Slovenská technická univerzita

Fakulta informatiky a informačných technológií

Podpora dištančného vzdelávania v predmete Systémové programovanie a asemblery

Tímový projekt

(Technická dokumentácia produktu)

Členovia tímu:

Bc. Ľudovít Fülöp

Bc. Martin Lacko

Bc. Ivan Malich

Bc. Dalimír Orfánus

Bc. Ivan Straka

Pedagogický vedúci:

Doc. Ing. Pavel Čičák, PhD.

2003/2004

Tím č.10

Zadanie

Počet tímov: 2

Vedúci tímov: doc. P.Čičák, Dr. K.Jelemenská

Hlavný dôraz na vyučovanie v predmete SPA sa kladie na metodológiu tvorby a samotnú tvorbu programov na strojovej úrovni. V priebehu semestra študenti vypracúvajú niekoľko dielčích úloh a zložitejších zadaní, ktoré sú osobitne hodnotené. Cieľom je, aby každý študent pracoval samostatne, čo prináša práve do procesu hodnotenia istú náročnosť. Napr. ako podchytiť "kopírovanie" programov, minimálne zmeny v už existujúcich programoch a pod. Celý problém je o to zložitejší, čím väčší počet študentov práve absolvuje predmet. Preto sa zameriavame na vytvorenie programového prostredia, ktoré podporí prácu pedagóga riešením aspoň časti problémov, ktoré sú spojené s vyhodnocovaním zadaní z predmetu SPA.

Ďalším rozšírením sú:

- demonštračné programy,
- funkcie pre podporu výučby a samoštúdia prostredníctvom Internetu,
- funkcie podporujúce automatizované preverovanie vedomostí vrátane generovania a vyhodnocovania testov.

V tomto projekte úlohou pre tím bude:

- analyzovať základnú problematiku predmetu SPA,
- analyzovať požiadavky, na ktoré je kladený dôraz pri tvorbe a hodnotení úloh a testov v SPA,
- navrhnúť vhodné doplnenie - rozšírenie množiny požiadaviek na systém,
- navrhnúť vhodné demonštračné programy a funkcie systému pre podporu dištančnej výučby vrátane preverovania vedomostí,
- navrhnúť funkčný systém,
- implementovať navrhnutý systém.

Výstupom prvého semestra bude podrobná dokumentácia prvých troch bodov, hrubý návrh systému, ukážky práce vo zvolenom prostredí a predbežný plán práce na nasledujúci semester.

Obsah

Prílohy:

Zmeny v dokumentácii

Uvádzame tu zoznam zmien dokumentácie 2. etapy oproti dokumentácii z prvej etapy.

- Zjemnili sme prvú kapitolu Úvod a dolpnili Štruktúru dokumentu.
- Pridali sme novú kapitolu Analýza zadania a presunuli do nej pôvodnú kapitolu 2.7.
- Pôvodné kapitoly 2 a 3 sme zlúčili do jednej Analýza zadania (kap. 3).
- Špecifikáciu sme doplnili o kapitoly z pôvodnej kapitoly 5 Návrh. Dolpnili sme tam informácie k modulu manažmentu predmtetu.
- Do návrhu sme vložili prekreslenú architektúru systému.
- Pridali sme kapitolu o prototype (kap. 6) a používateľskú príručku k prototypu (Príloha C).

1 Úvod

Dištančné štúdium je modernou formou štúdia a výučby. Ako také obmedzuje fyzický kontakt medzi študentom a vyučujúcim na minimálnu možnú mieru a je vlastne akýmsi predkrokom k virtuálnemu štúdiu a výsledne k vzniku virtuálnych univerzít.

V dnešnej dobe sa na štúdium kladú iné nároky ako kedysi. Študenti a absolventi nemusia byť chodiacimi bankami údajov, ale najdôležitejšie je to, že sú sami schopní sa naučiť, resp. doučiť množstvo nových vecí. Pre prax je zaujímavejší absolvent, ktorý dokáže naštudovať nové technológie a pracovať s nimi, ako absolvent, ktorý je živým a statickým úložiskom údajov. Táto požiadavka na absolventov, najmä technických disciplín, vychádza z celosvetového trendu, kedy sa nové veci a myšlienky tvoria rýchlejšie, a súčasne sa vďaka fenoménu zvaného Internet aj takmer okamžite prešíria do celého sveta. Aj vďaka dištančnému vzdelávaniu je možné udržiavať krok s novými trendmi, myšlienkami, technológiami a pod. Univerzita takto môže ponúkať akési doškolenie ľudí z praxe o nových veciach, samozrejme za úplatu a takto si aj pomerne jednoducho „prilepšiť“.

1.1 Motivácia

Dištančné štúdium technologických predmetov má za úlohu poskytnúť študentovi aktuálnosť informácií v študovanom technologickom odvetví, kde nové znalosti vznikajú tak povediac každú hodinu. Práve takéto predmety vyžadujú vysoký stupeň aktuálnosti, ktoré klasická forma štúdia pomocou skrípt nedokáže poskytnúť. Napríklad skriptá v oblasti informatiky môžu byť ako-tak aktuálne v čase svojho vzniku na stole autora alebo skupiny autorov, ale kým sa dostanú do tlače a následne do distribúcie pre študentov prebehne dlhá doba, niekedy aj 2 roky. Po uplynutí takéhoto dlhého obdobia sa však informácie uvedené v skriptách stávajú veľmi neaktuálne, prípadne dávno prekonané vývojom. A práve v tejto oblasti môže podpora dištančného vzdelávania a e-vzdelávacích technológií priniesť výhodu v štúdiu.

Naším cieľom je vytvoriť **platformu** pre podporu dištančného štúdia na Fakulte informatiky a informačných technológií STU a teda nielen pre predmet Systémové programovanie a asemblery. Chceme využiť **moderné technológie** založené na báze XML. Tieto technológie nám umožnia zabezpečiť **znovupoužiteľnosť** základných komponentov, a teda poskytujú výhody pri vylepšovaní riešenia do budúcnosti. Unifikácia rozhrania pre technológie XML nám zároveň poskytuje možnosť využívať komponenty, ktoré vznikli v minulosti.

Na Fakulte informatiky a informačných technológií STU momentálne neexistuje takýto informačný systém a preto sa naším projektom pokúsime overiť moderné technológie, ktoré by tvorili súčasť základu tohoto systému.

1.2 Štruktúra dokumentu

Najprv sme vykonali analýzu zadania (kap. 2) a na základe nej určili základné vlastnosti navrhovaného systému, ktoré sme potom v analyzovali (kap. 3). Na základe analýzy zadania sme spravili špecifikáciu (kap. 4). Na základe špeifikácie sme spravili hrubý návrh systému (kap. 5) a následne jeho vybrané časti zprototypovali (kap. 6). Potom nasledovala zmena špecifikácie v letnom semsetri (kap. 7.1) a následne implementácia systému (kap. 7.2 až 7.9). Systém sa počas vývoja testoval o očom prinášame správu v kap. 8. Používateľská príručka je v kap. 10 a systémová príručka v kap. 11. Čo sme nestihli a čo sme sa naučili je zhrnuté v kap. 9. Na záver prinášame zhodnotenie celkového projektu (kap. 12).

2 Analýza zadania

Výsledky tejto analýzy nám ukážu, na aké existujúce riešenia a technológie sa máme zamerať a zanalyzovať ich za účelom možného využitia v našom produkte.

Zo zadania projektu vyplýva základna predstava o systéme z pohľadu zákazníka:

- bude podporou pre bežné činnosti pedagóga pri veľkom počte študentov,
- bude vedieť podchytiť plagiátorstvo,
- bude podporou pri vyhodnocovaní zadaní
- bude podporovať výučbu a aj samoštúdium prostredníctvom Internetu,
- bude podporovať preverovanie znalostí študentov.

Predmet SPA absolvuje každým rokom naraz vždy veľký počet študentov. Príprava testov, vyhodnocovanie testov, testovanie a zber zadaní vrátene kontroly plagiátorstva je preto časovo náročná úloha, navyše nie je možné pri tak veľkom počte odovzdaných zadaní vždy možné, aby ho mohol cvičiaci jednoducho odhaliť.

Produkt má byť samozrejme aj podporou pri vyučovacom procese a jeho rozhranie a práca s ním by mali byť jednoduché a teda podporovať prácu pedagóga a nie ho zbytočne zaťažovať zložitou administráciou.

Pedagóg, a tak isto aj študenti, majú záujem na spoločnej diskusii v súvislosti s preberanou látkou, ale aj predmetu ako celok. Pedagóga zaujíma odozva od študentov na priebeh predmetu a študentov zaujímajú odpovede na otázky ohľadne preberanej látky ale aj o predmete ako takom. Za týmto účelom bude pre obe strany veľkou výhodou možnosť diskusie vo fórme fóra.

Z tejto základnej predstavy o systéme je zrejmé, že výsledkom projektu má byť produkt z oblasti e-vzdelávania a teda náš produkt bude založený na Internete (Web-based). Vypracovali sme preto analýzu niektorých existujúcich systémov e-vzdelávania. Na základe tejto analýzy budeme mať lepšiu predstavu o našom produkte z pohľadu funkcií, ale aj napr. o architektúre a použitých technológiách, ktorých využitie spadá do úvahy v našom produkte.

2.1 Analýza niektorých existujúcich projektov e-vzdelávania

Táto analýza sa zameriava na existujúce projekty e-vzdelávania a poskytuje prehľad ich podstatných vlastností a funkcií. Našim cieľom bolo analyzovať samotné programové systémy a nie ich konkrétne nasadenie v praxi, aby sme získali čo najviac informácií o nich, keďže nie všade sa ich funkcionalita využíva. Na konci analýzy prinášame porovnanie týchto existujúcich projektov s naším hrubým návrhom.

Nájsť požadované informácie k nespočetnému množstvu portálov e-vzdelávania nebolo vôbec jednoduché. E-vzdelávanie používajú vo svete skoro všetky veľké firmy ako lacný prostriedok na dopĺňanie vzdelania ich pracovníkov a teda ako nástroj k zvyšovaniu ich kvalifikácie. Existuje nespočetné množstvo internetových firiem, ktoré ponúkajú množstvo kvalifikačných kurzov za úplatu. Nájsť informácie o architektúre a vlastnostiach týchto systémov je prakticky nemožné, keďže je to ich firemné tajomstvo a nemalé investície do vývoja takéhoto produktu.

Po zadaní kľúča „Content Management System“ (CMS) sme sa dostali veľmi rýchlo k projektom s otvoreným zdrojovým kódom, ktoré sú pomerne dosť rozšírené. Kedže sme sa sústredili na projekty s otvoreným zdrojovým kódom, nie je až taký problém získať potrebné informácie.

Analyzovali sme dva CMS systémy zamerané na e-vzdelávanie (Moodle a Claroline), pretože dokumentovanosť ostatných je veľmi slabá. Vybrali sme aj jeden portálový CMS systém (Drupal) pre porovnanie.

2.1.1 Moodle

Je to projekt s otvoreným zdrojovým kódom, domovská stránka projektu je http://www.moodle.org. Bol špeciálne navrhnutý na manažovanie vzdelávacích kurzov založených na Internete. Návrh je silne ovplyvnený progresívnymi myšlienkami teórie vzdelávania (sociálny konštruktivizmus a pod.).

2.1.1.1 Hotová funkcionalita

Moodle je veľmi prispôsobiteľný. Podporuje rôzne témy, ktoré zmenou CSS (Cascade Style Sheet) dokážu zmeniť vzhľad celej stránky. V nasledujúcom texte je niekoľko charakteristických prvkov. Nie sú tu uvedené všetky pre ich rozsiahlosť, čitateľovi odporúčame návštevu domovskej stránky projektu. Snažili sme sa vybrať základné vlastnosti.

Administratívne prvky:

- **Rozsiahly autentifikačný mechanizmus**: Podporuje aj „plug-in“ autentifikačné moduly. LADP, IMAP, POP3 a NNTP prihlasovanie. SSL a TLS je tiež podporovaný.
- **Personalizovateľnosť**: Používateľ si môže nastaviť časové pásmo, vlastný online profil, vlastný jazyk.
- **Nastaviteľnosť** tém prostredníctvom „plug-in“ mechanizmu (písmo, farby,...).
- **Rozšírovateľnosť** je zabezpečená „plug-in“ mechanizmom.
- Plná **kontrola prístupu** používateľa: Zaznamenávanie jednotlivých aktivít používateľa vrátane jeho príspevkov v diskusiách a pod.

- **Prehľad**: Zabudované nástroje na „online“ analýzu tried. Sú vždy dostupné s množstvom grafov a údaje sa môžu stiahnuť vo forme tabuľky programu Excel. Spätná odozva je poskytovaná študentom aby sa porovnali s priemerom v triede.

Všeobecné prvky:

- **Manažment kurzov**: Kurzy zoradené podľa témy. Obsah sa môže vytvoriť zabudovaným WYSIWIG (What You See Is What You Get) HTML editorom, atď.
- **Manažment zadaní**: Zbieranie zadaní, zaznamenávanie času odoslania zadania, študent si môže neskôr pozrieť hodnotenie zadania, učiteľ sa môže rozhodnúť, či sa zadanie môže znovu odovzdať po dopracovaní pripomienok, atď.
- **Diskusie**: Synchrónna textová komunikácia vrátane fotky z vlastného profilu. Podpora vnorených URL, emotikoniek a obrázkov. Všetky stretnutia sú ukladané za účelom ich spätného vyvolania.
- **Fórum**: Podporujú sa tieto formy fóra: len pre učiteľov, novinky na predmetoch, fórum otvorené pre všetkých a jedna diskusia na používateľa. Diskusia môže byť utriedená podľa veku príspevkov, podľa vnorenia, podľa „threadu“. Učiteľ môže riadiť diskusiu. Dajú sa časovo obmedziť jednotlivé diskusie.
- **Kvíz**: Databáza otázok, ktoré su znovupoužiteľné v rôznych kvízoch. Kategorizácia otázok. Limitovaný čas na kvíz. Možnosť opakovania testu. Náhodné rozmiestnenie otázok v teste. Otázky dovoľujú HTML a obrázky. Viac správnych odpovedí na jednu otázku.
- **Žurnál**: Je súkromný medzi učiteľom a študentom. Každá položka žurnálu môže byť adresovaná ako otvorená otázka pre všetkých. Odozva učiteľa je pripojiteľná k otázke a študent je upozornený cez „email“.
- **Zdroje**: Podpora zobrazovania obsahu ľubovoľného elektronického dokumentu (Word, Powerpoint, Flash, Video, zvuky a pod.). Súbory sa môžu na server cez WWW rozhranie uložiť, alebo ich na ňom priamo vytvoriť.
- „**Workshop“**: Hodnotenie posudkov rovnakou stranou a triedenie hodnotení. Podpora rôznych spôsobov triedenia.

2.1.1.2 Platforma

Moodle bol testovaný na týchto platformách: Linux, Unix, Windows, Mac OS X a Netware. Je naprogramovaný v PHP jazyku (verzia 4.1.0), teda je použiteľný všade, kde je na HTTP servery nainštalovaný tento jazyk. Tak isto použité relačné databázy s ktorými pracuje sú platformovo nezávislé. Využíva teda jazyky s otvoreným formátom: PHP, SQL a HTML.

