. Radoslav Zachar
	Overil:	Bc. Marián Hlavenka

Téma stretnutia:
Nadviazanie na projekt minuloročného tímu.

Priebeh stretnutia:
Ing. Miroslav Galbavý nás bližšie oboznámil s témou projektu, zbežne vysvetlil, čo sa od nás očakáva a čo spravili minuloročné tímy.

Poznámky:

- oboznámenie sa s problematikou tvorby rozvrhov pre našu fakultu,
- posktytnutie mailing-listu vedúcemu: tim6@danada.sk,
- prevzatie elektronickej formy kompletnej dokumentácie k existujúcemu riešeniu minuloročného tímu (rozoslane mailom),
- zorganizovanie stretnutia s členmi minuloročného tímu v čase budúceho stretnutia, t.j. 12.10.2011 o 11:00.

Udelené úlohy:

- Úloha 1.1
  Preštudovanie obdržanej dokumentácie
  Zodpovednosť: Bc. Jakub Calík
  Termín splnenia: 12.10.2011

- Úloha 1.2
  Sprístupnenie a konfigurácia prideleného virtuálneho stroja
  Zodpovednosť: Bc. Viliam Kubis
  Termín splnenia: 12.10.2011

- Úloha 1.3
  Inštalácia a nastavenie web servera
  Zodpovednosť: Bc. Viliam Kubis
  Termín splnenia: 12.10.2011

- Úloha 1.4
  Vytvorenie základnej html prezenntácie tímu
  Zodpovednosť: Bc. Jakub Calík
  Termín splnenia: 12.10.2011

Zápisnica z tímového stretnutia č. 2

Dátum:	12.10.2011	
Čas:	11:00 – 13:00	
Miesto:	Softvérové štúdio FIIT STU v BA	
Účastníci:	Vedúci pedagóg:	Ing. Miroslav Galbavý
	Členovia tímu:	Bc. Jakub Calík Bc. Ondrej Danada Bc. Matúš Hitka Bc. Marián Hlavenka Bc. Viliam Kubis Bc. Pavol Škvarenina Bc. Radoslav Zachar
	Ďalší:	Predchádzajúci tím: Bc. Marcel Baláž, Bc. Mioslav Beno, Bc. Alojz Gomola, Bc. Peter Korenek, Bc. Ján Kováč, Bc. Ján Kvak, Bc. Roman Meszároš
Zapisnicu	Vypracoval:	Bc. Marián Hlavenka
	Overil:	Bc. Radoslav Zachar

Téma stretnutia:
Stretnutie s minuloročným tímom *Schedule of Pain* a oboznámenie sa s technológiami a dostupným systémom.

Priebeh stretnutia:
Stretnutie sa nieslo vo forme voľnej diskusie medzi členmi nášho a minuloročného tímu *Schedule of pain*. Objasnili nám aktuálny stav projektu, odporučili technológie na vývoj. Rozhodli sme sa použiť systém Redmine na manažovanie úloh v tíme.

Poznámky:

- zoznámenie sa s členmi minuloročného tímu
- výmena kontaktov medzi členmi tímov
- zoznámenie sa s odporúčanými a použitými technológiami: Redmine, NetBeans, Yii framework, C#, PSPad, Notepad++, XAMP, WAMP, Putty
- získanie jasnejšej predstavy o fungovaní hlavného algoritmu
- vysvetlenie dôvodov prechodu na Yii Framework
- obdržanie vytlačenej formy kompletnej dokumentácie celého rozpracovaného systému vrátane dátového nosiča obsahujúceho zdrojový kód systému a údaje potrebné pre vytvorenie a naplnenie databázy
- vytvorenie dlhodobého plánu projektu
- 3 iterácie za semester s označením: Iterácia I<poradové číslo iterácie>
- plán Iterácie I1
- plán iterácií
    - Iterácia I1 5.10 2011 - 2.11 2011
    - Iterácia I2 2.11 2011 - 30.11 2011
    - Iterácia I3 30.11 2011 - 21.12 2011

