Tím 6

Rozvrhový systém FIIT

Tímový projekt

Autori:	Bc. Radoslav Zachar, Bc. Viliam Kubis, Bc. Jakub Calík, Bc. Pavol Škvarenina, Bc. Marián Hlavenka, Bc. Matúš Hitka, Bc. Ondrej Danada
Kontakt:	tim6@danada.sk
Študijný odbor:	Softvérové inžinierstvo, Informačné systémy
Ročník:	1. Ing.
Akademický rok:	2011/2012
Vedúci práce:	Ing. Miroslav Galbavý

História vývoja dokumentu

Dátum	Verzia	Opis	Autor
3.11.2011	1.0	Vytvorenie dokumentu	Radoslav Zachar
4.11.2011	1.1	Vytvirenie hlavných kapitol, vloženie obsahu a zadania, spísanie úvodu	Radoslav Zachar
8.11.2011	1.2	Pridanie motivácie a opis vývojového tímu	Radoslav Zachar
9.11.2011	1.3	Doplnenie analýzy, návrhu, opis problému, instalačnej a používateľskej príručky	Radoslav Zachar
10.11.2011	1.4	Práca na analýze, návrhu a špecifikácii	Radoslav Zachar
11.11.2011	1.5	Vloženie inštalačnej a používateľskej príručky	Radoslav Zachar
25.11. 2011	1.6	Pridané kapitoly implementácia, GUI	Radoslav Zachar
9.12. 2011	1.7	Vloženie biznis procesov	Radoslav Zachar
12.12.2011	1.8	Korektúra textov	Radoslav Zachar

Obsah

Zoznam obrázkov

1. Úvod

1.1 Účel a rozsah dokumentu

Predkladaný dokument obsahuje analýzu rozvrhového systému, ktorý sme dostali ako tím na dokončenie a nasadenie. K systému sme dostali aj kompletnú dokumentáciu odovzdanú minuloročným tímom na Fakulte informatiky a informačných technológií Slovenskej technickej univerzity v Bratislave.

Tento dokument obsahuje zhrnutie všetkých riešení nášho tímu na vylepšenie rozvrhového systému. Dokument je určený hlavne na analýzu riešeného problému a návrh vývoja prevzatého systému.

Rozsah dokumentu je prispôsobený harmonogramu predmetu Tímový projekt a popisuje zatiaľ len analýzu systému a existujúcich problémov.

1.2 Motivácia

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.

1.3 Vývojový tím

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.

1.4 Použité skratky

AIS	–	Akademický Informačný Systém
LDAP	–	Lightweight Directory Access Protocol
FIIT	–	Fakulta Informatiky a Informačných Technológií
FEI	–	Fakulta Elektrotechniky a informatiky
DB	–	Databáza

2. Zadanie

Aj po nasadení systému AIS, zostáva stále otvorená problematika prípravy a vytvárania semestrálnych rozvrhov a následne aj skúškových rozvrhov.

V súčasnosti je rozpracovaný špecializovaný systém pre tvorbu rozvrhov (momentálne viazaný na prostredia PHP a PostgreSQL), ktorý už teraz umožňuje evidenciu údajov potrebných pre vytvorenie rozvrhov, poloautomatické (núdzovo aj "ručné") zapísanie/importovanie/exportovanie rozvrhu oprávneným používateľom, formuláciu požiadaviek na rozvrhy od pedagógov a čiastočne aj študentov, ako aj ich zverejňovanie pre rôzne komunity používateľov (so základnou podporou riešenia "konfliktných" požiadaviek a notifikácie pre všetky dotknuté strany), publikovanie rozvrhu pre ostatných používateľov cez internet.

Hlavné úlohy pre tento rok budú dopracovať doteraz vytváraný systém aj s prípadnými úpravami pre podmienky novej budovy (podľa vhodnosti a dostupných kapacít učební a laboratórií) a s ohľadom na externe zabezpečované prednášky a cvičenia (personálne a priestorovo / najmä väzby s FEI), podsystémy prípravy podkladov, zberu požiadaviek (garantov, prednášajúcich, cvičiacich, študentov), vytvárania a publikovania rozvrhov a nasadiť vytvorený systém do skúšobnej prevádzky a použiť ho pri vytvorení rozvrhu FIIT pre akad. rok 2012/2013.

3. Opis riešeného problému

3.1 Rozvhový systém

Tvorba rozvrhu nie je jednoduchá činnosť a väčšinou je sprevádzaná aj obrovským množstvom vzniknutých kolízií. Manuálne riešenie týchto kolízii je v dnešnej dobe veľmi zložité. Z tohto dôvodu vzniká požiadavka na tvorbu automatizovaného elektronického systému pre všetkých účastníkov tvorby rozvrhov, ktorý by bol schopný výrazne uľahčiť zdĺhavý proces tvorby rozvrhu a dokázal by vyriešiť kolízie bez akéjkoľvek pomoci.

3.2 Učiteľská časť

Učiteľov je nutné rozdeliť do viacerých kategórií podľa ich akademickej hodnosti a právomoci pre jednotlivé predmety:

- garant študijného programu,
- garant predmetu,
- prednášajúci,
- vedúci cvičení,
- cvičiaci,
- študenti doktorandského štúdia, ktorí vyučujú predmet ako cvičiaci.

Garant študijného programu

Garantom študijného programu je vždy vysokoškolský profesor alebo docent. Jeho hlavnou úlohou je zabezpečenie kvality daného študijného programu. Dohliada na dodržiavanie ustanovení pre daný študijný program a jednotlivé predmety, ktoré patria danému programu. Takisto má aj rozhodovacie práva pri schvaľovaní nových predmetov.

Garant predmetu

V rámci predmetu predstavuje najväčšiu autoritu garant predmetu.
Jeho úlohou je:

- určiť maximálne počty študentov pre prednášky a cvičenia
- navrhnúť vhodné druhy miestností
- požadovať potrebné vybavenie miestností
- rozložiť prednášky a cvičenia počas týždňa

Tieto požiadavky sú následne sprístupnené prednášajúcemu predmetu.

Prednášajúci

Prednášajúci je povinný sa riadiť pokynmi garanta predmetu. Medzi jeho základné požiadavky patrí voľba miestnosti a času prednášky. Jeho ďalšie povinnosti sú:

- rozdelenie prednášky na niekoľko termínov počas týždňa
- rozloženie prednášok počas semestra

- pridelenie miestnosti na prednášku s určitým vybavením, ktoré by malo byť k dispozícii počas každej alebo vopred určenej prednášky

Na prednášané učivo by sa mali adekvátne viazať aj cvičenia predmetu.

Cvičiaci

Cvičenia sú vedené zodpovednou osobou určenou garantom predmetu. Môže to byť učiteľ, prednášajúci, odborný asistent pridelený k danému predmetu alebo aj študent doktorandského štúdia.

Cvičiaci sa vyjadrujú k rozvrhu ako poslední. Cvičiaci má možnosť ovplyvniť:

- počet cvičení a ich rozsah
- počet študentov, ktorí sa môžu zúčastniť naraz jedného cvičenia
- možnosť viesť naraz viaceré cvičenia
- vybavenie učebne

Ako bolo spomenuté vyššie, špeciálnym prípadom cvičiacich sú doktorandi, pretože voľba rozvrhu sa v ich prípade riadi aj učiteľským princípom, aj študentským. V rozvrhu doktorandov sa vyskytujú predmety, na ktorých zastávajú rolu študenta, ale vyskytujú sa aj predmety, ktoré oni vyučujú – cvičia.

Úväzky

Dôležitým faktorom vplývajúcim na počet cvičení a prednášok sú zamestnanecké úväzky. Fakulta prerozdeľuje vlastné financie medzi predmety a samozrejme aj medzi prislúchajúcich pedagógov.

Úväzky majú teda nepriamy vplyv na počet otvorených predmetov, počty prednášok a cvičení.

3.3 Konflikty

Po zapracovaní pripomienok, ktoré majú pedagógovia k rozvrhom sa vytvára predbežná verzia rozvrhu. Táto je následne sprístupnená študentom. Je dôležité ošetriť všetky konflikty, ktoré vznikli pri jej vytváraní.

Druhy konfliktov, ktoré môžu nastať pri vytváraní rozvrhov :

- dve rozvrhové akcie sa konajú v rovnakých miestnostiach a čase,
- jednému vyučujúcemu je priradená výučba viacerých rozvrhových akcií v rovnakom čase,
- dvom vyučujúcim je priradená výučba tej istej rozvrhovej akcie,
- dve rozvrhové akcie rôznych predmetov sa konajú v rovnakom čase, takže študent, ktorý ma zapísané oba predmety, sa nemôže jednej z nich zúčastniť.