2.1.1.3 Architektúra

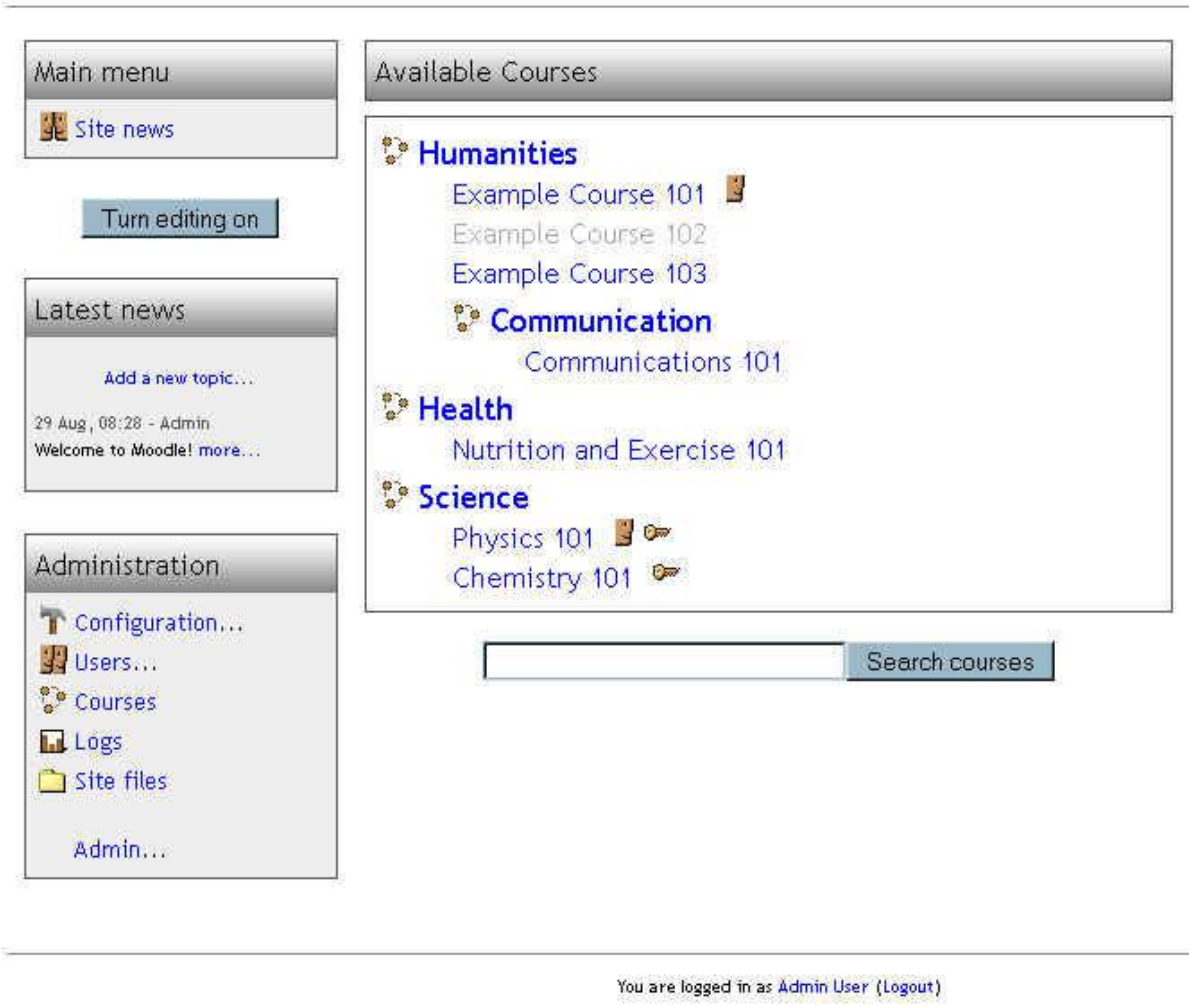
Modulárna architektúra, rozširovanie funkcionality je zabezpečené pridávaním dodatočných „activity“ modulov.

2.1.1.4 Použité štandardy

Okrem spomenutých technológií na modulárnu autentifikáciu (IMAP, POP3, NNTP, LDAP) a práci s relačnými databázami s jazykom SQL nevyžíva iné štandardy.

2.1.1.5 Celkové zhodnotenie

Je to veľmi progresívne sa vyvíjajúci projekt. Stojí za ním veľká komunita ľudí (učiteľov, administrátorov a jednotlivcov pracujúcich na univerzitách). Je lokalizovaný do 20 jazykov. Používateľom dovoľuje prispôsobovať si vzhľad. Je to hotový projekt, ktorý by sme mohli priamo použiť na naše účely a vhodne naplniť databázu jeho údajov. Treba mu vytknúť, že zatiaľ nepožíva moderné prostriedky e-vzdelávania založené na technológii XML (SCORM) i keď sa do budúcna plánuje s ich zahrnutím. Pohľad administrátora po prihlásení je na Obr. 1.



Obr. 1: Pohľad administrátora na systém Moodle pri prihlásení.

2.1.2 Claroline

Tento systém je tiež ako Moodle projekt s otvoreným zdrojovým kódom. Tento softvérový balík dovoľuje učiteľom vytvárať, spravovať a pridávať jeho kurzy prostredníctvom pavučiny. Cieľom jeho vývoja bola podpora dobrého vyučovania a vzdelávania a nie jeho nahradenie. Domovská stránka projektu je http://www.claroline.net/.

2.1.2.1 Hotová funkcionalita

Na domovskej stránke nie sú uvedené žiadne informácie tohto charakteru. Uvedenú funkcionalitu sme získali preštudovaním príručky pre učiteľa a študenta.

Administratíva:

- **Správa používateľov**: Klasická správa používateľov. Sledovanie prihlasovaní. Vytváranie skupín používateľov a prihlasovanie ich na kurzy.
- **Administrácia kurzu**: Možnosť aktivovať a deaktivovať položky pre študenta a napĺňať ich (dokumenty, dokumenty na štúdium, linky, video).

Všeobecné prvky:

- **Nástenka**: Aktuálne oznamy.
- **Agenda**: Kalendár plánovaných udalostí.
- **Diskusné fórum**: Pre danú tému fórum pre všetkých alebo len pre skupinu študentov.
- **Online diskusia**: Možnosť aktivovať diskutovanie pre skupinu počas riešenia problému.
- **Testovanie**: Testy sa vytvárajú cez formuláre, určia sa správne odpovede a spôsob hodnotenia.
- **Úlohy**: Vytváranie skupinových zadaní, ukladanie vypracovaných zadaní na server.

2.1.2.2 Platforma

Použitý programovací jazyk je PHP. Systém testovali na tejto platforme: Linux s HTTP serverom Apache a s MySQL databázou. Využíva služby mail servera Postfix alebo Sendmail. Tak isto ako Moodle používa jedine tieto jazyky s otvoreným formátom: PHP, HTML a SQL. Na strane klienta môže byť ľubovoľný WWW prehliadač.

2.1.2.3 Architektúra

V dostupnej literatúre nie sú o architektúre žiadne zmienky. Pre nedostatok času sme podrobne neanalyzovali zdrojový kód, aby sme zistili túto vlastnosť.

2.1.2.4 Použité štandardy

Okrem spomenutých žiadne iné.

2.1.2.5 Celkové zhodnotenie

Systém má tak isto niekoľko nadšencov po celom svete, ktorí pracujú na tomto projekte s otvoreným zdrojovým kódom. Na Slovensku ho používa Fakulta hospodárskej informatiky Ekonomickej univerzity (http://os.euba.sk/~learn) –viď Obr. 2. Funkcionalita je postačujúca na bežné vyučovanie. Samotný systém však nie je taký rozsiahly a porovnateľný vo funkcionalite k systému Moodle. Tak isto ako Moodle nepoužíva moderné technológie, ako napr. SCORM. Dokonca narozdiel od Moodle ani v najbližšej dobe neplánujú ich použitie: „ ... we are open to them but we think it is too early yet…“.

Obr. 2: Systém Claroline

2.1.3 Drupal

Dynamická WWW platforma, ktorá dovoľuje jednotlivcovi, alebo komunite používateľov publikovať, manažovať a organizovať rôznorodý obsah. Integruje veľa prvkov zo systémov manažmentu obsahu, kolaboratívne nástroje a softvéru pre diskusiu komunity. Domovská stránka projektu: http://drupal.org.

2.1.3.1 Hotová funkcionalita

Na podrobnejší výpis vlastností odporúčame domovskú stránku projektu.

Administratívne prvky:

- **Štatistiky**: Poskytuje štatistiky o počte prístupov, popularite jednotlivých obsahov, ako používatelia prechádzajú stránkami a pod.
- **Logovanie a podávanie správ**: Všetky dôležité udalosti a aktivity sa dajú zaznamenávať a ukladať do „log“ súborov.

- **Založené na WWW administrácii**: Drupal môže byť spravovaný aj cez WWW prehliadač.
- **Manažment používateľov**: Registrácia a autentifikácia môže byť lokálna alebo externá pomocou systémov Jabber, Blogger, LiveJournal alebo dokonca inou WWW stránkou systému Drupal.
- **Prístup podľa roly**: Administrátor jednoducho priradí práva jednotlivým rolám. Tak isto nastaví rolu pre používateľa ale aj pre skupinu a teda nie je nevyhnutné nastavovať práva pre každého používateľa dodatočne.

Všeobecné prvky:

- **Kolaboratívny dokument**: Tento jedinečný prvok dokáže nastaviť projekt alebo dokument tak, aby na jeho obsahu mohli pracovať autorizovaní používatelia.
- „**Online" pomoc**.
- **Diskusné fóra**: Všetky prvky diskusného fóra sú implementované.
- **Personalizácia**.
- **Vyhľadávanie**: Celý obsah stránky Drupal je úplne indexovaný.
- „**Site Cloud"**: Dovoľuje sledovať zmeny iných stránok, ktoré používateľ navštevuje a zaujíma sa o ne a sám dokáže zistiť ako boli posledne modifikované.

2.1.3.2 Platforma

Od začiatku bol navrhnutý tak aby bol platformovo nezávislý. Funguje na HTTP serveroch Apache a IIS a ich inštaláciách na týchto platformách: Linux, Unix, BSD, Solaris, Windows a Mac OS X. Nezávislosť nad rôznym databázovým softvérom (Oracle, MS SQL, MySQL, Postgres a iné SQL databázy) je umožnená vďaka abstraktnej databázovej vrstve.

2.1.3.3 Architektúra

Snahou vývojárov je, aby bol systém modulárny v maximálnej možnej miere. Modul v tomto systéme predstavuje súbor s PHP funkciami. Všetky tieto súbory sú v jednom adresári. Jadro systému je schopné vyvolať funkcie z modulov, ktoré sú exportované a vykonať ich v kontexte danej stránky.

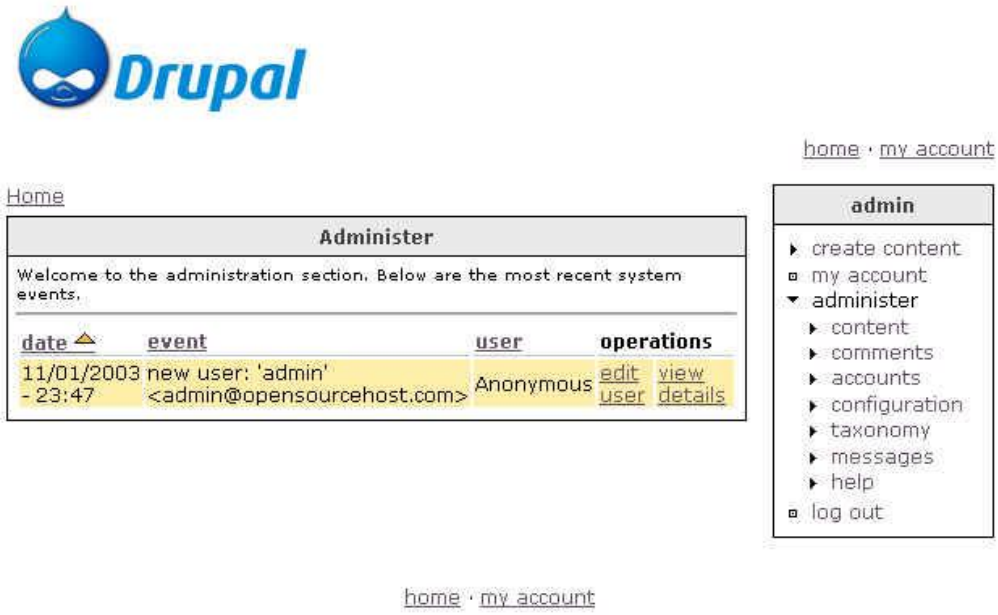
2.1.3.4 Použité štandardy

Okrem spomenutých (HTML, SQL a PHP) žiadne iné.

2.1.3.5 Celkové zhodnotenie

Systém sa dá použiť ako osobná WWW stránka, intranetová stránka spoločností, portál pre rôzne komunity (aj medzinárodné). Keďže je sním možné publikovať obsah rôznych dokumentov

a manažovať ho, je použiteľný aj pre potreby e-vzdelávania, i keď to nebol pôvodný zámer návrhárov. Preto systém nepodporuje technológie a jazyky na e-vzdelávanie (SCORM, SOAP, DocBook a pod.). Pohľad pre správcu systému je na Obr. 3.



Obr. 3: Pohľad správcu na systém Drupal.

2.1.4 Zhodnotenie

Za najprepracovanejší systém spomedzi analyzovaných považujeme Moodle. Je to rozsiahly projekt, dobre udržiavaný (dokumentácia) a najviac používaný (usudzujúc jeho až 20 jazykových mutácií). Tak isto sa teší nemalej podpore vývojárov a teda je to veľmi živý projekt. Spomedzi ostatných má najbližšie k realizácii (resp. je aspoň deklarovaná vôľa) štandardov na e-vzdelávanie na báze XML jazyka. Ďalej uvádzame zhodnotenie vlastností analyzovaných systémov, ktoré zvážime pri špecifikácii a návrhu.

Nezávislosť na platforme je prirodzenou vlastnosťou týchto systémov. Vyplýva to aj z faktu, že ako implementačný jazyk sa zvolil PHP.

Funkcionalita je podobná, majú aj spoločné prvky (manažment používateľov, pridávanie materiálov a organizácia štúdia a pod.). Rozširovanie funkcionality je zabezpečené prostredníctvom modulov. Každý zo systémov používa vlastný mechanizmus na integráciu modulov do systému.

Personalizácia je tiež základnou vlastnosťou. Cieľom je možnosť adaptovať systém do už existujúcich WWW portálov aby jeho vzhľad nebol v kontraste s iným stránkami. Na druhej strane je tu snaha, aby sa používateľ nerozptyľoval dizajnom, ktorý mu nemusí byť príjemný a mohol sa plne venovať učeniu.

Modularita je u projekov s otvoreným zdrojovým kódom veľmi často používaná. Dva analyzované projekty boli postavené modulárne (u Claroline sa nám to nepodarilo zistiť). Modularita je pri takýchto projektoch, kde participuje množstvo ľudí, najvhodnejším spôsob vývoja (paralelná práca ľudi na moduloch bez zásahu do práce).

Ani jeden z analyzovaných projektov nepoužíva jazyky ani protokoly na báze XML (SCORM, SOAP, DocBook, WSDL, UDDI), ktoré sa vo svete Internetu stávajú štandardom.

2.2 Požiadavky na systém

Na základe preštudovania zadania a niektorých existujúcich riešení a po diskusii so zákazníkom sme identifikovali základné prvky produktu:

- báza znalostí,
- testovanie znalostí,
- odovzdávanie zadaní,
- zisťovanie plagiátorstva,
- podora diskusie medzi študentami a pedagógom vo forme fóra,
- kontrola funkčnosti zadaní,
- jednoduchá správa používateľov (študentov a aj pedagógov).

Okrem týchto prvkov by mal mať náš systém nasledovné vlastnosti:

- modulárnosť,
- platformová nezávislosť,
- personalizácia rozhrania používateľa,
- použitie súčaných moderných techlógií.

Modulárnosť je veľmi dôležitá, keďže bude neskôr možné jednoducho dopĺňať funkcionalitu systému a navyše jednotlivé moduly systému môžu byť používané aj iným systémom (napr. aj informačným systémom fakulty). Docieli sa tak vysoká efektivita práce, keď každý modul bude ponúkať cez svoje rozhranie vlastnú funkcionalitu, ktorú bude môcť využívať ľubovoľná autorizovaná osoba (digitálna identita), resp. systém. Moduly by sa teda mali realizovať ako služby, ktoré sú dostupné z Internetu t.j. ako „web services“.

Platformová nezávislosť je dnes samozrejmosťou skoro každého riešenia založeného na Internete. Používateľovi nezáleží na akej platforme beží systém a že niektoré moduly sú integrované na rôznych platformách. Ďalej užívateľ systému nie je obmedzený operačným systémov, a môže systém prevádzkovať na ľubovoľnom, jemu vyhovujúcom systéme. Z hľadiska implementácie

a ďalšieho vývoja možno považovať platformovú nezávislosť zároveň aj za nezávislosť od implementačného prostredia. Za týmto účelom by sa mal využiť platformovo nezávislý jazyk ako napr. PHP, Java a pod.