Zhodnotenie úloh:

- Úloha 1.1
  Preštudovanie obdržanej dokumentácie
  Zodpovednosť: Bc. Jakub Calík
  Úloha sa priebežne plní

- Úloha 1.2
  Sprístupnenie a konfigurácia prideleného virtuálneho stroja
  Zodpovednosť: Bc. Viliam Kubis
  Úloha splnená

- Úloha 1.3
  Inštalácia a nastavenie web servera
  Zodpovednosť: Bc. Viliam Kubis
  Úloha splnená

- Úloha 1.4
  Vytvorenie základnej html prezenntácie tímu
  Zodpovednosť: Bc. Jakub Calík
  Úloha splnená

Udelené úlohy:

- Úloha 2.1
  Nainštalovať systém Redmine
  Zodpovednosť: Bc. Viliam Kubis
  Termín splnenia: 19.10.2011

- Úloha 2.2
  Nakonfigurovať systém Redmine (systém, projekt, práva, úlohy)
  Zodpovednosť: Bc. Pavol Škvarenina
  Termín splnenia: 19.10.2011

- Úloha 2.3
  Príprava úloh v systéme Redmine
  Zodpovednosť: Bc. Pavol Škvarenina
  Termín splnenia: 19.10.2011

Zápisnica z tímového stretnutia č. 3

Dátum:	19.10.2011	
Čas:	11:00 – 13:00	
Miesto:	Softvérové štúdio FIIT STU v BA	
Účastníci:	Vedúci pedagóg:	Ing. Miroslav Galbavý
	Členovia tímu:	Bc. Jakub Calík
		Bc. Ondrej Danada
		Bc. Matúš Hitka
		Bc. Marián Hlavenka
		Bc. Viliam Kubis
		Bc. Pavol Škvarenina
		Bc. Radoslav Zachar
Zapisnicu	Vypracoval:	Bc. Radoslav Zachar
	Overil:	Bc. Ondrej Danada

Téma stretnutia:
Konzultovanie s vedúcim o smere, ktorým budeme viesť obdržaný systém.

Priebeh stretnutia:
Najskôr Ing. Miroslav Galbavý vysvetlil životný cyklus tvorby rozvrhov (zber požiadaviek od učiteľov, vytvorenie rozvrhu, registrovanie predmetov študentmi, následné riešenie vzniknutých kolízií). Taktiež zdôraznil prepojenie rozvrhov FIIT a FEI. Potom nasledovala diskusia v tíme. Zhrnuli sme, čo sme zistili ohľadom spomenutých použitých technológií z minulého stretnutia, konkrétne išlo o PHP jazyk, Framework Yii a Redmine. Zistili sme, že nainštalovaný systém Redmine úplne nevyhovuje našim potrebám a je treba k nemu doinštalovať plugin BackLogs pre potreby agilného plánovania. Koniec stretnutia sme venovali pokusu nahodiť obdržaný systém na VM, neúspešne.

Poznámky:
- Zoznamovanie sa s Yii framework-om, PHP jazykom a systémom RedMine
- konzultácia s vedúcim projektu a predstavy o budúcom stave systému
- konkrétne rozoberanie algoritmov a problémov pri tvorbe rozvrhov
- rozoberanie špecifík tvorby rozvrhov pre spojené FEI a FIIT predmety

Zhodnotenie úloh:
- Úloha 2.1
  Nainštalovať systém Redmine
  Zodpovednosť: Bc. Viliam Kubis
  Úloha splnená

- Úloha 2.2
  Nakonfigurovať systém Redmine
  Zodpovednosť: Bc. Pavol Škvarenina
  Úloha odložená do 26.10.2011

- Úloha 2.3
  Príprava úloh v systéme Redmine
  Zodpovednosť: Bc. Pavol Škvarenina
  Úloha odložená do 26.10.2011

Udelené úlohy:

- Úloha 3.1
  Zverejniť zápisnice zo stretnutia na webe
  Zodpovednosť: Bc. Radoslav Zachar
  Termín splnenia: 26.10.2011

- Úloha 3.2
  Študovať framework YII v rozsahu 16 hodín
  Zodpovednosť: Bc. Jakub Calík
  Termín splnenia: 2.11.2011

- Úloha 3.3
  Inštalácia plug-inu Backlogs pre systém Redmine
  Zodpovednosť: Bc. Viliam Kubis
  Termín splnenia: 26.10.2011

Zápisnica z tímového stretnutia č. 4

Dátum:	26.10.2011	
Čas:	11:00 – 12:30	
Miesto:	Softvérové štúdio FIIT STU v BA	
Účastníci:	Vedúci pedagóg:	Ing. Miroslav galbavý
	Členovia tímu:	Bc. Jakub Calík
		Bc. Ondrej Danada
		Bc. Matúš Hitka
		Bc. Marián Hlavenka
		Bc. Viliam Kubis
		Bc. Pavol Škvarenina
		Bc. Radoslav Zachar
Zapisnicu	Vypracoval:	Bc. Radoslav Zachar
	Overil:	Bc. Matúš Hitka

Téma stretnutia:
Spoločné dokončenie inštalácie potrebných systémov a dohodnutie sa na spôsobe písania zápisníc zo stretnutí.

Priebeh stretnutia:
Prvé tímové stretnutie, ktoré neviedol vedúci pedagóg, ale bolo riadené celým tímom. Na stretnutí sme sa dohodli na formálnom spôsobe písania zápisníc zo stretnutí. Vytvorili sme šablónu pre zápisnice. Ku koncu stretnutia sme riešili problémy s databázou PostgreSQL.

Poznámky:
- inštalácia BackLogs pluginu pre RedMine
- riešenie problémov s PostgreSQL databázou
- začiatok kopírovania existujúceho riešenia minuločného tímu

Zhodnotenie úloh:
- Úloha 2.2
  Nakonfigurovať systém Redmine
  Zodpovednosť: Bc. Pavol Škvarenina
  Úloha splnená

- Úloha 2.3
  Nakonfigurovať systém Redmine
  Zodpovednosť: Bc. Pavol Škvarenina
  Úloha splnená

- Úloha 3.1
  Zverejniť zápisnice zo stretnutia na webe

Zodpovednosť: Bc. Radoslav Zachar
Úloha splnená

- Úloha 3.2
  Študovať framework YII v rozsahu 16 hodín
  Zodpovednosť: Bc. Jakub Calík
  Úloha sa priebežne plní

- Úloha 3.3
  Inštalácia plug-inu Backlogs pre systém Redmine
  Zodpovednosť: Bc. Viliam Kubis
  Úloha splnená

Udelené úlohy:

- Úloha 4.1
  Nakopírovanie celého systému od minuloročného tímu
  Zodpovednosť: Bc. Viliam Kubis
  Termín splnenia: 2.11.2011

- Úloha 4.2
  Naplnenie databáz potrebnými dátami
  Zodpovednosť: Bc. Viliam Kubis
  Termín splnenia: 2.11.201

Zápisnica z tímového stretnutia č. 5

Dátum:	2.11.2011	
Čas:	11:00 – 13:15	
Miesto:	Softvérové štúdio FIIT STU v BA	
Účastníci:	Vedúci pedagóg:	Ing. Miroslav Galbavý
	Členovia tímu:	Bc. Jakub Calík
		~~Bc. Ondrej Danada~~
		Bc. Matúš Hitka
		Bc. Marián Hlavenka
		Bc. Viliam Kubis
		Bc. Pavol Škvarenina
		Bc. Radoslav Zachar
Zapisnicu	Vypracoval:	Bc. Radoslav Zachar
	Overil:	Bc. Marián Hlavenka

Téma stretnutia:
Spôsob práce so systémom RedmMine a PostgreSQL databázou.