Konflikty môžeme podľa rôznych príčin vzniku riešiť nasledovne:

- posunom času rozvrhovej akcie na iný čas v rámci toho istého dňa,
- posunom termínu rozvrhovej akcie na iný deň,
- nájdením novej miestnosti, ktorá by vyhovovala požiadavkám,
- výmenou cvičiacich,
- zrušením termínu cvičenia a rozšírením počtu študentov na iných cvičeniach, ak to kapacita cvičenia dovoľuje,
- pridaním nového termínu cvičenia,
- posunom iného cvičenia.

Konkrétny výber riešenia konfliktu sa zvolí na základe preferencií tvorcu rozvrhov a možností fakulty. Za problematický spôsob riešenia považujeme poslednú možnosť, pretože posun jedného cvičenia môže ovplyvniť rozvrh iného predmetu.

3.4 Študentská časť

Cieľom študenta je mať rozvrh, ktorý spĺňa čo najviac jeho osobných kritérií. Tieto sa môžu od študenta k študentovi výrazne líšiť, napríklad podľa toho, či je študent ubytovaný na internáte, alebo dochádza, prípadne či popri škole pracuje. Jednou takou požiadavkou je napríklad rozvrh „bez dier". Málokomu vyhovuje mať medzi jednotlivými prednáškami a cvičeniami 1-2 hodinové pauzy, v ktorých sa len ťažko vykonáva zmysluplná aktivita. Medzi ďalšie kritériá môžu patriť:

- rozvrh mať rozdelený na čo najmenší počet dní
- v piatok voľno
- nezačínať skoro ráno, nekončiť neskoro večer. Alebo naopak, mať rozvrh rozdelený len do ranných a večerných hodín, aby mal cez deň čas na prácu
- každý deň mať predmety v súvislých blokoch. To znamená, že študent nemusí chodiť do školy napríklad do obedu a potom zase až večer

Spraviť rozvrh, s ktorým by bol spokojný každý, je zrejme nemožné. Preto je potrebné pri tvorbe prototypu rozvrhu v učiteľskej časti identifikovať také kritériá, ktoré by vyhovovali väčšine študentov.

Registrácia a zápis predmetov

Tvorbu svojho rozvrhu študenti začínajú registráciou predmetov, väčšinou na konci akademického roku. Ak sa študenti držia odporúčaného študijného plánu, tak v prvých ročníkoch bakalárskeho štúdia majú rozvrhy skoro identické. V bežnom akademickom roku pripadá na jeden semester približne 5-7 kreditovaných predmetov. Štandardný rozsah jedného predmetu je 3 hodinová prednáška a 2 hodinové cvičenie. Pri voliteľných predmetoch sa môže stať, že sa prekročí kapacita a nie je možné zobrať všetkých registrovaných študentov. Vtedy sa predmet pridelí študentom s vyšším VŠP. Môže sa stať, že sa nenaplní minimálna kapacita, a predmet sa neotvorí. Kvôli takýmto prípadom sa pri registrácii pri každom voliteľnom predmete uvádza aj jeden alternatívny voliteľný predmet z rovnakej skupiny.
Na začiatku akademického roka študenti dostanú zoznam predmetov, ktoré im boli na základe registrácie pridelené. Svoju účasť na predmetoch potvrdia zápisom do daného akademického roku. Na základe tohto zápisu je následne možné určiť presný počet študentov na jednotlivé predmety.

Výber termínov cvičení

Krátko po zápise do štúdia sa v systéme AIS všetkým študentom zverejní ich potenciálny rozvrh. Prednášky majú v rozvrhu pevne stanovené termíny. Na cvičenia k danému predmetu môže pripadať aj viac termínov, z ktorých jeden si študent musí vybrať. Vyberanie termínov na cvičenia začína pár dní po zverejnení prototypu rozvrhu. Študenti sú podľa VŠP rozdelení do skupín. Skupina s najlepším VŠP si môže vyberať termíny cvičení ako prvá. Najlepší študenti tak majú možnosť vybrať si presne tie termíny, ktoré im najviac vyhovujú. Po pár hodinách si môže začať vyberať ďalšia skupina, potom ďalšia, atď. Študentom s najnižším VŠP sa môže stať, že v čase, keď sa im sprístupní výber cvičení, už budú ich vytipované termíny obsadené, alebo tam bude voľných len pár miest. Pri študentoch, ktorí niektoré predmety opakujú, môže dokonca nastať situácia, že im zostanú len termíny, ktoré sú v konflikte napríklad s prednáškami. Platí pravidlo, kto skôr príde, ten skôr berie.

Nedostatky súčasného riešenia výberu termínov cvičení

Výber cvičení väčšinou vyzerá tak, že študent neuroticky obnovuje stránku AIS pokiaľ sa mu neotvorí možnosť vyberať si, a potom čo najrýchlejšie kliká na svoje termíny. Študentom, ktorí sa z určitých dôvodov oneskoria pri výbere cvičení, potom ostávajú väčšinou len tie najhoršie termíny, napr. ráno o 7:00, alebo v piatok poobede. Študenti sa jednoducho musia v presne stanovenom čase prihlásiť do systému AIS a čo najrýchlejšie si vyklikať cvičenia. Ak má niekto v danom čase neodkladné povinnosti, alebo sa vyskytnú nečakané udalosti, prakticky stráca možnosť vybrať si termíny, ktoré mu vyhovujú. Zle rozdelený rozvrh môže negatívne zasiahnuť do celkového priebehu semestra a prospechu študenta. Toto je jeden z hlavných

aspektov, ktoré by chcel náš systém vylepšiť. Študent nebude pri výbere cvičení viazaný na presný termín a čas a zároveň nebude odkázaný na rýchlosť svojho klikania, internetového pripojenia, počítača a pod.

Ako už bolo spomenuté, súčasný systém nerieši prípad, keď kvôli opakovaným predmetom vznikne študentovi konflikt a neostane mu žiadny vhodný termín. Možným riešením by bolo prioritné pridelenie kritických termínov danému študentovi tak, aby sa konfliktu zabránilo ešte pred začatím výberu cvičení. Toto by mohlo byť chápané ako uprednostňovanie, čo nie je fér voči ostatným študentom. Problém je v tom, že študent by dostal dané cvičenie prakticky „zadarmo“, bez príslušného postihu za to, že opakuje a musia sa kvôli nemu robiť výnimky. Nový systém bude založený na bodovom systéme, postih bude možné vykonať odčítaním definovaného počtu bodov.

3.5 Návrh študentskej časti minuloročným tímom

Bodový systém

V súčasnom systéme si môžu študenti s lepším VŠP vyberať termíny cvičení skôr a majú tak väčšiu šancu zostaviť si rozvrh podľa ich predstáv. Stále však ostáva problém, že ak si z rôznych príčin študent v stanovenom čase cvičenia vyberať nemôže, tak mu ani lepší VŠP nepomôže.

Bodový systém je v tomto ohľade flexibilnejší. Cieľom nového systému je poskytnúť študentom rovnakú šancu na výber cvičenia počas celého trvania procesu výberu cvičení. Bodový systém bude tiež založený na VŠP. Matematický model bodového systému je bližšie špecifikovaný v ďalšej časti dokumentu. Študent môže svoje body rozdeľovať na jednotlivé termíny cvičení. Podľa toho, akú prioritu má pre študenta daný termín, toľko bodov doňho investuje. Kto dá viac bodov, ten daný termín získa. Študent bude vidieť, koľko bodov investovali na daný termín ostatní študenti a podľa toho sa môže rozhodnúť, koľko bodov bude potrebovať na jeho získanie. Niekedy môže študentovi záležať len na jednom termíne a je ochotný doňho investovať všetky svoje body.

Rozdelenie bodov môže študent meniť počas celého trvania prihlasovania na cvičenia. Tým sa odbúra nutnosť riadiť sa presnými časovými termínmi. Študent má na tvorbu svojho rozvrhu celé vymedzené časové obdobie. Je dokonca výhodné najprv počkať, ako sa vyvinie situácia a potom rozdeľovať body na základe aktuálneho stavu a priorít. Na rozdiel od súčasného stavu, kedy sa všetci sústredia na hodinu začiatku prihlasovania na cvičenia, sa ťažisko presunie skôr na hodinu konca tohto procesu. Ak však bude študentovi na niektorých termínoch veľmi záležať, investuje do nich svoje body tak, aby sa nemusel báť, že ho niekto prebije. Je len na študentovi, či zvolí vyčkávaciu taktiku a bude špekulovať až do konca, alebo sa rozhodne ísť na istotu. Záleží to však aj od počiatočného množstva získaných bodov a počtu prioritných termínov, o ktoré sa študent rozhodne bojovať.