Personalizácia je pre takýto systém príjemným, a používateľom často vyžadovaným doplnkom. Používateľ si grafické rozhranie systému prispôsobí podľa vlastnému vkusu a nebude tak rušený jeho vzhľadom. Táto požiadavka sa dá najjednoduchšie splniť použitím CSS štýlov.

Využitie moderných technológií na báze jazyka XML nám dovolí jednoducho vymieňať informácie medzi modulmi, ktoré môžu byť implementované nezávisle. Tak isto aj uchovávanie informácií vo formáte aplikácie XML sa ukazuje ako efektívny spôsob a začina sa v praxi čoraz viac realizovať. Príkladom môže byť napr. DocBook aplikácia XML. Tak isto by mal produkt zohľadniť existujúce štandarty v oblasti e-vzdelávania.

Analýza požiadaviek sa má zamerať na:

1. bázu znalostí,
2. testovanie znalostí,
3. odovzdávanie zadaní,
4. zisťovanie modifikovaných zadaní (plagiátstvo),
5. kontrola funkčnosti zadaní,
6. diskusné fóra,
7. „web“ služby (web services),
8. DocBook,
9. Štandarty pre e-vzdelávanie a na
10. digitálnu identitu.

3 Analýza požiadaviek

Nachádza sa tu analýza existujúcich riešení, ktoré by sme mohli v našom systéme použiť na základe požiadaviek na systém (kap. 2.2). Je efektívnejšie, keby sme využili časti, ktoré už vytvorili iní. Práve použitie prostriedkov z minulosti a ich zakomponovanie do moderného riešenia je cestou k vysokej efektivite pri vynaloženom úsilí.

3.1 Báza znalostí

Základom každého e-vzdelávacieho systému je databáza, v ktorej sú uložené všetky učebné materiály. Medzi hlavné požiadavky na takúto databázu patrí jednoduchý spôsob manipulácie s uloženými údajmi, keďže používatelia systému nemusia mať veľké skúsenosti s prácou na počítači. Báza znalostí by mala svojou funkcionalitou odrážať jej primárne použitie. Príkladom môžu byť databázy štandardizačných organizácií, ktoré kladú hlavný dôraz na množstvo vložených informácií (väčšinou sa jedná o informácie vo forme textu) a možnosti vyhľadávania v tejto báze. Naproti tomu výučbové systémy si vyžadujú možnosti členenia informácií do kapitol a podkapitol, rôzne spôsoby indexovania a v neposlednej miere aj možnosť práce s multimediálnym obsahom.

Implementácií báz znalostí je niekoľko. Majú však niekoľko nevýhod, medzi ktoré patrí ich tesná zviazanosť s platformou, na ktorej sú používané. Odráža sa to napríklad v používanej znakovej sade či formáte databázy. Veľmi často sú bázy znalostí zviazané s celým dizajnom aplikácie, kedy jej možnosti priamo vychádzajú z používateľského rozhrania a jeho vlastností. Vtedy je pomerne ťažné pridávať do systému nové vlastnosti, rovnako môže byť problémom aj zmeniť členenie databázy.

3.2 Testovanie znalostí

Nasleduje prehľad niekoľkých existujúcich „WBT - Web-based training“ riešení.

3.2.1 WebToTest

Cieľom je testovanie pomocou priameho písania programov. Použitím „web“ prehliadača sa študent prihlási do systému a dostane zoznam otázok. Otázky, na ktoré ešte neodpovedal sú označené „???“. Vybratím otázky sa zobrazia všetky informácie o otázke a formulár pre odpoveď. Stlačením tlačítka „submit“ sa znovu zobrazí zoznam otázok. Ak chce študent zmeniť odpoveď, tak je tu možnosť opätovného výberu už zodpovedanej otázky.

Systém WebToTest podporuje rozhranie na programovací jazyk. Je tu možné vytvoriť programový projekt. Projekt pozostáva zo súborov, ktoré možno editovať, kompilovať a spúšťať. Po vytvorení programu možno zdrojový kód prekopírovať do formulára odpovede. Testy sú pripravované

definovaním otázok a podmienok pomocou „web“ formulárov. Existuje viacero spôsobov, ako tak urobiť. Keď odpoveď na otázku pozostáva z časti zdrojového kódu, skúšajúci definuje časti programov, ktoré sa nachádzajú pred a po tomto kóde. Tieto tri časti sú spojené a je umožnená ich kompilácia so vstupmi, ktoré zadefinuje inštruktor. Výstup programu je porovnaný s výstupom, ktorý je špecifikovaný.

Po skončení skúšky si môže inštruktor prezrieť odpovede pomocou WWW prehliadača. Otázky môžu byť zobrazené v poradí, v akom si ich vyberal študent alebo podľa čísiel otázok. Pre každú odpoveď sa zobrazí odhad správnosti (správne, nesprávne alebo neurčito) a poskytne skúšajúcemu zmenu alebo zpresnenie hodnotenia. Tak isto je zobrazená celková štatistika správnosti zodpovedaných otázok na uľahčenie určenia výslednej známky skúšajúcim.

Systém WebToTest má väčšinu vlastností, aké by mal mať testovací systém :

- Automatické známkovanie.
- Bezpečnosť.
- WWW charakter systému.
- Flexibilnosť.

3.2.2 QUIZIT

QUIZIT [1][2] je systém, ktorý umožňuje inštruktorom pohodlnú tvorbu testov. Podporuje rôzne druhov otázok :

- multi-choice,
- áno/nie,
- priraďovanie položiek,
- slovná odpoveď.

Systém QUIZIT dovoľuje nelineárne kladenie otázok -nazýva sa to adaptívne (prispôsobivé) testovanie. Možno si predstaviť, že test (postupnosť otázok) má štruktúru stromu. Otázka predstavuje uzol. Cesty, ktoré vedú z uzla von, potom predstavujú možné postupnosti otázok, ktoré budú položené študentovi na základe jeho odpovedí. Keď študent správne odpovie na otázku, tak je mu zadaná ťažšia otázka. V prípade chybnej odpovede je mu zadaná ľahšia otázka.

Testy a otázky sa definujú pomocou špeciálneho jazyka QML (QUIZIT Markup Language). QML je veľmi podobný jazyku HTML (HyperText Markup Language). Tu sa nachádza príklad časti testu napísaného v jazyku QML:

```
<!DOCTYPE QUIZ SYSTEM "quiz.dtd">
<QUIZ ID="quizAB.sgml">
<TITLE>Quiz -- Unit AA</TITLE>
<INSTRUCTION>
<P>
```

```
Teraz môžete začať týmto testom. Na absolvovanie testu máte k
dispozícii 90 minút. Prosím nepoužívajte žiadne pomocné materiály.
</P>
</INSTRUCTION>

<QUESTION-GROUP>
<MCHOICE ID="q0001">
<DESCRIPTION>
<P>Kotrý štát vyhral v roku 1930 svetový pohár vo futbale?</P>
</DESCRIPTION>
<ANSWER VALUE="WRONG" LABEL="a"><P>Brazília</P></ANSWER>
<ANSWER VALUE="WRONG" LABEL="b"><P>Nemecko</P></ANSWER>
<ANSWER POINTS="10" LABEL="c"><P>Uruguay</P></ANSWER>
<HINT>
<P>Tento štát vyhral svetový pohár dva krát</P>
</HINT>
</MCHOICE>
</QUESTION-GROUP>
</QUIZ>
```

Na začiatku je úvodná hlavička a inštrukcie k otázkam. Potom nasleduje definícia otázky typu „multi-choice“ (odpovede a, b sú nesprávne, c je správna) a za správnu odpoveď je možné získať 10 bodov. K tejto otázke je aj pomôcka (hint). Podobne možno zadefinovať i otázky iných typov.

3.2.3 ATK

Systém ATK [3] bol vyvinutý v roku 2003 na FEI STU v rámci diplomovej práce. Obsahuje všetky nevyhnutné funkcie pre testovanie znalostí:

- jednoduché vytváranie rôznych typov otázok a testov,
- testovanie vedomostí študentov,
- manuálne alebo automatizované (v závislosti od typu otázky) vyhodnocovanie odpovedí,
- prezeranie a archivovanie výsledkov,
- zabezpečenie systému voči možnému podvádzaniu pri testovaní a neoprávnenému prístupu k otázkam a výsledkom.

ATK podporuje otázky nasledovného typu:

- *multi-choice:* K otázke je priradených viacero odpovedí, pričom za každú odpoveď je možné získať určitý počet bodov. Pre správnu odpoveď je to zvyčajne kladné číslo a pre nesprávnu odpoveď nula bodov alebo dokonca záporný počet bodov.
- *áno/nie:* kde odpoveďou na otázku je áno alebo nie. Za správne priradenie odpovede je pridelený určitý počet bodov.

- *slovná odpoveď:* Na otázku je potrebné odpovedať celým slovom. Otázka môže byť tvorená aj súvislým textom (resp. zdrojovým kódom programu), kde sú vynechané niektoré slová a úlohou študenta je ich doplniť. Učiteľ môže zadať aj slová, ktoré sa majú doplniť a študent si len vyberie správne umiestnenie slov. Pri vyplnení prázdneho miesta správnym slovom je študentovi pridelený určitý počet bodov.
- *jednoduchý program:* Učiteľ zadá špecifikáciu a úlohou študenta je vytvoriť program. Je možné zadefinovať aj zdrojový kód, ktorý sa nachádza pred, a za týmto programom. Pri vytváraní otázky je možné určiť, či študent bude môcť využiť kontrolu napísaného programu alebo kompilátor daného jazyka. Inštruktor bude manuálne hodnotiť tento typ otázky.

Dĺžka riešenia každej otázky môže byť obmedzená určitým časom. Po prekročení tejto doby sa znižuje hodnotenie podľa zadaného kritéria alebo test pokračuje ďalšou otázkou. Tak isto na vyriešenie celého testu môže byť pridelený určitý čas a po jeho prekročení môže byť znížené jeho bodové hodnotenie alebo ukončené testovanie.

Pre jednotlivé testy je možné priradiť študentov, ktorí budú mať k nemu prístup. Tak isto je možné určiť časové obdobie, počas ktorého bude môcť študent zahájiť testovanie.

Celý systém kladie i dôraz na bezpečnosť, a tým v akceptovateľnej miere zamedzuje možnosti podvádzania a neoprávnenému prístupu do systému.

3.2.4 Porovnanie riešení

Výhodou systému WebToTest je podpora programátorských otázok. Poskytuje testovanie programátorských schopností, ktoré sa nedajú overiť bežným spôsobom. Testy a otázky sa definujú interaktívnym spôsobom - pomocou formulárov. Tento spôsob predstavuje jednu alternatívu, ako je možné zadávať testy.

Systém QUIZIT je trochu jednoduchší a nie je stavaný na testy z predmetov programovania. Na druhej strane dovuľuje adaptívne testovania. Testy a otázky v systéme QUIZIT sú definované neinteraktívnym spôsobom - pomocou špeciálneho jazyka QML. Takéto zadávanie otázok možno považovať za prehľadnejšie a otázky sa dajú jednoducho znovu použiť aj v iných testoch. Navyše je tu možnosť použiť (resp. vytvoriť) jednoduchý editor pre tvorbu testov v QML.

Systém ATK je kombináciou vyššie uvedených testov. Podporuje adaptívne testovanie, rôzne typy otázok, vrátane programátorských, pri vytváraní otázok možno použiť všetky možnosti, ktoré poskytuje špecifikácia HTML. Ide o pomerne dobre prepracovaný systém, a keďže vznikol na pôde fakulty, rozhodli sme sa práve tento zakomponovať do nášho systému dištančného vzdelávania. Budeme priamo využívať tento systém po jeho modifikácii na podporu QML pri zobrazovaní otázok.

3.3 Odovzdávanie zadaní

Modul pre zber zadaní je určený na uchovávanie zadaní, dodávaných z vyššej vrstvy. Z hľadiska trojúrovňového modelu systému sa jedná o strednú úroveň, zabezpečujúcu rozhranie medzi logickým a fyzickým odovzdaním zadania. Z toho vyplývajú i požiadavky na jeho funkčnosť:

- Prijatie a uloženie zadania na základe identifikátora (študenta).
- Poskytnutie vybraného zadania, resp. zadaní podla požiadaviek.
- Odstránenie zadania.
- Evidencia rôznych verzií zadania.

Modul by mal rozlišovať zadania podla identifikátora študenta, predmetu, a čísla zadania. Špecifikácia typu súboru so zadaním, príp. spôsob odovzdávania viacerých zadaní je daný vyučujúcim na vyššej úrovni systému, a z hľadiska modulu nie je zaujímavý.

3.4 Analýza problematiky zisťovania modifikovaných zadaní

Na zisťovanie kopírovania a modifikovania programov bolo vytvorených viacero algoritmov, ktoré je možné podľa typu použitého algoritmu rozdeliť do dvoch skupín:

- algoritmy založené na technike počítania atribútov,
- algoritmy založené na štrukturálnej analýze programu.

Algoritmy, ktoré pracujú na princípe zisťovania štrukturálnej analýzy sú novšie a majú oveľa vyššiu efektívnosť pri zisťovaní kopírovaných zadaní v porovnaní s algoritmami založenými na počítaní atribútov, ako je popísané v [4].

Najčastejšími postupmi, ktoré študenti používajú pri modifikácii skopírovaných programov sú predovšetkým zmena komentárov (najefektívnejšie najmä pri zadaniach z JSI, kde na prvý pohľad je každý program rovnaký), zmena používateľského rozhrania (v prípade zadaní z JSI textových výpisov na obrazovku), premenovanie premenných, volania procedúr, rozdelenie, príp. spájanie procedúr. Na druhej strane, ak je text zadania rovnaký pre všetkých študentov sú isté podobnosti nevyhnutné (v JSI napr. prerušenia). Zvýšenie podobnosti zadaní môže byť taktiež spôsobené, že študenti vo svojom zadaní použijú procedúru, ktorá sa nachádza v študijných textoch, prípadne v návode assembleru. Preto by navrhovaný modul systému mal vedieť rozlíšiť medzi nutnými podobnosťami medzi programami a modifikáciou zadania. Je ale zrejmé, že každá modifikácia nemôže byť odhalená, pretože ak študent použije už hotové zadanie len ako inšpiráciu a vytvorí od základu vlastný program je pravdepodobnosť odhalenia veľmi malá. Vytvorenie nového, prípadne nezistiteľná modifikácia existujúceho programu si však vyžaduje hlbšiu znalosť daného jazyka. Naším cieľom je práve vytvoriť

modul schopný odhaliť kopírovanie a modifikovanie programov študentmi, ktorí chcú zneužiť prácu iných a prezentovať ju ako svoju vlastnú.

3.4.1 Techniky počítania atribútov

Tieto techniky sú založené na zisťovaní počtu parametrov zdrojového textu programu, ktoré sú najčastejšie modifikované. Líšia sa len počtom zisťovaných parametrov, prípadne mierou akou sa zistené údaje podieľajú na celkovom hodnotení podobnosti. Ich najväčšou spoločnou nevýhodou je, že zanedbávajú programovú štruktúru a tok riadenia. Najčastejšie zisťovanými parametrami programov sú:

- počet unikátnych operátorov,
- počet unikátnych operandov (identifikátor, návestia, „integer-y“, real, reťazce a null),
- počet obsadených riadkov,
- počet prázdnych riadkov,
- počet riadkov komentára,
- použité a deklarované premenné,
- všetky riadiace príkazy.

Popisy niektorých konkrétnych implementácii je možné nájsť v [4] a [5]. Ako sa už spomenulo, tieto algoritmy patria medzi menej účinné a preto sa nimi nebudeme ďalej zaoberať.

3.4.2 Techniky štruktúrovanej metriky

Princíp práce algoritmov založených na štrukturálnej analýze programu je možné rozdeliť do dvoch častí:

- Vyprodukovanie postupnosti informačných znakov (tokenov) pre každý program. Takto vytvorená postupnosť obsahuje informácie o štruktúre analyzovaného programu. Jednotlivé tokeny popisujú príkazy, prípadne bloky príkazov.
- Porovnanie súborov tokenov jednotlivých programov medzi sebou so zahrnutím špecifík jazyka, v ktorom sú analyzované programy napísané.