Priebeh stretnutia:
Tímové stretnutie sa nieslo v duchu školenia. Každý člen tímu mal možnosť prezrieť nahodený minuloročný systém a Redmine.

Poznámky:

- Paľo vysvetlil, ako sa pracuje so systémom Redmine
- Vilo nás oboznámil so spôsobom práce s PostgreSQL databázou
- Prezreli sme importovanú databázu
- Dohodli sme sa na nasledovnej hierarchii úloh:
  - 3 základné typy úloh: support, feature a bug
  - základné úlohy môžu pozostávať z jednej alebo viacerých podúloh (Task)
- Nadradené úlohy bude vytvárať Paľo ako manažér plánovania a bude ich prideľovať Kubovi ako manažérovi kvality, ktorý dohliadne na ich splnenie
- Podúlohy bude prioritne vytvárať Paľo, ale môže ich aj vytvoriť ľubovoľný člen tímu, ak to uzná za potrebné
- Podúlohy sa prideľujú priamo osobe, ktorá ich bude vykonávať
- Každý člen tímu upravil stav svojich podúloh v systéme Redmine do súčastnej podoby
- Ukončenie a zhodnotenie Iterácie I1
- Priebežný plán Iterácie I2

Zhodnotenie úloh:

- Úloha 3.2
  Študovať framework YII v rozsahu 16 hodín
  Zodpovednosť: Bc. Jakub Calík
  Úloha splnená

- Úloha 4.1
  Nakopírovanie celého systému od minuloročného tímu
  Zodpovednosť: Bc. Viliam Kubis
  Úloha splnená

- Úloha 4.2
  Naplnenie databáz potrebnými dátami
  Zodpovednosť: Bc. Viliam Kubis
  Úloha splnená

Udelené úlohy:

- Úloha 5.1
  Naučiť sa používať Redmine
  Zodpovednosť: Bc. Jakub Calík
  Termín splnenia: 9.11.2011

- Úloha 5.2
  Spojazdniť SVN
  Zodpovednosť: Bc. Viliam Kubis
  Termín splnenia: 9.11.2011

- Úloha 5.3
  Analýza minuloročného systému
  Zodpovednosť: Bc. Jakub Calík
  Termín splnenia: 9.11.2011

- Úloha 5.4
  Framework YII v rámci minuloročného systému
  Zodpovednosť: Bc. Jakub Calík
  Termín splnenia: 2.12.2011

Zápisnica z tímového stretnutia č. 6

Dátum:	9.11.2011	
Čas:	11:00 – 13:30	
Miesto:	Softvérové štúdio FIIT STU v BA	
Účastníci:	Vedúci pedagóg:	Ing. Miroslav Galbavý
	Členovia tímu:	Bc. Jakub Calík
		Bc. Ondrej Danada
		Bc. Matúš Hitka
		Bc. Marián Hlavenka
		Bc. Viliam Kubis
		Bc. Pavol Škvarenina
		Bc. Radoslav Zachar
Zapisnicu	Vypracoval:	Bc. Ondrej Danada
	Overil:	Bc. Matúš Hitka

Téma stretnutia:
Projektová dokumentácia

Priebeh stretnutia:
Stretnutie sa nieslo vo forme brainstormingu. Dopracovali sme časti dokumentácie ohľadom analýzy problematiky a taktiež analýzy existujúceho riešenia. Začali sme pripravovať a písať dokumentáciu k riadeniu.

Poznámky:
- Úprava zápisníc do jednotného formátu
- Dopracovanie analýzy a návrhu do finálnej podoby
- Rozpracovanie dokumentácie k riadeniu
- Dokončenie plánu Iterácie I2

Zhodnotenie úloh:

- Úloha 5.1
  Naučiť sa používať Redmine
  Zodpovednosť: Bc. Jakub Calík
  Úloha splnená

- Úloha 5.2
  Spojazdniť SVN
  Zodpovednosť: Bc. Viliam Kubis
  Úloha splnená

- Úloha 5.3
  Analýza minuloročného systému
  Zodpovednosť: Bc. Jakub Calík
  Úloha sa priebežne plní