Pri prioritnom pridelení cvičenia študentovi tak, aby nevznikol konflikt, sa bude postihovať strhnutím adekvátneho počtu bodov. Študent tak síce dostane určitú výhodu oproti ostatným, lebo nemusí o dané cvičenie bojovať, ale zároveň je potrestaný a ostane mu menej bodov na investovanie. Výška strhnutých bodov sa bude líšiť v závislosti od „exkluzivity“ termínu cvičenia, ktorý bol študentovi prioritne pridelený.

Spolupráca študentov pri výbere termínov cvičení

V súčasnom systéme chýba podpora akejkoľvek spolupráce študentov pri výbere cvičení. Kamaráti, ak chcú mať podobné rozvrhy, sa musia dohodnúť napríklad mailom alebo ústne, na ktorých cvičeniach chcú byť spolu. Aj keď si však vyberú spoločný termín, nemajú záruku, že v rámci cvičenia budú aj v spoločnej skupine. Na jeden termín cvičenia často pripadá viac cvičiacich a každý vedie svoju skupinu. Navrhovaný systém bude orientovaný sociálne. Študenti budú môcť pri výbere svojich cvičení spolupracovať. Podobne ako na sieti Facebook, si študenti budú vyberať, s ktorými kamarátmi chcú zdieľať informácie o svojich aktivitách v systéme. Zoznam ľudí, ktorí budú prihlásení na určitý termín, bude anonymný. Ak sa však kamaráti dohodnú na zdieľaní informácií, budú sa v tomto zozname vidieť podľa mena. Študent si bude môcť napríklad aj zobraziť všetky cvičenia, na ktoré sú jeho kamaráti prihlásení.

Zaujímavým aspektom by mohla byť výmena bodov medzi študentmi. Ak niekomu súrne chýba pár bodov, môže poprosiť svojich kamarátov, či nemajú body na požičanie. Študent s veľa kamarátmi tak bude mať väčšiu šancu na získanie všetkých svojich prioritných termínov.

Okrem vytvárania skupín sa bude v systéme nachádzať aj všeobecné fórum, kde budú môcť študenti komunikovať navzájom medzi sebou. Fórum bude slúžiť napríklad na riešenie konfliktov. Ak bude niekto súrne potrebovať určitý termín, ale nebudú mu vychádzať body, môže sa snažiť na fóre presvedčiť niektorého zo študentov, aby mu daný termín prenechal, napríklad aj nejakou odmenou. Fórum bude slúžiť aj na navrhovanie nových termínov cvičení. Ak niekto navrhne nový termín a nájde dostatočný počet záujemcov, systém umožní odoslať tento termín na posúdenie kompetentným osobám. V systéme bude musieť byť jasne viditeľné, ktoré termíny, okrem tých oficiálnych, ešte prichádzajú do úvahy, aby študenti nenavrhovali nereálne termíny.

Burza cvičení

Po skončení procesu prihlasovania na cvičenia pomocou nového bodového systému sa stále môže stať, že niekto nedostal termín, ktorý chcel. Preto bude študentom umožnené po skončení prihlasovania na cvičenia využiť burzu cvičení. Na tejto burze bude môcť študent jednoducho ponúknuť svoj termín na výmenu a špecifikovať, ktorý termín, alebo termíny, by chcel. V súčasnosti sa takéto výmeny riešia mimo systému a keď sa študenti už aj dohodnú medzi sebou, musia sa ešte dohodnúť aj s príslušnými cvičiacimi. Burza cvičení by pre daný termín mohla teoreticky fungovať až do začiatku prvého cvičenia na tomto termíne. Cvičiaci by tak dostal finálny zoznam až tesne pred začatím cvičenia, napríklad v predchádzajúci deň do 22:00. Toto by však niektorým cvičiacim nemuselo vyhovovať a detaily systému burzy cvičení bude potrebné ešte prediskutovať s kompetentnými osobami.

Zhrnutie funkcionality študentskej časti systému

Cieľom nového systému je flexibilnejšie prihlasovanie na cvičenia pre študentov. Hlavný nedostatok aktuálneho riešenia, nutnosť pracovať so systémom v presne danom dni, hodine, minúte, by mal vyriešiť bodový systém a súťaž o termíny formou dražby. Bodový systém zároveň umožňuje prioritné prideľovanie termínov, aby sa predišlo konfliktom, aj s príslušným postihom vo forme strhnutých bodov.

Okrem vylepšenia súčasného riešenia ponúka nový systém aj významné novinky:

- možnosť vidieť, na aké cvičenia sa prihlásili kamaráti
- požičiavať si body
- ak sa nazbiera dostatok ľudí, odoslať žiadosť na otvorenie nového termínu cvičenia
- fórum
- po skončení prihlasovania na cvičenia využiť burzu cvičení

4. Analýza

4.1 Analýza existujúceho riešenia

Systém tvorby rozvrhov je vyvíjaný v rámci tímového projektu už niekoľko rokov.

Každý rok je nutné vykonať vykonať analýzu predchádzajúceho riešenia. Minulý rok sa tomuto projektu venoval tím *Schedule of Pain*. Tento tím sa postaral o prechod dovtedy vytváraného systému na inovatívny a ľahko použiteľný framework Yii.

Všetky dôležité formálne aspekty systému a aj požiadavky na systém z hľadiska každého typu používateľa boli spracované tímami pred nami.

Náš tím sa musí zamerať na overenie správnosti požiadaviek, funkčnosti systému a navrhnutého riešenia. Riešenie síce je navrhnuté, ale skoro nenaimplementované. Minuloročný tím odviedol obrovský kus práce a prechodom na Yii Framework spravil krok dopredu. Návrh ich riešenia je potrebné implementovať.

4.2 Použité technológie

Náš tím bude pri vývoji tohtoročného priradeného tímového projektu (téma Rozvrhový systém novej FIIT) používať najmä nasledovné technológie:

- skriptovací jazyk PHP, v ktorom bude napísaná celá používateľská časť systému
- framework YII, taktiež napísaný v jazyku PHP, extrémne vhodný na tvorbu Web 2.0 aplikácií
- ako webový server poslúži Apache, ako operačný systém *nix - like (konkrétne Fedora 15)
- ako databázový server poslúži PostgreSQL, ktorý beží na lokálnom stroji (Fedora 15)
- nástroj na manažovanie úloh v tíme RedMine
- C# (jadro systému)
- C++ (časovo náročné algoritmy)
- XHTML, CSS3, JavaScript (najmä knižnica jQuery), ktoré budú tvoriť samotnú web stránku a spojazdňovať skriptovanie na strane používateľa
- Eclipse a NetBeans ako vývojové prostredia

Projekt bol minuloročným tímom vyvíjaný vo vaccine daných technológií, ktoré sú podľa nášho uváženia skvelou voľbou programovacích prostriedkov a pre tvorbu web aplikácií.

5. Špecifikácia

5.1 Zber požiadaviek od učiteľov

Učiteľ vstupuje do tvorby rozvrhov ako prvý, od jeho požiadaviek sa odvíja forma proto–rozvrhu, ktorý je následne prezentovaný študentom. Prostredie by malo byť intuitívne aj pre neinformatika. Zber požiadaviek by mal byť jednorazová záležitosť, to znamená, že používateľ by mal do systému jednorazovo zadať požiadavky a viac ich nemodifikovať, pokiaľ si to situácia implicitne nevyžiada. Ďalej by bolo vhodné, keby riaditeľ ústavu mal prístup k týmto požiadavkám, ktoré môže modifikovať, pokiaľ nevyhovujú jeho predstave.

Učiteľ je ľubovoľný prednášajúci alebo garant predmetu. Pod pojmom učiteľ nebudeme chápať cvičiaceho, ktorý do požiadaviek nezasahuje. Požiadavky vypĺňa ten, ktorý rozhoduje o obsahu predmetu.