Medzi najúspešnejšie algoritmy štrukturovanej metriky patria Plague, Sim a Yap. Posledne menovaný algoritmus existuje vo viacerých verziách.

3.4.2.1 Plague

Algoritmus Plague [4] pracuje v troch fázach:

1. Vytvorenie štruktúrovaného opisu blokov príkazov (úloh) analyzovaného programu – tokenov.
2. Porovnanie tokenov dvojíc analyzovaných programov. Tie tokeny, ktoré sa nachádzajú v oboch programoch sa presúvajú do ďalšej fázy.
3. V poslednej fáze sa zhodné úlohy bližšie porovnávajú a vytvorí sa výstupný zoznam.

Použitie tohto algoritmu je účinné, ale ako je zrejmé z už popísaných fáz algoritmu je silne jazykovo závislý, písanie verzie pre každý ďalší jazyk je značne komplikované z dôvodu nutnosti novej konštrukcie syntaktického analyzátora. Aktuálne je Plague použiteľný na programy napísané v jazykoch Pascal, Prolog, Bourn Shell a Llam. Navyše existujú i účinnejšie algoritmy štruktúrovanej metriky.

3.4.2.2 Sim

Algoritmus Sim [4][5] používa na porovnanie zdrojových textov dvoch programov metódu zarovnania reťazcov. Pred samotným použitím tejto metódy je nutné vytvoriť z každého zdrojového textu programu súbor tokenov, pričom každý token reprezentuje operáciu, operand, príkaz, číslo, komentár alebo symbol. Napríklad príkaz

```
while ( n<5 )
```

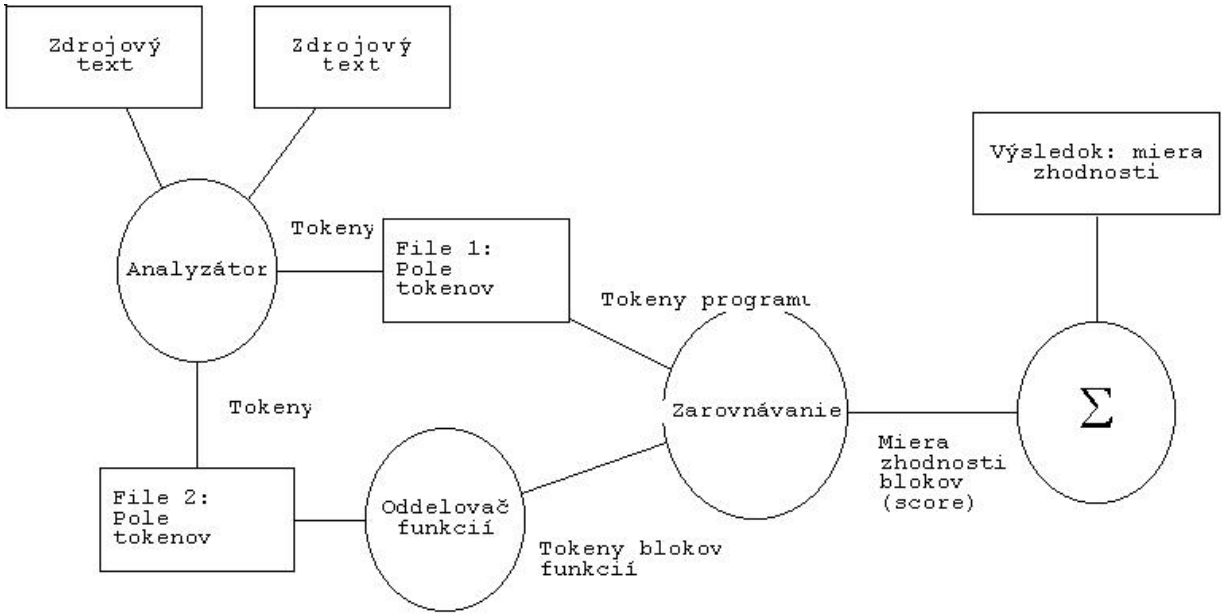
je nahradený tokom

```
TKN_WHILE TKN_LPAREN TKN_ID_N TKN_LESS TKN_5 TKN_RPAREN.
```

Tokeny pre príkazy, čísla, špeciálne symboly a pre všetky komentáre sú vopred definované. Ostatné tokeny (premenné, mená funkcií) sú prideľované počas vytvárania súboru tokenov. Informácie o typoch a počtoch tokenov sú ukladané do tabuľky symbolov. Tabuľka symbolov je zdieľaná pre oba analyzované zdrojové texty, takže dva výskyty dvoch tokenov sú zaznamenané jedným číslom. Zdrojový text je takto redukovaný do derivačného stromu, ktorý je oveľa kratší ako pôvodný text. Po dokončení tokenizovania je druhý program rozdelený na moduly, ktoré reprezentujú jednotlivé podprogramy. Každý takýto modul je potom samostatne zarovnávaný s tokenmi prvého programu. Zarovnávanie pracuje na princípe vkladania medzier do tokov čísel tak, aby sa získal reťazec s minimálnymi rozdielmi. Napríklad zarovnanie reťazcov "stars" a "masters" vyzerá nasledovne:

```
masters masters
sta rs stars
```

Tu je zrejmé, že môže existovať viacero možností zarovnania jednej dvojice reťazcov. V takom prípade sa vyberie možnosť, ktorá obsahuje najviac zhodných znakov (najmenšie rozdiely). V tomto prípade je to druhá možnosť. Celý algoritmus je znázornený na blokovej schéme na Obr. 4.



Obr. 4: Algoritmus metódy SIM

Výsledok sa vyráta podľa vzorca:

$$s = \frac{2 * zhoda(p1, p2)}{zhoda(p1, p1) + zhoda(p2, p2)}$$

pričom 'p1' a 'p2' sú zdrojové texty programov a 'zhoda' je výsledok zarovnávania dvoch programov. Výsledkom je číslo z intervalu <0, 1>, čo po vynásobení 100% dáva percentuálnu podobnosť dvoch analyzovaných zdrojových textov programov. Tento algoritmus je, ako je zrejmé z predchádzajúceho popisu, odolný voči všeobecným modifikáciám, ako sú premenovávanie identifikátorov, výmeny príkazov a funkcií, pridanie a odstránenie komentárov a medzier.

3.4.2.3 YAP (verzia 3)

YAP [4][5] je najprepracovanejší algoritmus na zisťovanie podobnosti zdrojových textov programov. Pôvodný návrh algoritmu YAP vychádzal z metódy Plague. YAP 3 je tretia verzia algoritmu, ktorá obsahuje nový porovnávací algoritmus Runing-Karp-Rabin Greedy-String-Tiling (RKS-GST), ktorý bol vytvorený na základe praktických skúseností s YAP verziami 1 a 2. V praxi sa totiž stáva, že pri modifikácii programov sú procedúry nielen rozdeľované (spájané), ale býva takisto zmenené poradie vykonávania nezávislých častí programu.

YAP 3 (rovnako ako pôvodná verzia YAP) pracuje v dvoch fázach. Pred začatím vykonávania prvej fázy sa zo zdrojového textu odstránia všetky komentáre a všetky veľké písmena v texte programu sa zmenia na malé. Samotná prvá fáza spočíva v nahradení zdrojového textu súborom tokenov (tokom celých čísel), pričom

- synonymá sú mapované na spoločnú formu (napr. strncpy je mapované na strcmp),

- funkcie sú usporiadané podľa volania, ktoré je označené identifikátorom FUN,
- vyhradené slová (tokeny, ktoré nie sú v lexikóne pre daný jazyk) sú vynechané.

Následne v druhej fáze sú porovnané súbory tokenov a výsledok predstavuje číslo od 0% do 100%, ktoré vyjadruje podobnosť zdrojových textov programov. Algoritmus druhej fázy je čiastočne odlišný pre každú verziu alegoritmu YAP. Výsledok je počítaný podľa vzorca:

```
match = (same - diff)/minfile - (maxfile - minfile)/maxfile
```

- 'minfile' je veľkosť menšieho programu,
- 'maxfile' je veľkosť väčšieho programu,
- 'same' je počet rovnakých prvkov,
- 'diff' je počet odlišných riadkov medzi dvoma rovnakými prvkami.

YAP 3 je navyše obohatený o algoritmus RKS-GST, ktorý je založený na hľadaný zhodných subreťazcov jedného reťazca v dlhom reťazci, pričom je definovaná minimálna dĺžka zhodnosti. Porovnávanie sa vykonáva vo viacerých cykloch tak, že sa hľadajú čo najdlhšie zhodné reťazce, ktoré sa následne označujú aby už neboli už porovnávané v ďalšom cykle. Po každom prechode cyklu sa znižuje dĺžka zhodnosti až po minimálnu dĺžku. Zhodné reťazce s dĺžkou menšou ako je minimálna, nie sú brané v úvahu.

Techniky modifikácie programov ako zmena komentárov, preformátovanie komentárov, zmena identifikátorov, zmena poradia operandov vo výrazoch, premenovanie premenných, zmena poradia nezávislých inštrukcií a nahradenie výrazov za ekvivalentné YAP 3 bez problémov odhalí. Zmeny ako modifikácie cyklov (zmena FOR za WHILE) alebo zmena volania procedúry vložením samotnej procedúry odhalí YAP 3 jedným odlišným tokenom, čo má minimálny vplyv na výsledok. V prípade kombinácie kopírovaného a vlastného programu je výsledkom miera podobnosti zdrojového textu týchto programov.

3.4.3 Zhodnotenie

Metódy založené na technike počítania atribútov sú založené len na spočítavaní príslušných vlastností daného programovacieho jazyka. Nie sú schopné analyzovať štruktúru zdrojového textu programu, a teda ani jeho prípadné zmeny. Na druhej strane sú implementačne jednoduché a pri zmene jazyka, v ktorom sú napísané analyzované zdrojové texty, stačí len určiť nový súbor vlastností jazyka.

Metódy založené na technike štruktúrovanej metriky porovnávajú okrem atribútov i zmenu štruktúry zdrojového textu, čiže porovnávanie je komplexnejšie a výsledok objektívnejší. Metóda Plague je veľmi silne jazykovo závislá, v súčasnosti implementovaná najmä na programy napísané v jazyku Pascal. Prispôsobenie metódy iným programovacím jazykom je dosť problematické. Metóda Sim je pomerne komplikovaná, súbory tokenov sa vytvárajú detailne v dvoch fázach a je konštruovaná

najmä na jazyk C. Navyše na zobrazenie výsledkov sa používa program Tcl/Tk. Metóda YAP 3 je z uvedených metód najjednoduchšia, najmenej jazykovo závislá a dáva najlepšie výsledky [4]. Navyše princíp jej činnosti je vhodný pre zdrojové texty programov napísaných v jazyku symbolických inštrukcií, ktorý sa používa v predmete Systémové programovanie a asemblery. Okrem iného je možné jednu implementáciu algoritmu bez problémov využiť aj pre iné jazyky.

V prácach [4] a [5] boli vytvorené konkrétne implementácie algoritmu YAP 3 na porovnávanie zadaní. Určité časti týchto implementácií je možné použiť i pri vytváraní nášho modulu na zisťovanie modifikovaných zadaní. Je nutné vytvoriť rozhranie na komunikáciu s riadiacim modulom a keďže implementácia v [5] je závislá na použitom operačnom systéme, je potrebné ju prepracovať tak, aby mohla byť použitá nezávisle od operačného systému. Spomenuté implementácie sú vytvorené v jazykoch C a Visual C++.

3.5 Kontrola funkčnosti zadaní

Testovanie je dôležitou súčasťou vývoja softvéru, už pri vývoji dokáže odhaliť závažné chyby a tým dopomôže k ich odstráneniu.

Nato aby sme program mohli otestovať sa musí najskôr preložiť do strojového kódu (skompilovať, zlinkovať) aby sme ho mohli spustiť, už v tejto etape testovania vieme povedať či obsahuje chyby. Ak neprebehne v poriadku, značí to že buď nie sú dodané všetky potrebné nástroje, knižnice, konfiguračné súbory na kompilácia programu, alebo sú v programe programátorské chyby, v obidvoch prípadoch je program nefunkčný a nepoužiteľný. Až potom môžeme zahájiť samotné testovanie programu. Sú rôzne techniky testovania, pre náš problém vyhovuje funkcionálne testovanie, ktoré je založené na metóde „čiernej skrinky“, tj. na základe špecifikácie programu. Cieľom je program otestovať na základe vzorky testovacích vstupov pričom sa neuvažuje vnútorná štruktúra programu.

Testovanie môžeme rozdeliť do viacerých fáz:

- kompilácia, linkovanie,
- spustenie, funkcionálne testovanie (otestuje či vstupno/výstupné správanie vyhovuje špecifikácií.

Existuje niekoľko spôsobov ako realizovať testovanie. Jednotlivé spôsoby väčšinou využívajú rôzne techniky simulácie reálneho prostredia, preto sa líšia najmä:

- vernosťou simulácie a teda aj presnosťou testov,
- stabilitou a odolnosťou voči programovým chybám,
- náročnosťou

- požiadavkami na systémové prostriedky,
- možnosťami testovania [6].

3.5.1 Spustenie programu v reálnom prostredí –hardvérové riešenie testov

Program pri preklade a spustení „beží“ v presne takom prostredí, na aké bol vytvorený. Z hľadiska vernosti simulácie je tento postup ideálny. Ak je hardvérovým spôsobom riešený reštart testovacieho počítača, a taktiež aj zadávanie vstupov a získavanie výstupov programu, tak nenastáva ani problém so stabilitou. Problém nastáva pri riešení zadávania vstupov a kontroly výstupov. Zadávanie vstupov je možné riešiť emulátorom klávesnice, pripojeným na klávesnicový vstup počítača, testovanie výstupov je náročnejšie, ale dá sa realizovať priamym prístupom k video pamäti, ktorý by vykonával modul, zasunutý v zbernici testovacieho počítača.

Riešenie je teda ideálne, pokiaľ ide o vernosť simulácie, na druhej strane je neúnosne drahé, lebo vyžaduje špecializovaný hardvér a vyhradený testovací počítač [6].

3.5.2 Spustenie programu v reálnom prostredí so softvérovou podporou testovania

Metóda spočíva v preklade a spustení programu na testovacom počítači a v hodnotení na tom istom počítači. Klávesnicové vstupy sú zadávané softvérovo, rovnako sú testované aj výstupy.

Softvérová podpora musí byť schopná prebrať kontrolu nad testovaným programov, prípade chyby, alebo nepovoleného zásahu do testovacieho prostredia. Toto riešenie si vyžaduje vyhradený testovací počítač.

3.5.3 Emulované prostredie – emulátory systému

Metóda je založená na spustení programu v emulovanom prostredí. Funkčnosť sa dá overiť len do tej miery akú dovoľuje emulované prostredie. Síce je tu zabezpečená bezpečnosť systému na ktorom beží emulované prostredie, ale je sú tu obmedzenia dané emulovaným systémov [6].

3.5.4 Emulované prostredie – emulátory hardvéru a hybridné riešenia

Emulátory hardvéru programovo emulujú celý výpočtový systém. Testovaný program pri takomto spôsobe práce beží v oddelenom prostredí a jednotlivé inštrukcie programu sa nevykonávajú, ale ich účinok sa simuluje. Výhodou takéhoto riešenia je absolútna bezpečnosť, daná princípom činnosti – program nijakým spôsobom nemôže zahltiť, poškodiť alebo zneužiť testovací systém. Ide pritom o najstabilnejšie prostredie po reálnom prostredí s hardvérovým riešením testov. Testy môžu prebiehať na ľubovoľnej platforme.

Emulátory hardvéru sú teda veľmi kvalitné, ale prílišná robustnosť je zaplatená nízkou rýchlosťou a veľkými nárokmi na pamäť. Hybridné riešenia nedostatočne odstraňujú tento hlavný handicap a sú technicky príliš zložité [6].

3.5.5 Zhodnotenie

Najideálnejšie riešenie je hardvérové riešenie, ale z pohľadu využitia výpočtovej kapacity toto riešení potrebuje vyhradený výpočtový systém. Nám sa ponúka použitie existujúcich riešení založených na emulovaných systémoch.