- Úloha 5.4
- Framework YII v rámci minuloročného systému
  Zodpovednosť: Bc. Jakub Calík
  Úloha sa priebežne plní

Udelené úlohy:

- Úloha 6.1
  Možnosti grafov v systéme Redmine
  Zodpovednosť: Bc. Pavol Škvarenina
  Termín splnenia: 16.11.2011

- Úloha 6.2
  Dokončiť analýzu a návrh riešenia
  Zodpovednosť: Bc. Radoslav Zachar
  Termín splnenia: 16.11.2011

Zápisnica z tímového stretnutia č. 7

Dátum:	16.11.2011	
Čas:	11:00 – 13:30	
Miesto:	Softvérové štúdio FIIT STU v BA	
Účastníci:	Vedúci pedagóg:	Ing. Miroslav Galbavý
	Členovia tímu:	Bc. Jakub Calík
		Bc. Ondrej Danada
		Bc. Matúš Hitka
		Bc. Marián Hlavenka
		Bc. Viliam Kubis
		~~Bc. Pavol Škvarenina~~
		Bc. Radoslav Zachar
Zapisnicu	Vypracoval:	Bc. Viliam Kubis
	Overil:	Bc. Marián Hlavenka

Téma stretnutia:
Zisťovanie a postupné odľaďovanie chýb minuloročného systému + spojazdniť GUI framework-u Yii (Gii).

Priebeh stretnutia:
Opravenie chyby v *eventcontroller.php*. Úspešné spojazdnenie grafického generátora kódu Gii a nastavenie hesla. SVN checkout do webroot-u a commit zmien späť do SVN.

Poznámky:

- Eventcontroller.php obsahoval pozostatok po copy-paste operácií (zlý index v poli)
- Nastavenie Gii pre prístup zo všetkých IP adries
- Pokus zistiť priebeh *save* operácie modelu *event* v rámci framework-u Yii
- Update webroot-u najnovším checkoutom z SVN

Zhodnotenie úloh:

- Úloha 5.3
  Preštudovať minuloročný systém
  Zodpovednosť: Bc. Jakub Calík
  Úloha sa priebežne plní

- Úloha 5.4
  Framework YII v rámci minuloročného systému
  Zodpovednosť: Bc. Jakub Calík
  Úloha sa priebežne plní

- Úloha 6.1
  Možnosti grafov v systéme Redmine
  Zodpovednosť: Bc. Pavol Škvarenina
  Úloha splnená

- Úloha 6.2
  Dokončiť analýzu a návrh riešenia
  Zodpovednosť: Bc. Radoslav Zachar
  Úloha splnená

Udelené úlohy:

- Úloha 7.1
  Priebežné spisovanie zistených chýb v minuloročnom systéme
  Zodpovednosť: Bc. Jakub Calík
  Termín splnenia: 30.11.2011

- Úloha 7.2
  Rozchodenie vytvárania rozvrhových akcií
  Zodpovednosť: Bc. Viliam Kubis
  Termín splnenia: 23.11.2011

- Úloha 7.3
  Vytvorenie Cron skriptu pre automatický checkout najnovšieho kódu z SVN do webrootu každé ráno o 5:00
  Zodpovednosť: Bc. Viliam Kubis
  Termín splnenia: 23.11.2011

Zápisnica z tímového stretnutia č. 8

Dátum:	23.11.2011	
Čas:	11:00 – 13:30	
Miesto:	Softvérové štúdio FIIT STU v BA	
Účastníci:	Vedúci pedagóg:	Ing. Miroslav Galbavý
	Členovia tímu:	Bc. Jakub Calík
		Bc. Ondrej Danada
		Bc. Matúš Hitka
		Bc. Marián Hlavenka
		Bc. Viliam Kubis
		Bc. Pavol Škvarenina
		Bc. Radoslav Zachar
Zapisnicu	Vypracoval:	Bc. Jakub Calík
	Overil:	Bc. Radoslav Zachar

Téma stretnutia:
Spôsob a postup opravovania chýb v minulorocnom projekte v Yii frameworku.