5.2 Implicitne získané požiadavky

Musíme si uvedomiť, že niektoré údaje, ktoré súvisia so zberom požiadaviek, vieme implicitne získať, bez potreby interakcie zadávateľa.
Medzi tieto požiadavky patria:

- počet študentov
- veľkosť miestnosti
- vybavenie miestnosti
- počet cvičení
- typ predmetu
- časové obmedzenia učiteľa

Počet študentov

Počet študentov je získaný na základe registrácie predmetov, keďže registrácia predmetov nie je záväzná so zápisom predmetov, pretože medzi zápisom a registráciou môžu nastať malé zmeny medzi počtami študentov na jednotlivých predmetoch. Toto by sme mali zohľadniť aj pri počte cvičení a systém by mal umožňovať pridať aj nejaké cvičenia navyše. Existujú však aj predmety s pevným horným limitom a tieto majú počet cvičení pevne ohraničený. Aj v takýchto predmetoch však môžu vznikať kolízie.

Veľkosť miestnosti

Je daná počtom zapísaných študentov. Miestnosť rezervovaná na účely prednášky môže byť aj o kategóriu väčšia, ako je minimálna veľkosť miestnosti pre daný počet študentov. Na jednotlivých predmetoch môže vzniknúť požiadavka, aby aj cvičiaci boli účastní na prednáškach. Príkladom takejto prednášky je predmet tímový projekt.

Vybavenie miestnosti

Vybavenie miestnosti je implicitne daný povahou predmetu. Bolo by však vhodné, keby učiteľ mohol zadávať dodatočné požiadavky na miestnosť.

Ako dodatočné požiadavky sme identifikovali:

- tablet
- projektor – klasický
- mikrofón

Počet cvičení

Počet cvičení závisí od počtu zapísaných študentov na daný predmet a od maximálneho počtu študentov na cvičení. Ďalej by sme mali brať do úvahy aj typ predmetu, od ktorého závisí aj štandardný počet študentov na jedno cvičenie.

Predmety môžeme rozdeliť do 3 kategórií:

- Spoločensko–sociálne
- Odborné
- Matematické

Vo všeobecnosti cvičenia sú masové, to znamená pre celý ročník alebo skupinu. Cvičenia v niektorých predmetoch sú spojené s prednáškou, preto je potrebné takéto cvičenia pokladať za prednášku a započítať ich do rozsahu prednášok.

Cvičenia môžeme rozdeliť do dvoch kategórií:

- seminárne
- laboratórne/výpočtové

Tab. 3. udáva štandardný počet študentov na cvičení:

Tab. 3. Štandardné počty študentov na cvičeniach.

Typ predmetu/ typ cvičenia	**Seminárne**	**Laboratórne/výpočtové**
Spoločensko–sociálne	∀	60
Odborné	∀	20
Matematické	∀	20

Samozrejme prednášajúci môže počtom študentov na jednotlivé cvičenia hýbať oboma smermi. Vznikla aj požiadavka, aby učiteľ mohol povoliť pretečenie horného limitu študentov na cvičenie v prípade rozvrhovej kolízie.

Časové obmedzenia učiteľa

Časové obmedzenia učiteľa sú rozhodujúce pri vytváraní rozvrhu. Učiteľ má tendenciu zneužívať akúkoľvek voľnosť, ktorá mu je poskytnutá. Bolo by však vhodné, aby učiteľ mohol vyjadriť svoje časové preferencie. Tieto preferencie potom budú brané do úvahy pri tvorbe rozvrhu.

Pri tvorbe časových obmedzení treba brať do úvahy aj faktor mimo pedagogických aktivít učiteľa, preto je nutné tieto obmedzenia zakomponovať do celkového rozvrhu.

Zdrojom týchto informácií sú:

- ústavy (schôdze ústavov, porady, ...)
- dekanát (schôdze fakulty, kolégium dekana, ...)
- pracovný čas externistov
- úväzky na výskumných úlohách a projektoch

Tieto obmedzenia sú pri tvorbe rozvrhov záväzné, ale máme aj nezáväzné obmedzenia, ktorých zdrojom sú samotní učitelia. Bolo by vhodné doplniť tieto obmedzenia o ich preferované a nepreferované časy.

5.3 Explicitne získané požiadavky

Niektoré požiadavky sme nútení vyžiadať od učiteľa explicitne. Ako sme už naznačili, chceme získať časové obmedzenia učiteľa a aj jeho požiadavky na jednotlivé predmety. Konkrétne rozloženie prednášok a cvičení v jednotlivých predmetoch, aj ako rozloženie predmetov samotných.

Explicitne získané požiadavky sú:

- časové preferencie učiteľa
- rozloženie a časová dotácia prednášok počas semestra
- rozloženie a časová dotácia cvičení počas semestra
- preferované časy prednášok a cvičení

Učiteľ by mal mať možnosť skopírovať si požiadavky na predmet aj časové preferencie z minuloročných, čo veľmi zrýchli zadávanie požiadaviek do celého systému.

Časové obmedzenia učiteľa

Časové požiadavky učiteľa môžeme rozdeliť do štyroch kategórií

1. Obsadený čas
2. Nevyhovujúci čas
3. Vyhovujúci čas
4. Voľný čas

Prvý typ požiadavky je záväzný, zatiaľ čo ostatné tri majú odporúčací charakter. Kvantita nevyhovujúceho času by mala byť obmedzená, aby sa predišlo zneužívaniu danej možnosti. Vhodné by bolo pridať aj komentár k celkovým časovým preferenciám. Daný komentár by mal byť viazaný hlavne na nevyhovujúci čas.

Rozloženie prednášok

Rozloženie prednášok je v kompetencii učiteľa. Nie každý učiteľ používa štandardné rozloženie prednášok.

Prednáška je charakterizovaná:

- časovou dotáciou
- týždňom, kedy sa uskutoční

Učiteľ by mal mať možnosť zadať svoje požiadavky veľmi rýchlo, preto je vhodné zaviesť predvolené rozloženia na týždne. Napríklad 2 + 2, ak má prednáška časovú dotáciu 4 hodiny a podobne. Učiteľovi by mal byť poskytnutý týždenný náhľad aj semestrálny náhľad.

Učiteľ by mal dostať absolútnu voľnosť v navrhnutom rozložení, mala by však byť kontrolovaná časová dotácia prednášok.

Rozloženie cvičení

Rozloženie cvičení je o to špecifickejšie, že cvičení môže byť viac, alebo sú rozdelené do časových skupín na cvičenia v párny/nepárny týždeň.

Cvičenie je charakterizované:

- časovou dotáciou
- týždňom, kedy sa uskutoční
- typom
- počtom skupín na cvičeniach

Ak sa jedná o cvičenie, ktoré má viacero skupín, učiteľ by mal mať možnosť zadať týždne, kedy sa cvičenie uskutoční a časovú dotáciu jednotlivých cvičení, obdobne ako pri rozložení prednášok v semestrálnom pohľade na rozvrh. Učiteľ by mal mať možnosť doplniť časovú nadväznosť cvičení na

prednášku. Pri cvičeniach, ktoré sú rozdelené na párny a nepárny týždeň, by mala byť doplnená aj informácia o skupinách.

Preferované časy prednášok a cvičení

Tieto informácie sú spoločné pre všetky prednášky a cvičenia. Učiteľ si pri pohľade na týždenný rozvrh zvolí miesta:

- ideálneho času prednášok
- ideálneho času cvičenia (ak je len 1. termín cvičenia)

Bolo by vhodné oznámiť učiteľovi, že toto nie je jeho finálny rozvrh a že finálny rozvrh sa môže líšiť od jeho predstáv. Učiteľ by mal mať možnosť skopírovať si svoj minuloročný rozvrh, ak existuje.

Učiteľ by mohol zadať niektoré doplňujúce požiadavky na svoj rozvrh, ako napríklad:

- maximálny počet prednášok na deň
- požiadavka na pauzu medzi jednotlivými prednáškami
- aby boli 2 prednášky v rôzne dni (napr. PSI)

5.4 Zberu požiadaviek od cvičiacich

Cvičiaci môže byť:

- garant predmetu
- prednášajúci
- vedúci cvičení
- obyčajný cvičiaci

Cvičiaci vstupuje do tvorby rozvrhov až vtedy, keď sú známe termíny cvičení, vyberá si preferované termíny cvičení. Vznikli požiadavky, aby termíny cvičení prideľoval v nasledovnom poradí priorít:

1. garant predmetu
2. prednášajúci
3. vedúci cvičení
4. riaditeľ ústavu
5. tvorca rozvrhov

Časové preferencie cvičiaceho

Časové preferencie cvičiaceho, ak je garant alebo prednášajúci, preberajú sa jeho preferencie, ktoré vyplnil pri zadávaní požiadaviek na predmet. Tieto preferencie sú doplnené o časy prednášok, ktoré sú označené ako obsadený čas.