3.6 Diskusné fóra

Problematika diskusných fór je riešená v sieti Internet pomerne často. Dá sa povedať, že každý navštevovanejší informačný portál obsahuje aspoň minimálnu podporu na vyjadrenie názoru používateľov. Rozdiel medzi rôznymi implementáciami je hlavne v poskytovaných možnostiach komunikácie. Väčšina fór dovoľuje hierarchické štruktúrovanie správ, kedy môžu diskutujúci reagovať nie len na hlavnú tému, ale aj na reakcie iných používateľov. S tým sa potom viažu rôzne spôsoby zobrazovania a vyhľadávania v uložených správach. Niektoré systémy (ide hlavne o systémy pre menšie komunity) umožňujú aj dodatočnú úpravu príspevkov.

3.6.1 Slashdot.org

Portál Slashdot.org ponúka klasický príklad informačného portálu. Väčšinou však neposkytuje vlastné informácie, ale odkazy na zaujímavé články na iných serveroch. Okrem toho ponúka čitateľom aj možnosť vyjadriť sa k ponúkaným informáciám v diskusii k danému odkazu. Príspevky používateľov sú ukladané hierarchicky, čo dovoľuje meniť témy diskusie a širšie rozvinúť danú tému (typicky niekoľko stoviek príspevkov za hodinu).

Slashdot.org je postavený na vlastnom kóde napísanom v jazyku Perl. Kompletné zdrojové kódy sú voľne prístupné pod licenciou GNU GPL na stránke Slashcode (http://slashcode.com). Celý kód portálu je postavený modulárne, takže jeho jednotlivé časti je možné pomerne jednoducho použiť aj pri iných projektoch. Toto navyše uľahčuje rozdelenie na výkonnú logiku a zobrazovanie, kedy je zobrazovanie používateľského rozhrania riadené "témami". Nemenej zaujímavá je možnosť použiť implementované vyhľadávanie v príspevkoch, či rôzne spôsoby zobrazovania stromu príspevkov. Informácie zo systému Slashdot.org je možné okrem priameho prezerania v WWW prehliadači získať aj ako XML dokument a použiť napríklad vo vlastnom portáli.

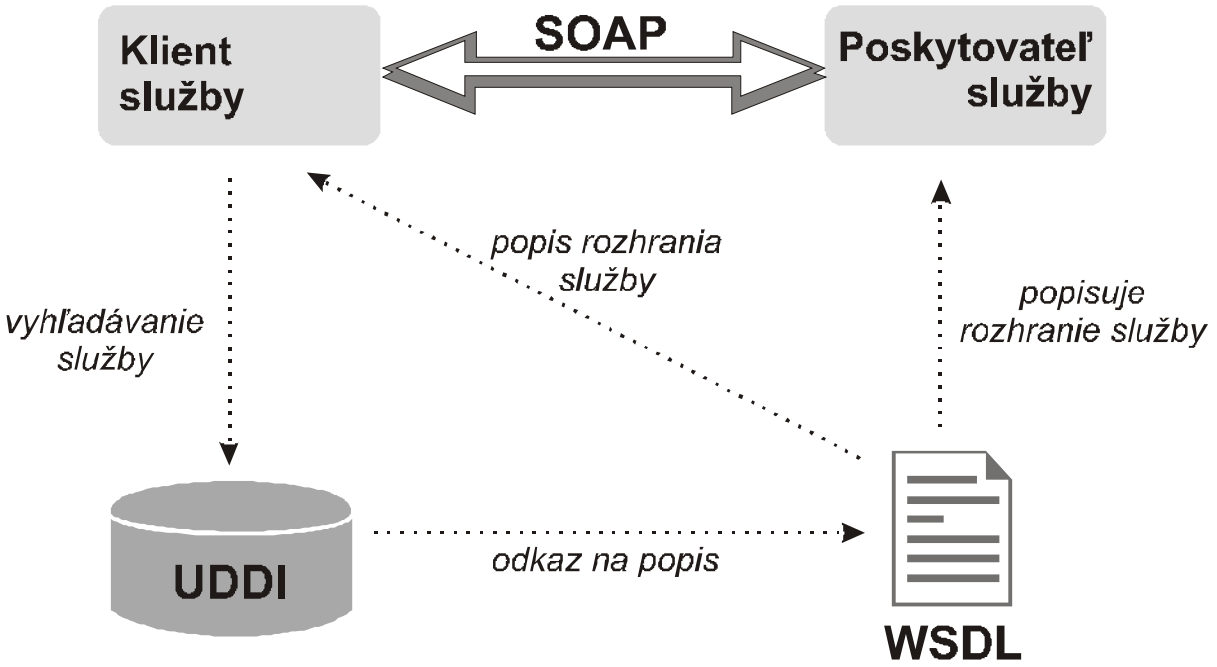
3.7 „Web“ služby

„Web“ služby umožňujú jednoduchú komunikácii medzi aplikáciami vo veľmi heterogénnom prostredí, pretože komunikácia je založená na platforme nezávislých štandardov – hlavne na jazyku XML a prenosovom protokole HTTP. Aplikácie si medzi sebou posielajú správy vo forme XML dokumentu, ktorý prenáša žiadosti a odpovede jednotlivých aplikácií. Celá infraštruktúra „Web“ služieb je postavená na troch základných technológiách [7][8][9][10]:

- SOAP (Simple Object Access Protocol) – protokol používaný pri komunikácii.
- WSDL (Web Services Description Language) – štandardný formát pre opis rozhrania „web“ služby.
- UDDI (Universal Description, Discovery and Integration) – štandardný mechanizmus poskytujúci registráciu a vyhľadávanie „web“ služieb.

Vzájomné vzťahy medzi týmito technológiami sú znázornené na Obr. 5. Ku každej „web“ službe má byť k dispozícii jej WSDL opis. Práve na základe tohto opisu sa ľahko vytvárajú SOAP požiadavky. Vo väčších systémoch je na distribúciu WSDL možné s výhodou využiť UDDI ako akýsi katalóg WSDL opisov. Klient, ktorý chce použiť „web“ službu, potrebuje k tomu jej WSDL opis. Tento popis môže získať buď priamo alebo prostredníctvom UDDI. Z popisu je potom klientovi zrejmé kam má poslať požiadavku na službu, navyše, je tam aj formát požiadavky.

Základom „web“ služieb je protokol SOAP využívaný na posielanie XML správ medzi aplikáciami. Štandardy WSDL a UDDI vznikli neskôr a rozširujú možnosti protokolu SOAP a jeho nasadzovanie v prostredí intranetu a Internetu. Jedna aplikácia vystupuje v úlohe klienta a druhá v úlohe poskytovateľa. Najrozšírenejším použitím protokolu SOAP je náhrada služieb na báze RPC. Klient pošle správu ako požiadavku poskytovateľovi a poskytovateľ vráti výsledok požiadavky klientovi.



Obr. 5: Vzájomné vzťahy medzi službami SOAP, UDDI a WSDL.

Prvá verzia (1.0) protokolu SOAP vznikla na konci roku 1999 ako výsledok práce firiem DevelopMentor, Microsoft a UserLand, ktoré sa snažili vytvoriť protokol pre vzdialené volanie procedúr (RPC) založený na XML. Táto verzia nadväzovala na jednoduchší (menej flexibilný) o rok mladší XML-RPC protokol. V priebehu roku 2000 sa protokol SOAP vo verzii 1.1 dostal pod správu konzorcia W3C. V dnešnej dobe je protokol vo verzii 1.2 určený na pripomienkovanie.

3.7.1 Štruktúra správy SOAP

SOAP správa má jednoduchú štruktúru. Ide o XML dokument s koreňovým elementom „**envelope**“, v tejto obálke podobne ako v HTML dokumente sú uzavreté dva elementy „**header“** (hlavička) a „**body**“ (telo). Hlavička je nepovinná a používa sa na prenos pomocných informácií na spracovanie správy (napr.: identifikácia používateľa, rola, autentifikačné, autorizačné informácie a pod.). Najdôležitejšou časťou je telo správy, pretože sa v ňom sa preášajú: Informácie identifikujúce volanú službu, predávané parametre služby alebo návratové hodnoty služby. Samotný SOAP má definovaný vlastný menný priestor (http://schemas.xmlsoap.org/soap/envelope/) ale samotná správa ich môže obsahovať niekoľko. Príklad jednoduchej SOAP správy:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV=
"http://schemas.xmlsoap.org/soap/envelope/"
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<SOAP-ENV:Body>
<m:GetLastTradePrice xmlns:m="urn:x-example:services:StockQuote">
<symbol>SUNW</symbol>
</m:GetLastTradePrice>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Príklad ukazuje možnosť získania burzovej informácie pomocou protokolu SOAP. Správa je veľmi jednoduchá neobsahuje hlavičku iba telo, V tele správy je požiadavka na volanie funkcie GetLastTradePrice s parametrom SUNW (kód firmy Sun Microsystems). V ďalšom príklade si uvedieme ako by mohla vyzerať SOAP odpoveď na predchádzajúcu požiadavku:

```
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
<SOAP-ENV:Body>
<m:GetLastTradePriceResponse xmlns:m=
"urn:x-example:services:StockQuote">
<Price>3.65</Price>
<Currency>USD</Currency>
</m:GetLastTradePriceResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

3.7.2 Transportný mechanizmus

Na prenos SOAP požiadaviek a odpovedí sa v dnešnej dobe používa HTTP protokol. Dôvodom je jeho jednoduchá implementácia a rozšírenosť. V tomto prípade WWW server obsluhuje prichádzajúce požiadavky a buď ich priamo sám spracováva alebo ich poskytuje ne externé spracovanie. Ďalšou výhodou HTTP je možnosť komunikácie bez zásahu do bezpečnostnej infraštruktúry, narozdiel od technológií DCOM a CORBA, ktoré vyžadujú vlastné špeciálne komunikačné rozhrania. Okrem toho je možné transparentné zabezpečiť šifrovanie prenášaných správ pomocou TLS/SSL

SOAP požiadavka sa zasiela v tele HTTP požiadavky použitím metódy POST, ktorá dovoľuje posielanie údajov v tele HTTP správy. HTTP požiadavka musí obsahovať hlavičku SOAPAction, ktorá identifikuje SOAP požiadavku. Túto hlavičku môžu potom bezpečnostné brány využívať na blokovanie prenosu a okrem toho môže obsahovať URI na identifikáciu obslužnej služby. Pokiaľ je obsah hlavičky prázdny je služba identifikovaná priamo adresou, na ktorú je smerovaná požiadavka.

Príklad SOAP požiadavky pomocou HTTP:

```
POST /StockQuote HTTP/1.1
Content-Type: text/xml; charset=utf-8
Content-Length: nnn
SOAPAction: ""
<SOAP-ENV:Envelope xmlns:SOAP-ENV=
"http://schemas.xmlsoap.org/soap/envelope/"
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<SOAP-ENV:Body>
<m:GetLastTradePrice xmlns:m="urn:x-example:services:StockQuote">
<symbol>SUNW</symbol>
</m:GetLastTradePrice>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Odpoveď pri prenose pomocou HTTP nenesie so sebou žiadne prídavné informácie. Odpoveď je identifikovaná pomocou MIME typu ako text/xml. Odporúča sa štandardné XML kódovanie natívnych znakov v znakovej sade UTF-8.

SOAP odpoveď prenášaná pomocou HTTP:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: 166

<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
<SOAP-ENV:Body>
<m:GetLastTradePriceResponse xmlns:m=
"urn:x-example:services:StockQuote">
<Price>3.65</Price>
<Currency>USD</Currency>
</m:GetLastTradePriceResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Okrem HTTP sa na prenos SOAP správ môžu používať aj iné protokoly. V praxi sa používa ešte protokol SMTP (Simple Mail Transfer Protocol) alebo „Javovská“ služba na posielanie správ JMS.

3.7.3 Web Service Desription Language

Jazyk WSDL [10] slúži k popisu sieťových služieb ako množiny koncových bodov zpracovávajúcich správy. Operácie a správy sú popisované na abstraktnej úrovni a až následne potom sú zviazané s konkrétnym sieťovým protokolom a dátovým formátom. To umožňuje jednoduché vytvorenie popisu rozhrania, ktoré poskytuje tú istú službu niekoľkými rôznymi spôsobmi. V praxi sa ale najviac využívajú na popis služieb na báze SOAP a prenosového protokolu HTTP.

WSDL vzniklo ako spoločná iniciatíva firiem Microsoft a IBM, ktoré si uvedomili potrebu zjednotenia jazyka používaného pre popis rozhraní „webových“ služieb. WSDL je v súčasnej dobe vydaný ako informatívna poznámka W3C a pracuje sa na vytvorení skutočného štandardu.

WSDL súbor s definíciou rozhrania služby je XML dokument. Skladá sa najmä z následujúcich elementov, ktoré tvoria základné časti každého WSDL popisu.

- **types** - Obsahuje definíciu dátových štruktúr používaných v správach. Zväčša sa používa XML schéma na definíciu aj keď to nie je nutné. Nástroje „web“ služieb sa postarajú o naviazanie dátových typov podľa XML schémy na natívne dátové typy použitého jazyka služby.

- **message -** Definuje formát správ pomocou definovaných dátových typov. Správy slúžia ako vstupné alebo výstupné štruktúry jednotlivých operácií. Každá správa sa môže skladať z viacerých pod častí s rôznym dátovým typom.
- **operation -** Abstraktná definícia operácií, ktoré služba podporuje. U operácie sa definuje aj aké má vstupné a výstupné správy.
- **portType** – Združuje dokopy viacero operácií.
- **binding** – Slúži naviazanie určitého portu (portType) na konkrétny protokol a formát prenášaných správ. V našom prípade zväčša HTTP a XML.
- **port –** Jeden koncový bod služby definovaný ako kombinácia sieťovej adresy a definovanej väzby (binding).
- **Service –** Združuje spolu viacero portov do jednej služby.

Nie je cieľom tejto analýzy urobiť kompletný opis možností WSDL. Krátky príklad ako môže vyzerať WSDL opis jednoduchej SOAP služby na zisťovanie kurzu je v Príloha A -.

3.7.4 UDDI

Vzhľadom na to, že nepredpokladáme zatiaľ využívanie služby UDDI, bude jej opis veľmi stručný a len informatívny.

UDDI poskytuje prostriedky na zaregistrovanie, kategorizáciu a vyhľadávanie „web“ služieb v rámci organizácie alebo Internetu. Jedná sa vlastne o akýsi katalóg poskytovaných služieb. Samotná služba tiež pracuje vo forme „web“ služby a teda poskytuje svoje údaje pomocou protokolu SOAP. UDDI register pracuje s nasledujúcimi druhmi entít:

- **podnikateľské entity (business entity)** – Každá spoločnosť má v registri uvedené základné identifikačné a kontaktné údaje. Katalogizácia môže byť vykonaná potom podľa geografickej oblasti alebo oblasti podnikania.
- **služby (business service)** – V registry je pri každej firme uvedený zoznam služieb, ktoré poskytuje a zoznam šablón služieb, prípadne technické údaje nutné pre využitie služby.
- **šablóny väzieb (binding template)** – Šablóny popisujú možnosti ako so službou komunikovať. Typicky sa jedná o odkazy na WSDL dokumenty služby. Šablóna okrem toho odkazuje na typ služby.
- **typu služieb (service type) –** Typ služby definuje abstraktnú službu a funguje ako obdoba rozhrania.(napr. niekoľko firiem poskytuje ten istý druh služby s rovnakým rozhraním).

Typický príklad práce s UDDI prebieha tak, že vývojár si nájde v UDDI registry vhodnú službu alebo služby, získa ich WSDL opis a potom ich začne rovno využívať.

3.8 DocBook

DocBook je štandard na písanie dokumentov na báze XML [9]. Napríklad armáda spojených štátov amerických požaduje odovzdávanie všetkej dokumentácie vo formáte DocBook, ale aj bežné vydavateľstvá požadujú od autorov aby svoje diela písali v tomto formáte. Aká je teda hlavná výhoda formátu DocBook oproti ostatným formátom?