Priebeh stretnutia:
Na stretnutí sa zúčastnil aj člen minuloročného tímu, ktorý pracoval prioritne s Yii frameworkom. Za jeho pomoci sme konzultovali objavené chyby a riešili aplikačnú logiku webovej aplikácie. Keďže celý systém pozostáva hlavne z modulov a controllerov automaticky vygenerovaných Gii generátorom v rámci Yii frameworku, konzultovali sme aj štruktúru celého systému a jednotlivé jeho zložky. Zhodli sme sa na tom, že systém by bolo treba previesť do novej verzie Yii frameworku. Keďže nová verzia ale nie je úplne kompatibilná s verziou, v ktorej bol systém vyvíjaný, bude potrebné nanovo vygenerovať väčšinu modulov v aplikácii a následne ich upraviť podľa potreby a požiadaviek na systém.

Zhodnotenie úloh:

- Úloha 5.3
  Preštudovať minuloročný systém
  Zodpovednosť: Bc. Jakub Calík
  Úloha úspešne splnená

- Úloha 5.4
  Framework YII v rámci minuloročného systému
  Zodpovednosť: Bc. Jakub Calík
  Úloha úspešne splnená

- Úloha 7.1
  Priebežné spisovanie zistených chýb v minuloročnom systéme
  Zodpovednosť: Bc. Jakub Calík
  Úloha sa priebežne plní

- Úloha 7.2
  Rozchodenie vytvárania rozvrhových akcií
  Zodpovednosť: Bc. Viliam Kubis
  Úloha úspešne splnená

- Úloha 7.3
  Vytvorenie Cron skriptu pre automatický checkout najnovšieho kódu z SVN do webrootu každé ráno o 5:00
  Zodpovednosť: Bc. Viliam Kubis
  Úloha sa o týždeň presunula

Udelené úlohy:

- Úloha 8.1
  Inštalácia najnovšieho Yii framewoku
  Zodpovednosť: Bc. Jakub Calík
  Termín splnenia: 30. 11. 2011

- Úloha 8.2
  Pregenerovanie modulov a vytvorenie controllerov pre novú verziu Yii frameworku
  Zodpovednosť: Bc. Jakub Calík
  Termín splnenia: 30. 11. 2011

- Úloha 8.3
  Sfunkčnenie LDAP prihlásenia do systému
  Zodpovednosť: Bc. Jakub Calík
  Termín splnenia: 30. 11. 2011

Zápisnica z tímového stretnutia č. 9

Dátum:	30.11.2011	
Čas:	11:00 – 13:30	
Miesto:	Softvérové štúdio FIIT STU v BA	
Účastníci:	Vedúci pedagóg:	Ing. Miroslav Galbavý
	Členovia tímu:	Bc. Jakub Calík
		Bc. Ondrej Danada
		Bc. Matúš Hitka
		Bc. Marián Hlavenka
		Bc. Viliam Kubis
		Bc. Pavol Škvarenina
		Bc. Radoslav Zachar
Zapisnicu	Vypracoval:	Bc. Jakub Calík
	Overil:	Bc. Radoslav Zachar

Téma stretnutia:
Ukončenie Iterácie I2, plánovanie Iterácie I3 a generovanie modulov systému v novej verzii Yii frameworku.

Priebeh stretnutia:
Na stretnutí sme rozdelili úlohy tímu pre ďalšiu iteráciu. Časť tímu bude pracovať na prezentácii vynaloženého úsilia za prvý semester tímového projektu. Druhá časť tímu sa bude zaoberať sústavným riešením chýb starého systému a generovaním novej stabilnej verzie modulov systému v Gii generátore. Všetky rozdelené úlohy sú zaznamené v systéme Redmine, spolu s pridelenou zodpovedajúcou osobou k danej úlohe.