Ak sa jedná o cvičiaceho, ktorý cvičí, tak vyplní časové preferencii obdobne ako prednášajúci. Počet nevyhovujúcich hodín by mal byť obmedzený obdobne ako pri cvičiacom.

Prideľovanie termínov cvičení

Právomoc prideliť cvičenia postupne putuje od vyššie postavených rolí k nižšie postaveným rolám, pričom riaditeľ ústavu má právo kedykoľvek vstúpiť do prideľovania cvičení. Tvorca rozvrhov má právo meniť pridelenie jednotlivých cvičení v prípade vzniknutých kolízií alebo nevyplnenia cvičiacich žiadnym z oprávnených používateľov.

Putovanie právomoci na prideľovanie cvičiacich by malo byť implicitne zabezpečené. Pokiaľ oprávnený používateľ do určitého termínu nevyplní požadované údaje, tak právomoc na vyplnenie týchto údajov prechádza na nižšie postaveného používateľa.

Pridelenie cvičení prebieha tak, že oprávnený používateľ, na základe vlastného usúdenia a odporúčania systému, pridelí jednotlivých cvičiacich na termíny cvičení. Toto odporúčanie je vypracované na základe výberu preferovaných termínov cvičiacimi.
Systém by mal kontrolovať obmedzenia na počet cvičiacich. Systém by mal kontrolovať časové obmedzenia jednotlivých cvičiacich.

Časové obmedzenia môžeme rozdeliť na:

- implicitné – vyplývajúce z časových obmedzení cvičiaceho
- explicitné – vyplývajúce z iných cvičení na iných predmetoch

Kontrola úväzkov je zabezpečená explicitne prostredníctvom riaditeľa ústavu, ktorý schvaľuje finálne rozmiestnenie cvičiacich na cvičenia.

Výber preferovaných termínov cvičení

Preferované termíny cvičení môže vyberať len ten používateľ, ktorý participuje na danom predmete ako cvičiaci. Cvičiaci si volí preferované termíny cvičení v týždennom pohľade na rozvrh nasledovne:

- vyhovujúce termíny cvičení
  - 1. beh
  - 2. beh
- nevyhovujúce termíny cvičení

Cvičiaci by mal mať možnosť zadať, či si želá mať cvičenia:

- v jeden deň
- v rade za sebou

Preferované termíny cvičení majú len informatívny charakter a slúžia ako podpora pri prideľovaní termínov cvičení oprávneným používateľom.

Kolaboratívne prostredie pre učiteľov a cvičiacich

Dané prostredie je určené pre cvičiacich a učiteľov, aby si mohli dohadovať jednotlivé úlohy a pridelenia cvičení. Prostredie by malo byť intuitívne a jednoduché. Okrem samotnej komunikácie by malo poskytovať aj notifikáciu a správu členov.

Daný by bol systém skupín, pričom jeden pedagóg (učiteľ alebo cvičiaci) môže byť členom viacerých skupín. Mená skupín budú mať, pre lepšiu orientáciu pedagóga, mená predmetov alebo ich skratiek.

Skupina je charakterizovaná:

- menom skupiny
- moderátorom skupiny
- členmi skupiny
- obsahom správ v skupine

Moderátor skupiny je človek, ktorý je zodpovedný za vedenie skupiny a mal by mať možnosť :

- spravovať príspevky v rámci skupiny
- spravovať členov skupiny

Moderátorom skupiny je buď vedúci cvičení alebo prednášajúci, alebo obaja. Členovia skupiny by mali mať možnosť byť upozornení na nové príspevky alebo odpovede na svoje príspevky.

6. Návrh

6.1 Prehľadávanie stavového priestoru

Prehľadávanie stavového priestoru je najdôležitejšia a najnáročnejšia činnosť v rámci celého systému. Priestor stavov je možné pred začiatkom zúžiť, čím zjednodušíme a urýchlime tento proces. Neskôr budú opísané viaceré postupy na predspracovanie potrebných požiadaviek.

Pre samotné prehľadávanie stavového priestoru a výber riešení bez kolízie s potrebou maximálnej spokojnosti používateľov je navrhnutá funkcia, ktorá nám dovoľuje zaviesť ohodnotenie výhodnosti jednotlivých riešení.

6.2 Inicializácia prehľadávania

Prehľadávanie inicializujeme zberom dát. Zber dát je viazaný na implementačné prostredie preto nebude možné ho opísať. Inicializácia prebehne nasledovne:

1. Načítanie vonkajších údajových štruktúr
2. Naplnenie vnútorných údajových štruktúr
3. Pridelenie termínov prednášok
4. Pridelenie termínov cvičení/seminárov (1. termín)
5. Vyhľadanie kolíznych kandidátov
6. Pridelenie kolíznych kandidátov

7. Grafické používateľské rozhranie

Systém má nasledovné grafické rozhranie:

Figure 1 Ukážka GUI

8. Implementácia

Implementácia prebehla do štandardnáho frameworku Yii. Preto konkrétne technické riešenia sa odvíjajú od metodiky implementácie formulárov pre daný framework. Vzhľadom na to, že neboli použité nadštandardné komponenty, či knižnice, neuvádzame implementáciu logiky. Statická štruktúra Yii frameworku je znázornená na Obrázku č.1.

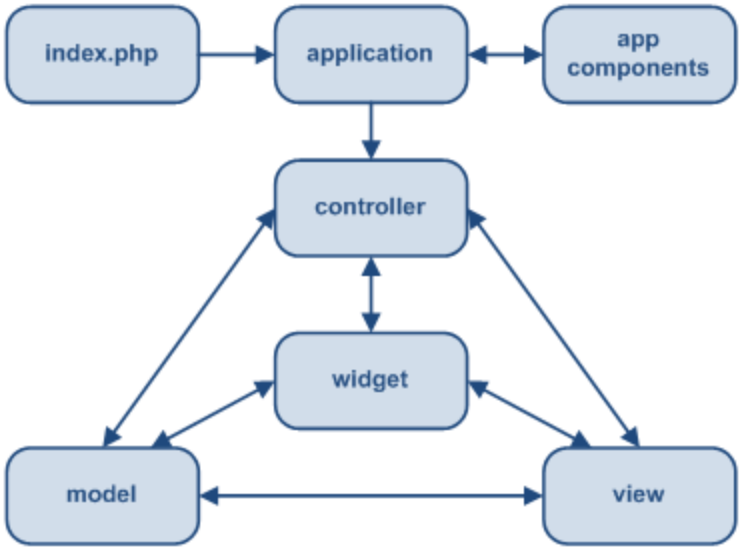

Figure 2 Štruktúra YII

Pre interaktívny formulár rozvrhu sme použili knižnicu jQuery. Zmeny v dátovom modeli sú zapracované v dátovom návrhu.

8.1 Biznis procesy

Nasledujúce biznis procesy zobrazujú súčasný stav rozvrhového systému a štruktúru nami navrhnovaného a vytváraného systému.