DocBook definuje len obsahovú a štruktúrnu časť dokumentu a nezaoberá sa zbytočnými informáciami ako je štýlovanie, veľkosti fontov a podobne. Tieto informácie sú dôležité iba pre prezentačné účely, sú závislé od prezentačného média. Práve preto je postačujúce aby sa tieto informácie pridávali až v čase nutnosti prezentácie dokumentu v určitej forme (HTML, PDF). Tým odpadá nutnosť vyrábať z jedného formátu „odvodeninu“ iného formátu. Okrem toho je v dnešnej dobe bežné pre všetky oblasti, ktoré využívajú počítačové spracovávanie, že sa úlohy automatizujú. Pre počítače sú informácie o vizuálnom formátovaní irelevantné, dôležitá je štruktúra a obsah.

U dokumentov je kladený dôraz na:

- Generovanie výslednej dokumentácie v rôznych formátoch z jednej predlohy:
    - Tlačená forma (PostScript, PDF) .
    - on-line help (HTMLHelp, info, JavaHelp).
    - WWW prezentácia dokumentácie (HTML stránky) .
- Možnosť využiť schopnosti jednotlivých výstupných formátov (obsah, register, vyhľadávanie „fulltext“ a pod.).
- Podporovať časté zmeny v dokumente:
    - Technológie sa rýchlo vyvíjajú.
    - Automatizácia generovania jednotlivých výstupných formátov.
- Bez limitov:
    - Podpora národného prostredia.
    - Neobmedzovanie celkovej dĺžky dokumentu.
    - Fixná štruktúra dokumentov.

Práve tieto požiadavky sú kladené aj na študijné materiály v rámci dištančného vzdelávania a teda formát DocBook spĺňa tieto požiadavky. Okrem DocBook formátu spĺňa tieto požiadavky aj viacero iných formátov ale tieto sú zväčša proprietárne.

Proces prípravy dokumentu vo všeobecnosti prebieha v týchto troch fázach.

1. *Editovanie a vytváranie* – V tejto fáze autor aktívne vytvára dokument (knihu, skriptum, článok, dokumentáciu a pod.). Tento dokument potom opravuje, koriguje, schvaluje a pod.
2. *Publikovanie* – v tejto fáze dochádza k samotnej vizuálnej prezentácii dokumentu vo výslednej forme (HTML stránka, PDF, tlačená kniha,...). Vzhľadom na zachovanie jednotného vzhľadu v rámci organizácie sa používa automatická konverzia do výstupného formátu. Práve automatické generovanie prináša výhodu možnosti častých korekcií a vylepšovaní dokumentu.
3. *Prezeranie* – Je konečná fáza dokumentu, kedy je dokument poskytnutý čitateľovi za účelom čítania. Z tejto fázy sa potom prostredníctvom spätnej väzby môže znova iniciovať celý cyklus dokumentu.

Kedže dokument sa môže počas svojho životného cyklu často meniť je vhodné mať k dispozícii systém na správu dokumentov. Takéto systémy sú pre proprietárne formáty dokumentu (MS Word DOC) veľmi drahé. V prípade XML dokumentov vo formáte DocBook sa dá na tento účel použiť napr. nástroj pre správu verzií (Concurrent Vesion System -CVS).

3.8.1 História

DocBok je momentálne po HTML najrozšírenejší formát jazykov SGML a XML. Vznikol v roku 1991 ako následovník formátu SGML určeného primárne pre výmenu „unixovej“ dokumentácie. V roku 1999 sa správa formátu DocBook presunula pod združenie OASIS. DocBook má dvé možné syntaxy: SGML alebo XML. SGML sa využíva skôr z historického hľadiska, pre naše účely sa budeme zaoberať len XML syntaxou.

Vo svete je DocBook používaný najmä na tvorbu dokumentácie pre softvérové a hardvérové aplikácie, ale aj na publikáciu kníh a príručiek prípadne článkov. Nakladateľstvo O'Reilly používa DocBook na prijímanie publikácií, Sun Microsystems využíva odnož DocBooku (SolBook) na tvorbu a publikáciu dokumentácie svojich produktov pričom výstupy sú vo forme HTML, PDF a v tlačenej verzii.

Samotný DocBook nie je nič iné ako DTD definícia, ktorá definuje elementy a atribúty, ktoré sú povolené v DocBook aplikácii XML dokumentu. Táto definícia sa stále vyvíja a vylepšuje, momentálne je aktuálna verzia 4.2.

3.8.2 Formát DocBooku

DocBook je založený na XML to znamená, že každý element musí byť uzavretý štítkami (tags). V tomto ohľade je to veľmi príbuzné formátu HTML.

Na kódovanie znakov lokálnej abecedy sa v XML používa znaková sada ISO 10646, a kódovanie UTF-8 alebo UTF-16, ale je možné používať aj iné, aj keď sa to neodporúča.

Príklad hlavičiek XML dokumentu bez uvedenia kódovania predpokladá sa UTF-8:

```
<?xml version="1.0"?>
```

S uvedením kódovania napr. windows-1250.

```
<?xml version="1.0" encoding="windows-1250"?>
```

Okrem tejto hlavičky sa predpokladá ešte uvedenie typu dokumentu a miesto kde je umiestnené DTD dokumentu.

```
<!DOCTYPE book PUBLIC '-//OASIS//DTD DocBook XML V4.2//EN'
'http://www.oasis-open.org/docbook/xml/4.2/docbookx.dtd '>
```

Príklad jednoduchého DocBook dokumentu:

```
<?xml version='1.0' encoding='windows-1250'?>
<!DOCTYPE book PUBLIC '-//OASIS//DTD DocBook XML V4.2//EN'
'http://www.oasis-open.org/docbook/xml/4.2/docbookx.dtd '>
<book lang="sk">
<bookinfo>
<title>Príklad knihy</title>
<author>
<firstname>Jozef</firstname>
<surname>Mrkvak</surname>
</author>
</bookinfo>
<preface>
<title>Úvod</title>
<para>Odstavec textu.</para>
<para>...</para>
</preface>
<chapter>
<title>Prvná kapitola</title>
<para>Text prvej kapitoly</para>
<para>...</para>
</chapter>
<chapter>
<title>Druhá kapitola</title>
<para>Text druhé kapitoly</para>
<para>...</para>
</chapter>
<appendix>
<title>Prvá príloha</title>
<para>Text prílohy</para>
<para>...</para>
</appendix>
</book>
```

Priamo z dokumentu vidíme určitú podobnosť s HTML. Je to veľmi jednoduchý dokument s jednoduchou štruktúrou:

- **book** – Hlavný element dokumentu, v ktorom je celý obsah, niekedy býva nahradený ekvivalentným tagom „**article**“ (článok), „**qandaset**“ (FAQ) alebo „**refentry**“ (referenčné

stránky). Typ použitého „tag-u“ poukazuje na typ dokumentu. Ako atribút „tag-u“ býva uvedený jazyk v ktorom je dokument napísaný.

- **bookinfo –** element s meta-informáciami o dokumente (nadpis, podnadpis, autor, venovanie, tiráž, copyright, vydavateľ, abstract ...)
- **preface** – úvod, predhovor.
- **chapter** – kapitola. Miesto kapitol sa využíva aj štruktúrovanie na pomocou sekcií (sect1-sect5) prípadne bez uvedenia exaktnej úrovne (section).
- **appendix -**príloha.
- **para** – odstavec, iné typy odstavcov „**simpara**“ a „**formalpara**“.
- **orderedlist** – očíslovaný zoznam položiek.
- **itemizedlist** - zoznam s odrážkami .

Okrem týchto elementov a „tag-ov“ je ich v DocBook-u definovaných oveľa viacej. Ide najmä o „tag-y“, ktoré upresňujú svoj obsah napríklad „**programlisting**“ – časť programu ale aj iné. Tieto elementy elementy uľahčujú zpracovanie a výslednú vizualizáciu dokumentu. Podrobnejší popis jednotlivých elementov je v literatúre (http://www.kosek.cz/xml/db/)

3.9 Štandardy pre e-vzdelávanie

Rešpektovanie existujúcich štandardov v oblasti e-vzdelávania je nutnosťou na zachovanie interoperabilty medzi rôznymi systémami na vzdelávanie a dištančné štúdium.

V oblasti e-vzdelávania je k dispozícii viacero štandardov (AICC, IMS, LOM, ADL, SCORM,...) práve tieto štandardy sú natoľko silno etablované, že ich podporujú aj firmy známe vydávaním vlastných proprietárných štandardov.

3.9.1 AICC

Je jedným z prvých štandardov v oblasti e-vzdelávania. Vytvorila ho profesná organizácia AICC (Aviation Industy CBT Commitee). Štandard AICC špecifikuje proces výmeny výučbových materiálov medzi systémami a kurzami, ako aj systém uchovávania výsledkov štúdia.

3.9.2 IMS

IMS Global Learning Consortium združuje viacero spoločností (cca 150) s cieľom navrhovať platformovo ale aj podľa odborov nezávislé štandardy ma výmenu dát v oblasti e-vzdelávania na báze XML. V dnešnej dobe už existujú špecifikácie pre výmenu meta-údajov o výukových materiáloch, o študentoch a systémami elektronického testovania znalostí. Báza formátov ešte nie je úplne

dopracovaná, ale tvorba štandardov je relatívne veľmi rýchla a odráža reálne potreby v oblasti e-vzdelávania. Špecifikácie štandardov sú voľne k dispozícii (http://www.imsproject.org/).

3.9.3 IEEE

Medzi rešpektované štandardy v tejto oblasti patrí aj IEEE Standard for Learning Object Metadata (štandardy 1484.12.xx od Learning Technology Standards Committee). Úplné znenie tohto štandardu je ale prístupné len za poplatok, ale tento štandard sa stal čiastočne súčasťou štandardu SCORM.

3.9.4 ADL – SCORM

Okrem konzorcií z prevažne komerčnej sféry sa prejavuje aj otvorený štandard ADL (Advanced Distributed Learning Initiative – http://www.adlnet.org). Tento zastrešuje americké ministerstvo obrany. ADL si dáva za cieľ pôsobiť ako štandard preklenujúci most medzi komerčnými a akademickými štandardmi (IEEE, AICC, IMS) a typickými štandardizačnými organizáciami (ISO, W3C). Výsledkom tohto procesu je súbor doporučení a špecifikácii SCORM (Sharable Content Object Reference Model).

SCORM vychádza z existujúcich štandardov IMS, IEE a AICC a je udržiavaný tak, aby bol trvalo kompatibilný s týmto štandardami. Jedná sa o veľmi presné špecifikácie všeobecných princípov na vysokej úrovni (najmä XML špecifikácie).

3.10 Digitálna identita

Internet zahŕňa obrovské množstvo rôznorodých pripojených uzlov a sietí rôznych organizácií, od malých lokálnych sietí spájajúcich niekoľko počítačov až po rozľahlé siete poskytovateľov. Každý poskytovateľ alebo systém implementuje vlastný spôsob autentifikácie používateľov a vyžaduje od používateľov rôzne osobné dáta. Každá služba vlastne vytvára vždy novú a novú identitu používateľa. Používateľ je potom vystavený nutnosti nastaviť si vždy znovu svoj osobný profil a pamätať si ďalšie prihlasovacie meno (login) a jemu prináležiace heslo, prípadne inú autentifikačnú informáciu.

Opačným extrémom je, že používateľ má všade rovnaké heslo a stúpa možnosť kompromitácie vzhľadom na to, že niektorí poskytovatelia a systémy nezaručujú bezpečné uloženie hesiel. Heslá sú v súboroch alebo databázach v otvorenej forme, pri kompromitácii služby má útočník k dispozícii heslá všetkých používateľov a tak isto aj určitú časť ich osobných údajov. Množstvo osobných údajov, ktoré používateľ poskytuje službe, je rôzne a často služba požaduje od používateľa také osobné údaje, ktoré nepotrebuje k svoje činnosti.

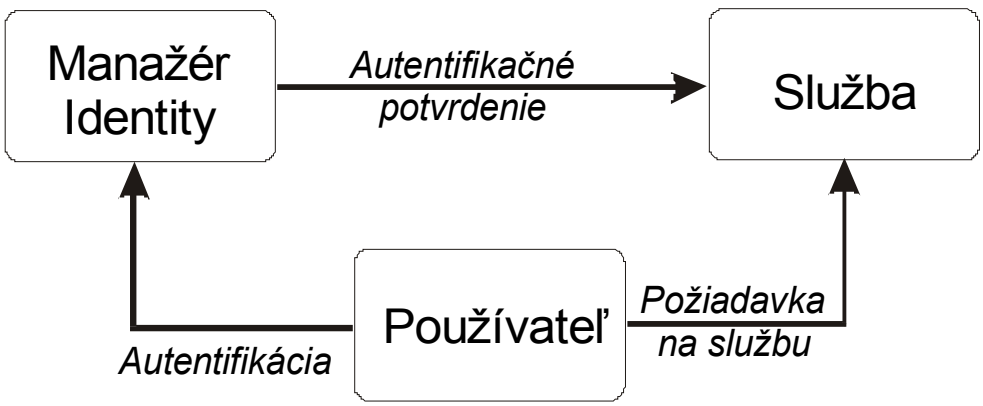
Riešením tohto problému by bolo zavádzanie čiastočne centralizovanej databázy používateľov, kde by boli ich profily, autentifikačné schémy a osobné údaje. Používateľ by manažoval iba prístup jednotlivých služieb k týmto údajom.

3.10.1 Jednotné prihlásenie

S narastajúcim množstvom služieb vo firemných sieťach, a teda z toho vyplývajúcej nutnosti používateľov sa autentifikovať, vznikli systémy na jednotné prihlasovanie sa k niektorým službám. Používateľ sa autentifikuje na svoju pracovnú stanicu alebo na firemný portál, autentifikácia sa udeje na centrálnom serveri firmy. Keď potom pristupuje na iné služby v sieti, je automaticky autentifikovaný pomocou centrálneho serveru. Nevýhodou týchto systémov jednotného prihlasovania je uzatvorenosť celého riešenia a teda obmedzenie iba na určité služby v sieti. Končí to tým, že vo firme je nasadených viac systémov s jednotným prihlasovaním, ale chýba spolupráca medzi týmito systémami. V dnešnej dobe vznikajú systémy digitálnych identít, ktoré ponúkajú aj možnosť jednotného prihlasovania.

Vo firemných sieťach čiastočne rieši jednotné prihlasovanie sa k službám Kerberos [11] alebo jeho modifikácie, ale pre globálne prostredie nie je Kerberos vhodný. Riešením je využívanie prihlasovacích technológií na báze XML.

Každý systém digitálnej identity implementuje alebo špecifikuje vlastnú metódu pre jednotné prihlasovanie. Spoločnou črtou viacerých implementácií je, že používateľova identita a autentifikačné informácie má manažér identity (Obr. 6). Používateľ sa najprv autentifikuje manažérovi identity a potom pri prístupe na službu je autentifikovaný manažérom identity pomocou autentifikačného tvrdenia (Authentification Assertion). Služba, ku ktorej používateľ pristupuje, môže byť v inej organizácii ako manažér identity. Preto je nutné aby medzi nimi existovala istá úroveň dôvery. Každý systém digitálnej identity rieši tento problém iným spôsobom. Väčšina systémov je vyvinutá pre prostredie založené na protokole HTTP [12] a na výmenu jednotlivých informácií využívajú práve výhody tohto prostredia [13].



Obr. 6: Zjednodušená schéma jednotného prihlásenia.

3.10.2 Používateľské profily

Jednou z pridaných hodnôt digitálných identít je možnosť vytvárať a ukladať používateľský profil. Používateľský profil obsahuje stručnú charakteristiku používateľa (napr. osobné údaje, obľúbená farba, šťastné číslo, veľkosť topánky, záľuby, ...). Systémy, ktoré majú oprávnenie čítať všetky alebo niektoré údaje z profilu, potom používateľovi poskytujú údaje a formu takú, ako jemu vyhovuje. Používateľ si sám manažuje prístup k jednotlivým údajom. V systémoch digitálnych identít sa poskytovaním a správou profilov zaoberá manažér identity. Manažér identity zabezpečuje výmenu profilových informácií s používanou službou [13].