Poznámky:
- Zhodnotenie a ukončenie iterácie I2
- Plán iterácie I3

Zhodnotenie úloh:

- Úloha 7.1
  Priebežné spisovanie zistených chýb v minuloročnom systéme
  Zodpovednosť: Bc. Jakub Calík
  Úloha úspešne splnená

- Úloha 7.3
  Vytvorenie Cron skriptu pre automatický checkout najnovšieho kódu z SVN do webrootu každé ráno o 5:00
  Zodpovednosť: Bc. Viliam Kubis
  Úloha úspešne splnená

- Úloha 8.1
  Inštalácia najnovšieho Yii framewoku
  Zodpovednosť: Bc. Jakub Calík
  Úloha úspešne splnená

- Úloha 8.2
  Pregenerovanie modulov a vytvorenie controllerov pre novú verziu Yii frameworku
  Zodpovednosť: Bc. Jakub Calík
  Úloha sa o týždeň presunula

- Úloha 8.3
  Sfunkčnenie LDAP prihlásenia do systému
  Zodpovednosť: Bc. Jakub Calík
  Úloha sa o týždeň presunula

Udelené úlohy:

- Úloha 9.1
  Vytvorenie prezentácie pre odovzdanie prototypu za prvý semester
  Zodpovednosť: Bc. Jakub Calík
  Termín splnenia: 13. 12. 2011

Zápisnica z tímového stretnutia č. 10

Dátum:	07.12.2011	
Čas:	11:00 – 13:30	
Miesto:	Softvérové štúdio FIIT STU v BA	
Účastníci:	Vedúci pedagóg:	Ing. Miroslav Galbavý
	Členovia tímu:	Bc. Jakub Calík
		Bc. Ondrej Danada
		Bc. Matúš Hitka
		Bc. Marián Hlavenka
		Bc. Viliam Kubis
		Bc. Pavol Škvarenina
		Bc. Radoslav Zachar
Zapisnicu	Vypracoval:	Bc. Matúš Hitka
	Overil:	Bc. Marián Hlavenka

Téma stretnutia:
Dokončenie migrácie systému na novú verziu frameworku Yii a práca na finálnej prezentácii.

Priebeh stretnutia:
Na stretnutí sme sa dohodli na obsahu a forme prezentácie zimnej časti tímového projektu. Dohodli sme sa taktiež na finalizácii dokumentácie k riadeniu a softvérovému dielu. Každý člen tímu dopíše do dokumentácie k riadeniu časť ohľadom manažmentu, ktorý počas semestra vykonával. Tiež sme na stretnutí hľadali najvhodnejší termín prezentácie projektu, ktorý by vyhovoval všetkým zúčastneným stranám.

Poznámky:

Zhodnotenie úloh:

- Úloha 8.2
  Pregenerovanie modulov a vytvorenie controllerov pre novú verziu Yii frameworku
  Zodpovednosť: Bc. Jakub Calík
  Úloha úspešne splnená

- Úloha 8.3
  Sfunkčnenie LDAP prihlásenia do systému
  Zodpovednosť: Bc. Jakub Calík
  Úloha úspešne splnená

- Úloha 9.1
  Vytvorenie prezentácie pre odovzdanie prototypu za prvý semester
  Zodpovednosť: Bc. Jakub Calík
  Termín splnenia: 13. 12. 2011
  Úloha sa priebežne plní

Udelené úlohy:

- Úloha 10.1
  Dokončenie dokumentácie k riadeniu
  Zodpovednosť: Bc. Jakub Calík
  Termín splnenia: 13. 12. 2011

- Úloha 10.2
  Dokončenie a konsolidácia prototypu pred odovzdaním
  Zodpovednosť: Bc. Jakub Calík
  Termín splnenia: 13. 12. 2011

- Úloha 10.3
  Dohodnutie dátumu konania prezentácie s druhým tímom
  Zodpovednosť: Bc. Marián Hlavenka
  Termín splnenia: 14. 12. 2011