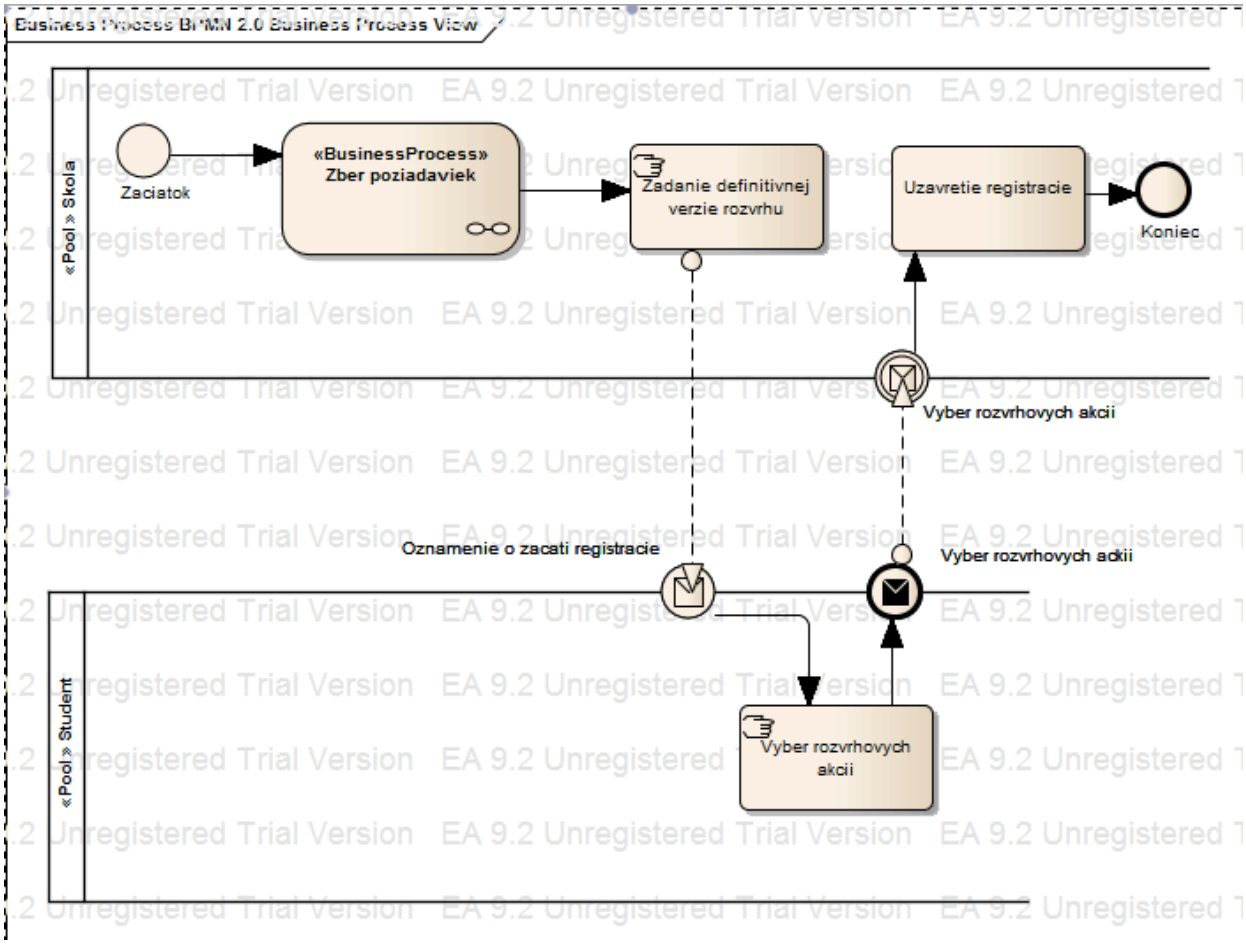

Figure 3 Súčasný stav systému

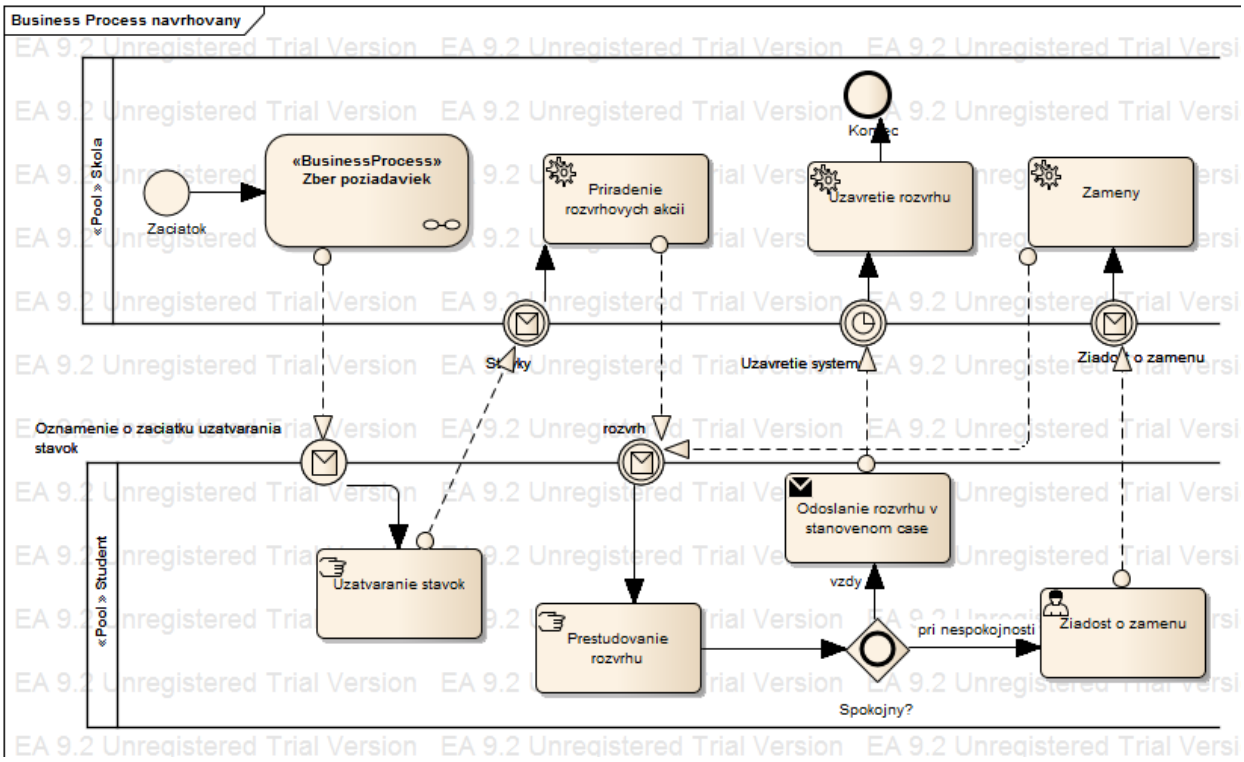

Figure 4 Navrhovaný systém

9. Inštalačná príručka

Inštalačná príručka vznikla vďaka tímu Schedule of Pain.Slúži pre inštaláciu systému na podporu tvorby rozvrhov.

Pojednáva o aplikačnom serveri Apache, databáze PGSQL ich konfigurácií pre platformy Linux/Windows 7/XP. Daná používateľská príručka bola testovaná pre Ubuntu Server 10.6 a Windows 7/XP. V prípade iných Unix-like systémov sa môže inštalácia líšiť.

9.1 Použité skratky

Ubuntu – Ubuntu Server 10.6
PGSQL -- Databáza PostgreSQL
VM -- Virtual Machine

9.2 Inštalácia a konfigurácia aplikačného servera APACHE

V prvom rade si uvedomme, že Apache nám slúži ako prezenčná a stredná vrstva systému. Zabezpečuje komunikáciu pomocou protokolu http/https s okolím. Interpretuje PHP skripty. Aplikačný server neobsluhuje JavaScript. Pretože JavaScript je záležitosť klienta.

Pre správnu funkčnosť konfigurácie odporúčam aby ste mali voľný port 80, toto je štandardný http port. Pokiaľ nemáte http port uvoľnený tak pozrite na sekciu konfigurácia portov.

Upozorňujem, že nasledovná časť príručky vyžaduje práva administrátora, preto:

- Do konzoly napíšte *su root*,
- alebo pre každý príkaz pridajte prefix sudo (SuperUser Do).

Inštalácia APACHE pre Linux

V prvom rade je potrebné aby ste si stiahli aplikačný server APACHE. Tento býva vo väčšine systémov označený ako balík apache2.

Takže máme dve možnosti

- Inštalovať *apache* a *php5* manuálne:
  - Stiahneme si z repozitára *apache* a *php5*
  - Skompilujeme *apache2* do inštalačného balíka
  - Nainštalujeme *apache2* ako service
  - Nainštalujeme *php5*
  - Zadáme do konzoly *apache2 –d*
  - Teraz máme aplikačný server *apache2* spustený ako démona
- Inštalovať pomocou *apt-get*/*appitude* manažéra
  - Do konzoly zadáme nasledovné riadky:
  - *apt-get install apache2*
  - *apt-get install php5*
  - *apt-get install libapache2-mod-php5*

- */etc/init.d/apache2 start*

Keďže už máme nainštalovaný aplikačný server, základná cesta pre web súbory je: */var/www/*. Pokiaľ chcete zmeniť túto cestu, respektíve pridať iný adresár pozrite sa na konfiguráciu aplikačného servera APACHE.

Upozornenie, aplikačný server APACHE je nadstavený na port 80, pokiaľ si želáte nadstaviť iný port pozrite sa na sekciu zmena portu.