3.10.3 Security Assertion Markup Language

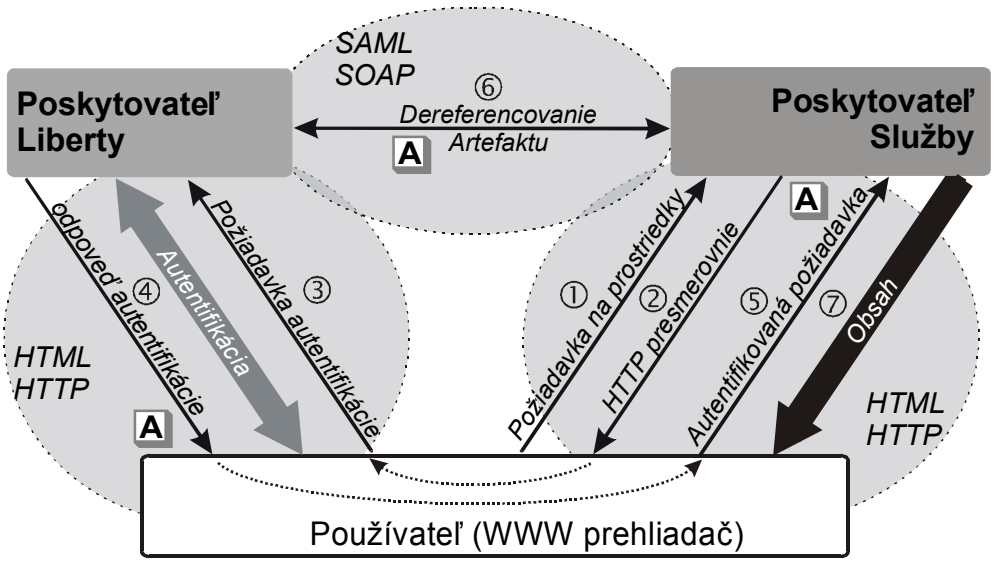
Security Assertion Markup Language (SAML)[14][15] je špecifikáciou syntaxe a sémantiky bezpečnostných tvrdení (Security Assertion) založený na protokole XML[16]. Jazyk SAML definuje formáty požiadaviek a odpovedí. Špecifikáciu SAML publikovala Organization for the Advancement of Structured Information Standards (OASIS). V dnešnej dobe je SAML v stave štandardizácie a pripomienkovania.

Bezpečnostné tvrdenie SAML (SAML assertion) môže dokazovať špecifickú charakteristiku entity, autentifikačný stav, autorizačné možnosti alebo určitú vlastnosť entity. Tvrdenia (assertion) vydáva a prideľuje autentifikačná autorita. Autentifikačné tvrdenie umožňuje používateľovi využívať inú službu alebo prostriedok bez nutnosti opätovnej autentifikácie. Používateľ sa autentifikuje zdroju a autentifikačná autorita zdroja mu poskytne tvrdenie (assertion). Pri prístupe na cieľ používateľ zasiela okrem požiadavky aj pridelené potvrdenie. Cieľ preskúma potvrdenie a ak je v súlade s bezpečnostnou politikou cieľa a cieľ dôveruje zdroju, že autentifikoval používateľa, potom poskytne prostriedky v súlade s autorizáciou používateľa bez nutnej opätovnej autentifikácie.

3.10.4 Projekt Liberty Alliance

Projekt Liberty Alliance [17][18] je skupina komerčných ale aj nekomerčných organizácii s úlohou vytvoriť štandard pre sieťové systémy digitálnej identity. V požiadavkách je decentralizovanosť a otvorenosť celej architektúry. Ako cieľ je vytvorenie spoločnej identity (Federated Identity). Projekt vznikol v septembri roku 2001 ako reakcia na projekt Microsoft Passport a požiadavky trhu.

Liberty Alliance projekt je architektonicky postavený na báze Security Assertion Markup Language (SAML)[14]. Na prenos autentifikačných a iných správ sa využíva práve SAML. Architektúra Liberty je zameraná prevažne pre prostredie s využitím bežných nemodifikovaných WWW prehliadačov, ale špecifikuje aj požiadavky na vnútornú podporu Liberty (Liberty-enabled) v prehliadačoch alebo iných aplikáciách na strane používateľa. Priebeh autentifikácie Liberty s použitím artefaktu je na Obr. 7.



Obr. 7: Priebeh autentifikácie Liberty s využitím artefaktu [13] .

Postup autentifikácie:

1. Používateľ žiada o prístup k obsahu u poskytovateľa. Manažér prostriedkov u poskytovateľa zisťuje, či používateľ priložil platný SAML artefakt k požiadavke.
2. Ak platný artefakt nie je priložený, vykoná sa HTTP presmerovanie [12] na poskytovateľa služby Liberty.
3. Poskytovateľ Liberty prijme presmerovanú požiadavku o autentifikáciu na poskytovateľa. Ak používateľ ešte nebol autentifikovaný, vykoná autentifikáciu.
4. Poskytovateľ Liberty zašle autentifikačnú odpoveď pre cieľ vo forme artefaktu.
5. Prehliadač používateľa presmeruje odpoveď aj s artefaktom na cieľ ako opätovnú požiadavku na prístup k obsahu.
6. Poskytovateľ služby dereferencuje artefakt a overí si jeho platnosť pomocou SAML s použitím SOAP protokolu [19] u poskytovateľa Liberty, prípadne si vymení ešte niektoré informácie o používateľovi.
7. Ak je všetko v poriadku sprístupní sa obsah používateľovi.

Ak je prehliadač Liberty-enabled potom je aktívnou súčasťou a môže komunikovať priamo Liberty protokolom s poskytovateľom Liberty. Táto komunikácia potom prebieha na báze protokolu HTTP [12], použitým na priamu výmenu SAML potvrdení pomocou protokolu SOAP [19].

Špecifikácia Liberty doporučuje používanie TLS/SSL pre zabezpečenie komunikačného kanálu a digitálne podpisovanie SAML správ.

4 Špecifikácia

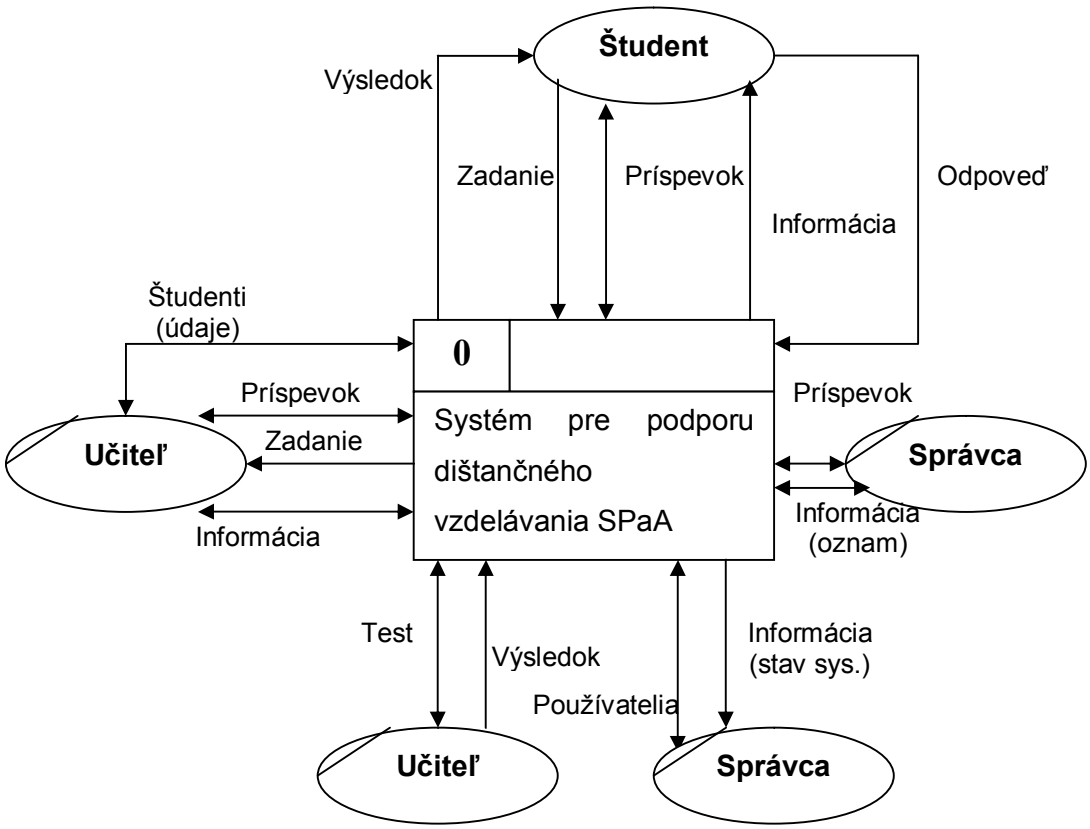
V nasledujúcich podkapitolách sa nachádza špecifikácia systému ako celok z pohľadu celkového opisu a informácií, ktoré do neho vstupujú a vystupujú. Potom nasleduje špecifikácia každého modulu zvlášť.

4.1 Celkový opis

Cieľom riešenia je vytvoriť architektúru samostatných modulov, aby bola zachovaná požiadavka na znovupoužiteľnosť komponentov a otvorenosť architektúry, aby nedochádzalo k zbytočnému plýtvaniu ľudských zdrojov na opakovaný vývoj nedokonalých a nekordinovaných riešení problematiky e-vzdelávania na FIIT.

Systém bude riešený ako množina „web services“, ktoré poskytujú požadovanú funkcionalitu a údaje. Tieto informácie a funkcie bude vyvolávať manažmentový modul, ktorý ich bude poskytovať používateľom prostredníctvom „web“ prehliadača.

Meta-informácie o predmete (prednášajúci, cvičiaci, názov koreňového dokumentu v báze konceptov, základné informácie o predmete, zápočte a pod.) sa budú zadávať prostredníctvom XML dokumentu, ktorý vyučujúci importuje do systému. Táto informácia sa bude uchovávať v module manažmentu predmetu.



Obr. 8: Kontextový digram systému.

4.2 Opis informácií

Informačné toky, ktoré vstupujú a vystupujú zo systému sú zachytené v kontextovom diagrame systému na Obr. 8. Identifikované sú základné roly používateľov, ktorí vystupujú ako externé entity.

4.2.1 Učiteľ

Učiteľ môže mať rolu prednášajúci, alebo cvičiaci. Obe roly však majú rovnaké informačné toky.

- Študenti(údaje): Môže študentov rozdeľovať do krúžkov, prípadne modifikovať niektorý údaj. Nie je mu dovolené študentov vytvárať ani mazať zo systému.
- Príspevok: Môže zaslať do fóra príspevok a ich aj prezerať. Tak isto môže „četovať“.
- Zadanie: Zadanie, ktoré študent odovzdal, vrátane výsledkov testov na funkcionalitu a podobnosť.
- Informácia (uč. materiál): Materiál, ktorí si študenti majú naštudovať. Môžu ho tvoriť aj oznamy týkajúce sa daného predmetu.
- Test: Zadávanie testov a výsledky uskutočnených testov.
- Výsledok: Výsledok hodnotenia zadania, testov alebo finálnej známky z predmetu.

4.2.2 Študent

- Výsledok: Výsledok hodnotenia zadania alebo testu, vrátane učiteľových komentárov. Súčasťou výsledku je aj štatistika (priemerné hodnotenie a pod.).
- Zadanie: Zadanie, ktoré sa musí odovzdať.
- Príspevok: Môže zaslať do fóra príspevok a ich aj prezerať. Tak isto môže „četovať“.
- Informácia (uč. materiál): Materiál, ktorý si má naštudovať, prípadne oznamy súvisiace s predmetom.
- Odpoveď: Odpovede, ktoré zadáva študent počas testovania.

4.2.3 Správca

- Študenti (údaje): Bežná administrácia používateľov systému a teda údaje, ktoré s tým súvisia.
- Informácia (stav. sys): Prehľad, ako systém funguje, štatistiky o používateľoch a pod.
- Informácia (oznam): Oznamy pre používateľov.
- Príspevok: Môže zaslať do fóra príspevok a ich aj prezerať. Tak isto môže „četovať“.

4.3 Báza znalostí

Modul bázy znalostí je určený na všeobecné použitie, preto sme sa nesnažili presne vymedziť prípady jeho použitia. Skôr sme sa pokúsili o zistenie čo najväčšej množiny vlastností, ktorých kombináciou by sme boli schopní implementovať rôzne typy znalostných systémov. Návrh bázy znalostí by mal byť

natoľko všestranný, aby podporoval skoro ľubovoľné informácie, ktoré sa môžu vyskytnúť pri vyučovaní na univerzite.

Jednou z hlavných požiadaviek je platformová nezávislosť. Túto vlastnosť je možné zabezpečiť používaním štandardných nástrojov a protokolov na komunikáciu a ukladanie dát. Ako najvýhodnejšie je zabezpečovať komunikáciu prostredníctvom XML podľa definície konzorcia W3C a samotné informácie ukladať vo formátie DocBook (kapitola 3.8).

Používaním XML a DocBook-u je možné okrem platformovej nezávislosti zabezpečiť aj všeobecnosť bázy znalostí - pri definovaní týchto protokolov boli brané do úvahy jednoduché texty, ako aj rozsiahle texty rozdelené do kapitol so sprievodným multimediálnym obsahom. Pomocou rôznych nástrojov na prácu s formátmi DocBook a XML je potom možné všetky uložené informácie jednoduchým spôsobom skonvertovať do požadovanej podoby. Klasickým príkladom je použitie na internetovom vzdelávacom portáli vo formáte HTML, je však aj možnosť výstupu vo forme dokumentu PDF alebo samostatnej prezentácie spúšťanej z média CD-ROM.

Databáza znalostí musí okrem samotného obsahu popisovať aj spôsob použitia uložených informácií. Jedná sa o definovanie poradia kapitol (lekcií), prípadne ďalšie naviazanie na testy znalostí z práve prebratej látky. Tieto vlastnosti je možné dosiahnuť použitím modelu SCORM (bližší popis v kapitole 3.9.4).

4.3.1 Opis údajov

Medzi hlavné údajové typy patria texty a multimédiá.

4.3.1.1 Textové údaje

Textové údaje budú ukladané vo formáte DocBook, ktorý už sám o sebe ponúka možnosti členenia dokumentov. Keďže ide o formát, ktorý je možné prečítať aj v jednoduchom textovom editore, aj vyhľadávanie v takto uložených súborov je pomerne jednoduché. Presnejší popis ukladaných atribútov je možné nájsť v kapitole 3.1. Toto budú vstupné aj výstupné údaje.

4.3.1.2 Multimédiá

Multimediálne údaje tvoria dôležitú súčasť vyučovania. Môže ísť napríklad o obrázky a schémy popisujúce vysvetľovaný problém, alebo aj o videosekvencie, ktoré môžu obsahovať nahovorené časti prednášok. Všetky tieto multimediálne súbory budú ukladané v databáze samostatne. Pri generovaní WWW stránky z obsahu sa tieto súbory jednoduchým spôsobom pripoja podľa značiek v textovej časti. Toto budú vstupné aj výstupné údaje.

4.3.2 Opis funkcií

Modul bázy znalostí poskytuje tieto základné funkcie:

Vlož dokument	
Popis	Pomocou tejto funkcie je vytváraný obsah výučbového systému. Vkladať je možné všetky podporované typy údajov. Vkladať dokumenty môže len ten používateľ, ktorý týmto právom v danom predmete disponuje.
Vstup	• typ vkladného súboru • identifikátor predmetu • umiestnenie v rámci už existujúceho dokumentu
Výstup	• potvrdenie vloženia

Čítaj dokument	
Popis	Táto funkcia je opakom predchádzajúcej. Slúži na sprístupnenie informácií uložených v báze znalostí. Opäť je potrebné rozlišovať, ktorí používatelia majú oprávnenie prístupu k požadovaným informáciám.
Vstup	• identifikátor predmetu • umiestnenie dokumentu v hierarchii (napr. číslo kapitoly)
Výstup	• požadovaný dokument • typ súboru

Zmaž dokument	
Popis	Táto funkcia umožňuje mazanie textových informácií (ľubovoľnej časti, ak je to povolené), aj multimediálnych súborov.
Vstup	• identifikátor predmetu • špecifikácia mazaného dokumentu
Výstup	• potvrdenie zmazania

Nastav atribúty	
Popis	Táto funkcia umožňuje nastavovať privilégiá jednotlivých používateľov. Je možné definovať skupiny, ktoré majú právo upravovať alebo prehliadať obsah bázy znalostí predmetu.
Vstup	• identifikátor predmetu • zoznam povolených operácií (čítanie, zapisovanie) • identifikátor skupiny používateľov
Výstup	• potvrdenie nastavenia

4.4 Testovanie znalostí

Tento modul je učený na automatizované testovanie vedomostí. Slúži na tvorbu testov a ich prostredníctvom na overenie získaných vedomostí študentov. Skúšajúcemu umožňuje jednoduchšie preskúšanie vysvetlenej látky tak, ako by to robil bežným spôsobom. Automatizovanie spočíva v automatickom hodnotení odpovedí pomocou počítača a v celkovom zjednodušení práce skúšajúceho v procese testovania.