Nadstavenie portu pre server APACHE

V prvom rade nájdeme súbor *ports.conf* alebo súbor *httpd.conf*. na servery Ubuntu sa tieto súbory nachádzajú v priečniku /etc/apache2.

- Otvoríme si súbor *ports.conf* pomocou editora *vi (vi ports.conf)*.
- Nájdeme riadok *NameVirtualHost *:80*.
- Nasledovaný riadkom *Listen 80*.
- Prepíšeme číslo 80 na nami požadovaný port
- Uložime súbor (*:wq*).

Keď chceme pridať viac portov na ktorých bude aplikačný server apache počúvať tak by konfiguračný súbor mal vyzerať nasledovne (Kód 1.)

```
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default
# This is also true if you have upgraded from before 2.2.9-3 (i.e. from
# Debian etch). See /usr/share/doc/apache2.2-common/NEWS.Debian.gz and
# README.Debian.gz

#Port 1
NameVirtualHost *:80
Listen 80

#Port 2
NameVirtualHost *:123
Listen 123

#Port n
NameVirtualHost *:456
Listen 456
```

Kód 1. Konfiguračný súbor ports.conf po pridaní n portov.

Podotýkam, že je nesmierne dôležité aby sa čísla portov pri *NameVirtualHost, Listen* zhodovali.

Aby bolo možné vidieť zmeny je potrebné apache reštartovať:
/etc/init.d/apache2 restart

Nadstavenie iných priečinkov ako východzích

Ako som už naznačil v kapitole 1.1 základný priečinok, ktorý server apache považuje za koreňoví je : *var/www/*.

Keďže my môžeme chcieť umiestniť priečinok inam uvažujme : /<*Somepath*>/<*SomeFolder*>

Vytvoríme priečinok <*SomeFolder*> nasledovne*:*

- *cd* /<*Somepath*>/
- Vytvoríme priečinok *mkdir* <*SomeFolder*>

- Zmeníme vlastnické práva *chmod 766 <SomeFolder> (drwxr-xr-x)*
- Zmeníme vlastníka*chown www-data:www-data <SomeFolder>*

Poznámka: používateľ www-data nemá heslo, preto odporúčame po nahratí dát zmeniť prístup nasledovne:
*chmod 666 <SomeFolder>/**

Po vytvorení priečinka treba ešte serveru povedať že má spúšťať skripty z iného priečinka. To sa vykoná nasledovne nájdeme priečinok *./apache2/sites-available* v tomto priečinku sú dva zaujímavé súbory *default* a *default-ssl*. Tieto je potrebné zmeniť (Kód 2.).

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost

    DocumentRoot "<SomeFolder>"
    <Directory />
        Options FollowSymLinks
        AllowOverride All
    </Directory>

    <Directory "<SomeFolder>">
        Options Indexes FollowSymLinks MultiViews
        AllowOverride All
        Order allow,deny
        allow from all
    </Directory>
</VirtualHost>
```

Kód 2. Potrebné zmeny v konfiguračnom súbore default.

Uvedomte si, že tieto zmeny vám zmenia správanie servera len pre port 80, pre ďalšie porty pridajte záznamy do konfiguračného súboru *default*.

```
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
    ServerAdmin webmaster@localhost

    DocumentRoot "<SomeFolder>"
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory "<SomeFolder>">
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
    </Directory>
</VirtualHost>
```

Kód 3. Potrebné zmeny v konfiguračnom súbore default-ssl.

Uvedomte si, že tieto zmeny vám zmenia správanie servera len pre port 80, pre ďalšie porty pridajte záznamy do konfiguračného súboru *default-ssl*.

Keďže už máme urobené zmeny potrebné pre zavedenie iných pracovných priečinkov alebo iných portov tak reštartujeme server:
/etc/init.d/apache2 restart

9.3 Inštalácia a konfigurácia pre Windows XP/7

Pre návod na inštaláciu pre Windows XP/7 pozri návod, ako nainštalovať balík XAMP.

9.4 Inštalácia a konfigurácia PGSQL

Inštalácia PGSQL pozostáva z nasledovných krokov: Inštalácia PGSQL servera, vytvorenie používateľa, konfigurácia PGSQL a naplnenie databázy údajmi.

Inštalácia PQSQL Servera

Dosiahneme zadaním :

- *apt-get install postgresql-9.1 postgresql-client-9.1 postgresql-contrib-9.1*

Pre manažment databázy odporúčam nainštalovať nasledovné balíky:

- *apt-get install pgadmin3 postgresql-client-9.1 postgresql-doc-9.1*

Poznámka: Inštalácia je dokončená, používateľ postgres je vytvorený, ale bez hesla. Preto odporúčam zmeniť heslo na PGSQL hneď po dokončení inštalácie.

Vytvorenie používateľa

Budeme pracovať s SQL konzolou PGSQL preto najprv zapneme konzolu pomocou príkazu:
sudo -u postgres psql

Vytvorenie používateľa: dosiahneme zadaním nasledovných dopytov:

```
CREATE USER <USERNAME>
CREATE ROLE <USERNAME>
ALTER ROLE <USERNAME> WITH ENCRYPTED PASSWORD '< SomePaswd >';
```

Kód 4. Vytvorenie používateľa pre databázu PGSQL.

Poznámka: Z PGSQL konzoly sa dostanete pomocou \q espace sekvencie.

Zmena hesla pre používateľa postgres je obdobná ako pre akéhokoľvek používateľa. Do PGSQL konzoly zadáte:

ALTER ROLE postgres WITH ENCRYPTED PASSWORD '<SomePaswd >';

Konfigurácia PGSQL

V prvom rade si nájdeme konfiguračný súbor *postgresql.conf*, keďže sme postgres inštalovali manuálne odporúčam na nájdenie použiť príkaz:
find / -name postgresql.conf 2>/dev/

Konfiguračný súbor obsahuje konfiguráciu pripojenia a bezpečnosti. Keďže z databázou nekomunikujú žiadne externé prostriedky tak nie je potrebné aby sme toto niekomu umožnili:
Pripojenie len z lokálneho systému teda od komentujeme riadok:
listen_addresses = 'localhost'

Poznámka: Komentáre v konfiguračných súboroch sú všetko, čo nasleduje za znakom mriežky. Komentáre v konfiguračných súboroch sa vzťahujú iba na text medzi mriežkou a znakom konca riadku.

Keďže aplikácia je multipoužívateľská, každý používateľ si vytvára pripojenie do databázy. Preto maximálny počet pripojení nadstavíme na 1500 (FIIT má asi 1200 študentov). Toto dosiahneme riadkom:

max_connections = 1500

Posledným podstatným parametrom je port ten nadstavíme tak, že zmeníme nasledovný riadok:

port = <číslo portu>

Ďalšie nadstavenia sú nepodstatné, výsledný konfiguračný súbor by mal mať zmenené 3 riadky. Avšak pre bezpečnostných „expertov" odporúčam použiť paranoidný mód bezpečnosti, ktorý je vyjadrený (Kód 5.).

```
listen_addresses = 'localhost'
port = 5432
max_connections = 1500
unix_socket_directory = '/var/run/postgresql'
unix_socket_group = 'www-data '
unix_socket_permissions = 0770

authentication_timeout = 1min
ssl = true
password_encryption = on
db_user_namespace = off
```

Kód 5. Nadstavenie databázy z ohľadom na maximálnu bezpečnosť.

Spustíme PGSQL servis zadaním nasledovného výrazu
/etc/init.d/postgresql-9.1 start

Naplnenie databázy

Pre jednoduchosť odporúčam použiť nástroj PGPHPAdmin. Inými prípadmi sa odmietam zaoberať. V prvom rade si potrebujeme vytvoriť databázu takže potrebujeme si spustiť SQL konzolu:

sudo -u postgres psql

Databázu vytvoríme zadaním príkazu:

CREATE DATABASE <dbname> ENCODING 'UTF-8' OWNER < rolename>;

Upozornenie: Použité kódovanie nesmierne povplivňuje použiteľnosť. Pre šikovných informatikov neodporúčam používať iné ako UTF-8 pre tých čo mi neveria skúste ANIS-1520.

Samotné naplnenie pozostáva z nasledovných krokov:

- Otvorenie vytvorenej databázy pomocou nastroja PGPhPAdmin
- Načítanie existujúceho exportu do databázy

Zmeny konfigurácie PHP5

Keďže sa na databázu pripájame pomocou mapovačou, je potrebné tieto mapovače povoliť v konfiguračnom skripte *php.ini*

Vykonajte nasledovné zmeny:

- Dokomentujete nasledovné riadky
  - *extension=php_pdo_pgsql.dll*

  extension=php_pgsql.dll
- Reštartujeme server:

- */etc/init.d/apache2 restart*

Po aplikácií týchto zmien sme pripravený na inštaláciu sýtemu.

Poznámka: na databázu sa dá pripojiť aj inak pozri kapitolu 5 – typy a triky.

9.5 Inštalácia a konfigurácia YII Frameworku

Systém je postavený na frameworku YII, teda je potrebné tento framework nainštalovať.

Inštalácia systému

Keďže predpokladám, že zdrojové súbory systému ste získali spolu s touto príručkou nebudem uvádzať spôsob obstarania kódu.

Prvá možnosť bez inštalácie yii frameworku

- Do Vami vytvoreného priečinka nakopírujte obsah zložky */system-student*
- Do priečinka na vyššej úrovni (cd ..) nakopírujte priečinok /yii

Druhá možnosť s inštaláciou Yii frameworku

- Z http://www.yiiframework.com/ stiahnite YII framework
- Nainštalujte ho podľa návodu http://www.yiiframework.com/tour/
- Vytvorenú aplikáciu následne premažte obsahom priečinka system-student

Pokiaľ ste postupovali správne výsledok by mal byť nasledovný:

```
drwxr-xr-x  www-data  www-data - system-student
drwxr-xr-x  www-data  www-data - yii
```

Kód 6. Správne nadstavené prístupové práva a vlastník

Nadstavenie databázy

Je potrebné nadstaviť pripojenie na databázu. To vykonáme nasledovne:

- Prejdeme do priečinka system-student/protected/config/
- Otvoríme súbor *main.php*
- Upravíme riadky podľa (Kód 7.)
- Uložíme zmeny.

```
'db'=>array(
        'class'=>'CDbConnection',
          'connectionString' => 'pgsql:host=localhost;
         port=<PGSQL PORT>;
        dbname=<DBName>',
        'emulatePrepare' => true,
        'username' => '<DBowner>',
        'password' => '<DBPASS>',
        'charset' => 'utf8',
          ),
```

Kód 7.Šablóna pre nadstavenie pripojenia na databázu

Ak ste všetko vyplnili správne tak po zadaní url http:/localhost/.../ by sa mal zobraziť systém z počitáča na ktorom beží server.

Poznámka: Ak sa vám napriek všetkej snahe nepodarilo nahodiť systém skúste si overiť správnosť údajov. Údaje ste nastavovali počas inštalácie APACHE/PGSQL (Kapitoly 1,2).

Skúste tiež reštartovať služby PGSQL/APACHE.

9.6 Inštalácia systému študent pre Windows XP/7

Inštalácia systému študent pre Windows je jednoduchá potrebujete nainštalovať dva balíky:

- XAMPP http://www.apachefriends.org/en/xampp-windows.html
- PGSQL http://www.postgresql.org/download/

Následne vykonajte nasledovné kroky

- Inštalácia XAMPP vám vytvorí priečinok *xampp\htdocs* Do tohto priečinku nahrajte adresáre:
  - *yii*
  - *system-student*
- Inštalácia PGSQL vytvorí používateľa postgres bez hesla. Pre vytvorenie používateľa s databázy použite kroky 2.2 a 2.3.
- Vykonajte kroky 2.4 a 2.5 pre naplnenie databázy.
- Vykonajte krok 3.2 pre nadstavenie databázy (Kód 6.)
- Reštartujte apache pomocou XAMPP konzoly

Poznámka: Ak ste všetko vykonali správne tak na adrese http://localhost/system-student/ sa vám zobrazí uvítacia obrazovka.

10. Používateľská príručka

Používateľskú príručku spísal tím Schedule of Pain.

10.1 Prihlásenie do systému

1. Po zobrazení stránky systému sa v pravom hornom rohu nachádza tlačidlo na prihlásenie používateľa.
2. Po kliknutí sa zobrazí stránka, kde používateľ zadá svoje meno a heslo do AIS.
3. Potvrdí zadanie kliknutím na tlačidlo *Prihlásenie*.
4. Po úspešnom prihlásení sa v pravom hornom rohu obrazovky nachádza namiesto tlačidla na prihlásenie tlačidlo na odhlásenie s menom aktuálne prihláseného používateľa.

10.2 Funkčný návrh časti Skupiny

Čo?

- Podporenie sociálneho aspektu tvorby rozvrhov – študent chce mať spoločné cvičenia s vybratými kolegami, študent chce zdieľať informácie o svojom rozvrhu.

Ako?

- Možnosť vytvárania sociálnych kontaktov v podobe priateľstiev respektíve skupín známych.
- Študent si môže vybrať, koho si pridá medzi známych, pričom pre zdieľanie informácií a akcií je nutné **obojstranné potvrdenie** vzťahu.
- Na vytvorenie vzťahu sa použije mechanizmus pozvánok, ktorý zároveň realizuje potvrdenie druhej (pozvanej) strany, dôsledkom čoho je vytvorenie kontaktu (priateľstva).
- Tento vzťah možno kedykoľvek zrušiť oboma zainteresovanými stranami.
- V dôsledku vytvoreného kontaktu je možné zobraziť detailnejšie informácie o známych, prístupné je **zdieľanie rozvrhových akcií** známeho, a súvisiacich zadaných priorít známeho.
- Umožnené je tiež zvýšenie šance získať spoločné cvičenie pre skupinu známych na báze kreditového systému prístupného jednotlivým študentom.
- Eliminujú sa rozdiely medzi množstvom dostupných kreditov medzi študentmi, ktorí si zvolia **zdieľanie kreditov** – každý vloží nejaké množstvo kreditov, následne sa vložené množstvo kreditov prerozdelí medzi dvoma alebo viacerými známymi.
- Zdieľania sú **viazané na jednu rozvrhovú akciu**, ktorú majú prístupnú všetci aktéri zdieľania.

- Je stanovený obmedzený počet známych zdieľajúcich kredity na jeden termín rozvrhovej akcie (napríklad polovica kapacity jedného cvičenia, a pod.) – zrejme pevne stanovené globálne.
- Zdieľanie je realizované mechanizmom pozvánok na zdieľanie – akceptovať musia obidve strany.
- Zo zdieľania je možné kedykoľvek vystúpiť.
- Každému študentovi, ktorý zdieľa kredity so známymi sa zobrazuje v jeho primárnom individuálnom výbere aj počet kreditov, ktoré sú relevantné pre výpočet rozvrhu – teda počet bodov vyplývajúcich so zdieľania so známymi. Dotknuté rozvrhové akcie sú **farebne odlíšené**.

Príklad:

- Študent A pozve pozvánku študentovi B, so žiadosťou o pridanie medzi známych.
- Študent B akceptuje pozvánku študenta A.
- V tomto momente je vytvorený vzťah medzi A a B, zdieľajú medzi sebou informácie o dostupných rozvrhových akciách (celkový rozvrh), a zadaných kreditových preferenciách.
- Študent B pozve pozvánku študentovi C, ten ju akceptuje.
- Študenti A, B, aj C majú k dispozícii konkrétnu rozvrhovú akciu R (cvičenie) predmetu P. Pričom A vyjadril preferencie vložením 100 kreditov na túto akciu, B 50 a C 10 v dôsledku svojich kreditových možností.
- Študent B požiada o zdieľanie kreditov na akciu R študenta A, ktorý žiadosť akceptuje.
- V tomto momente majú obidvaja študenti priradených k rozvrhovej akcii rovnaké množstvo kreditov 75 (počet celkovo vložených kreditov / počet známych).
- Algoritmus pri vyhodnocovaní rozvrhov nie je zaťažený počítaním so zdieľanými preferenciami, pracuje sa s každým študentom individuálne, zároveň je však vyššia pravdepodobnosť, že študenti A a B dostanú rovnaké cvičenie, keďže sa pri zoraďovaní študentov na cvičenie podľa kreditov majú priradené rovnaké množstvo kreditov.

Ak študent C požiada o zdieľanie kreditov na akciu R študenta B, a tento žiadosť akceptuje, rozloženie bodov bude vyzerať nasledovne: A: 53,3, B: 53,3, C:53,3. Výpočet výslednej hodnoty kreditov sa opäť realizuje podľa vzorca (súčet vložených kreditov známych / počet zdieľajúcich známych).