Modul má byť univerzálny z hľadiska typov otázok a druhov testov. Odpoveďou na otázku môže byť aj jednoduchý program, kde si študent môže overiť syntax alebo funkčnosť programu. Mal by obsahovať podporu tzv. adaptívnych testov, kde sa náročnosť otázok mení podľa správnosti zodpovedania predchádzajúcich otázok Odpovede na otázky môže hodnotiť aj skúšajúci alebo môže meniť hodnotenie určené počítačom. Výsledky testov sa archivujú. Vo fáze prípravy testov by mala byť možnosť náhodného vytvárania testov, priradenie skupiny študentov konkrétnemu testu, príp. generovanie náhodných testov pre každého študenta zvlášť.

4.4.1 Údaje v module

Keďže sa jedná o nezávislý modul, musí vnútorne uchovávať rôzne typy údajov, ktoré prostredníctvom svojho rozhrania sprístupňuje okoliu. Na základe požiadaviek klienta a analýzy existujúceho riešenia (ATK [3]) nasleduje popis vstupov, výstupov, a jednotlivých údajových entít evidovaných v module.

4.4.1.1 Vstupy

Do modulu vstupujú len údaje definujúce testy:

- *Test* – základné parametre charakterizujúce test.
- *Otázka* – údaje charakterizujúce otázku, ako znenie, odpovede, bodové ohodnotenie.
- *Odpoveď na otázku* – vstupuje do modulu počas testovania.

4.4.1.2 Výstupy

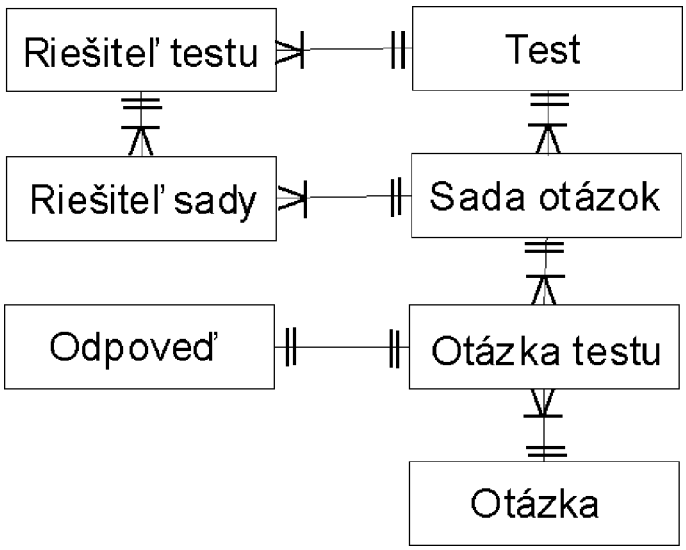
Výstupom z modulu sú jednotlivé otázky, a výsledky testov.

- *Otázka* – vystupuje počas testovania, obsahuje znenie otázky, odpovede.
- *Výsledok testu* – vystupuje po ukončení testovania, obsahuje informácie o výsledku testu pre študenta.
- *Štatistické výstupy* – rôzne štatistické prehľady podľa požiadaviek používateľa.

4.4.2 Základné údajové entity

V module sú využité nasledovné údajové entity (viď Obr. 9):

- *test* - je charakterizovaný určitými parametrami. Je to entita na najvyššej úrovni.
- *sada otázok* – je entita na strednej úrovni, charakterizuje skupinu otázok.
- *otázka* - je najelementárnejšia entita z pohľadu testu. Každá otázka je charakterizovaná parametrami.
- *otázka testu* – každá otázka prislúcha k jednej alebo aj viacerým sadám otázok, a má stanovené svoje číslo a variant.
- *odpoveď* - Odpoveď na otázku a jej hodnotenie.
- *riešiteľ testu* – charakterizuje výsledok testu pre daného študenta. Študent alebo inštruktor môže k odpovedi pridať poznámku. Touto formou sa zabezpečí komunikácia medzi inštruktorom a študentom pri hodnotení. Každému riešiteľovi je pridelené ID, unikátne pre každý vykonaný test a študenta.
- riešiteľ sady – charakterizuje výsledok pre sadu otázok.



Obr. 9: Logický model údajov modulu testovania znalostí.

4.4.2.1 Popis základných údajových entít

Každá entita je opísaná svojimi atribútmi a primárnym kľúčom, unikátnym pre každú fyzickú inštanciu entity. Nasleduje popis jednotlivých atribútov:

- *Test* združuje údaje charakterizujúce konkrétny test, a ovplyvňujúce jeho funkčnosť.
- *Sada otázok* reprezentuje objekt združujúci viacero otázok. Každá sada otázok prislúcha ku konkrétnemu testu.
- *Otázka* je definovaná svojimi parametrami, každá otázka môže prislúchať k jednej alebo viacerým sadám. Každej otázke je priradené číslo otázky a variant. Skupina otázok s rovnakým číslom obsahuje otázky s rôznym variantom. Pri generovaní testu je z každej skupiny s rovnakým číslom otázky vybraný len jeden variant.

- Každá *Otázka testu* prislúcha aspoň k jednej sade. Táto entita znázorňuje vzťah medzi sadou a otázkami.
- Každý *Riešiteľ testu* musí mať nejaký okruh riešiteľov (študentov). Táto entita vyjadruje vzťah medzi testom a študentom.
- *Riešiteľ sady* je podobný entite *Riešiteľ testu*, ale vzťahuje sa len na jednu sadu otázok z testu. Táto entita vyjadruje vzťah medzi sadou a študentom.
- Každá *Odpoveď* je spätá s *Testom*, *Sadou otázok*, *Otázkou*, a *Študentom*. Obsahuje samotnú odpoveď na otázku, a prípadné dodatočné informácie, ako čas odpovede, použitie pomôcky.

4.4.3 Opis funkcií

Modul pozostáva z troch častí [3] – definovanie testov, testovanie a prezeranie výsledkov. Prvá časť (proces 1) slúži na tvorbu testov. Inštruktor má možnosť definovať otázky a parametre testu.

V druhej časti (proces 2) je vykonávané testovanie. Študent dostane otázku (resp. otázky) a musí na ňu odpovedať. Inštruktor má tak isto možnosť vykonať test za účelom praktického overenia testu.

Tretia časť (proces 3) slúži na prezeranie výsledkov a úpravy hodnotenia. Študent má prístup k hodnoteniu vykonaných testov a k odpovediam na otázky. Na prípadnú reklamáciu hodnotenia môže študent k odpovedi pridať poznámku pre inštruktora. Inštruktor má prístup k všetkým výsledkom a hodnoteniam. Nevyhodnotené odpovede na otázky musí vyhodnotiť a môže zmeniť hodnotenie (na základe poznámky od študenta) alebo pridať poznámku k odpovedi. [3].

Jednotlivé funkcie budú sprístupňované prostredníctvom definovaného rozhrania modulu. Nasleduje popis jednotlivých funkcií, rozdelený podľa uvedených funkčných častí.

Všetky funkcie vracajú i chybový kód v prípade neúspechu, jedna sa však o štandardný výstup, ktorý nie je v popisoch funkcií explicitne definovaný.

4.4.3.1 Tvorba testu

Test je definovaný parametrami testu, ako sú typ, inicializačná sada otázok, časové obmedzenie, a pod. Rozhranie modulu pre prácu s testom definuje nasledovné funkcie:

Vytvor test	
Popis	Vytvorí nový test pre daný predmet, a definuje jeho parametre.
Vstup	• ID predmetu - identifikátor predmetu ktorému daný test prislúcha • Názov testu - názov testu • Typ testu - adaptívny / neadaptívny

Výstup	• ID testu - identifikátor novovytvoreného testu

Edituj test	
Popis	Zmení parametre testu.
Vstup	• ID testu - identifikátor testu, ktorého parametre sa menia Nepovinné: • Názov testu - názov testu • Typ testu - adaptívny / neadaptívny • ID východzej sady - identifikátor incializačnej sady otázok testu • Pravidlo zreťazenia - pravidlo náväznosti sád • Inštrukcie - inštrukcie pre riešiteľov testu • Max. čas - maximálny čas trvania testu • Postih - bodový postih za prekročenie času testu • Prístup - IP adresy počítačov s povoleným prístupom k testu
Výstup	—

Duplikuj test	
Popis	Duplikuje vybraný test, spolu s údajmi na nižšej úrovni, t.j. sady a otázky.
Vstup	• ID testu - identifikátor testu, ktorý sa má duplikovať
Výstup	• ID testu - identifikátor novovytvoreného testu

Zmaž test	
Popis	Zmaže vybraný test, spolu s údajmi na nižšej úrovni, tj. sady a otázky.
Vstup	• ID testu - identifikátor testu, ktorý sa ma zmazať
Výstup	—

4.4.3.2 Tvorba sád

Každý test obsahuje jednu alebo viacero nadväzujúcich sád. Sada je skupina otázok, ktorá sa samostatne vyhodnocuje, a na základe vyhodnotenia a pravidla zreťazenia definovaného v teste, sa pokračuje ďalšou sadou. Rozhranie definuje nasledovné funkcie pre prácu so sadami:

Vytvor sadu	
Popis	Vytvorí sadu otázok pre daný test.

Vstup	• ID testu - identifikátor testu, ktorému prislúcha daná sada • Typ sady - typ sady s hľadiska výberu otázok: Náhodný výber / náhodný variant / adaptívny
Výstup	• ID sady - identifikátor novovytvorenej sady otázok

Edituj sadu	
Popis	Zmení parametre sady.
Vstup	• ID sady - identifikátor sady, ktorej parametre sa menia Voliteľné: • Typ sady - typ sady z hľadiska výberu otázok • Max. čas - maximálny čas na zodpovedanie všetkých otázok sady • Postih - bodový postih za prekročenie max. času
Výstup	—

Duplikuj sadu	
Popis	Duplikuje sadu spolu s údajmi na nižšej úrovni, t.j. otázky.
Vstup	• ID sady - identifikátor sady, ktorá sa má duplikovať • ID testu - identifikátor testu, do ktorého sa má nová sada vložiť
Výstup	• ID sady - identifikátor novovytvorenej sady

Zmaž sadu	
Popis	Vymaže danú sadu spolu s údajmi na nižšej úrovni, t.j. otázky.
Vstup	• ID sady - identifikátor sady, ktorá sa má vymazať
Výstup	• ID sady - identifikátor novovytvorenej sady

4.5 Diskusné fóra a komentáre

Modul Fórum slúži na vzájomnú komunikáciu medzi používateľmi systému prostredníctvom textových správ. Jednotlivé príspevky od používateľov môžu byť rozdelené do viacerých tematických okruhov, prípadne sa môžu byť reakciami používateľov na informácie obsiahnuté v ostatných moduloch. Modul by mal byť vytvorený tak, aby bol závislý od informácií v iných moduloch len

pomocou jednoduchých parametrov. Týmto jednoduchým spôsobom by mohol byť použiteľný na spoluprácu s akýmkoľvek iným modulom v systéme.

4.5.1 Opis informácií

Štruktúra údajov v systéme je založená na zreťazení jednoduchých textových príspevkov do stromovej štruktúry. Týmto spôsobom je možné vytvárať aj obsiahlejšie diskusie tak, aby ostala zachovaná prehľadnosť.

Správa tvorí základný údajový typ. Jeho atribúty sú:

- ID správy - jedinečný identifikátor
- rodič - obsahuje identifikátor rodičovského príspevku, ak ide o odpoveď na iný príspevok
- téma - identifikátor z iného modulu, ku ktorému je diskusia vedená (napr. číslo lekcie)
- modul - identifikátor modulu, ku ktorému sa diskutovaná téma vzťahuje
- autor - obsahuje identifikátor autora príspevku
- čas - obsahuje dátum a čas zaslania príspevku
- viditeľnosť - atribút umožňujúci vytvoriť moderovanú diskusiu
- read-only - tento atribút umožňuje uzavrieť diskusiu k téme
- skupina - označuje skupinu používateľov, ktorí sa môžu k téme vyjadrovať
- názov - názov (nadpis) príspevku
- text - obsahuje samotný text príspevku

Štruktúra diskusného fóra je vytváraná stromovým reťazením správ pomocou atribútu rodič. Pri zobrazovaní je možné takto vytvorený strom zoraďovať podľa vytvorenej stromovej štruktúry, času, kľúčových slov a podobne.

4.5.2 Opis funkcií

Modul fórum poskytuje základné funkcie na správu diskusného fóra:

Vlož správu	
Popis	Táto funkcia umožňuje vložiť nový príspevok. Príspevok je odpoveďou na nejakú inú správu, ktorá sa v systéme už nachádza. Ak ide o novú správu do moderovanej diskusie, správa je označená ako neviditeľná, pokým ju moderátor nezviditeľní. Špeciálnym prípadom novej správy je správa, ktorá nie je odpoveďou na žiadnu predchádzajúcu

	správu. Vtedy ide o tému novej diskusie. Vkladať nové témy diskusie môže iba moderátor diskusnej skupiny.
Vstup	• Text • Typ správy • Voliteľné: ID správy na ktorú pripojiť
Výstup	• Stav operácie

Uzavri správu	
Popis	Táto funkcia zamedzí vkladanie nových správ do celého podstromu uzavretej správy. Takýmto spôsobom je možné uzavrieť diskusiu na danú tému.
Vstup	• ID správy
Výstup	• Stav operácie

Zmaž správu	
Popis	Pomocou tejto funkcie je možné zmazať správu a celý jej podstrom odpovedí. Je možné zmazať aj celú tému so všetkými príspevkami.
Vstup	• ID správy
Výstup	• Stav operácie

Zobraz správu	
Popis	Táto funkcia vracia text a atribúty požadovanej správy. Podľa parametrov je možné vrátiť presne jednu správu alebo správu spolu s celým jej podstromom odpovedí.
Vstup	• ID správy
Výstup	• Text

4.6 Odovzdávanie zadaní

Zber zadaní je dôležitou súčasťou systému pre podporu výučby. Cieľom tohto modulu je spravovať odovzdané zadania (ukladať, mazať, zaznamenávať manipuláciu so zadaniami), a tým uľahčiť celkový proces spracovávania zadaní. Návrh modulu je vytvorený univerzálne, umožňuje odovzdávať rôzne druhy zadaní bez závislosti od predmetu, ktorého sa zadania týkajú.

4.6.1 Opis informácií

Pre každé odovzdané zadanie sú evidované nasledovné údaje:

Zadania - vypracované zadanie odovzdané študentom (môže pozostávať z viacerých súborov). Zadanie musí obsahovať všetky potrebné súbory, ktoré sa viažu na dané zadanie.

Čas odovzdania - čas odovzdania zadania študentom. Ako referenčná hodnota sa berie systémový čas servera, na ktorom je spustený modul odovzdávania zadaní. Zároveň ide aj o verziu zadania.

Identifikátor študenta - jednoznačná identifikácia študenta, ktorý zadanie odovzdal

Identifikátor zadania - jednoznačná identifikácia odovzdaného zadania

Identifikátor predmetu - jednoznačná identifikácia predmetu, ktorého sa týka odovzdané zadanie

Vstupy:
- *zadanie:* je jeden, alebo viac súborov obsahujúcich vypracované zadanie

Výstupy:
- *zadanie:* je jeden, alebo viac súborov obsahujúcich vypracované zadanie.
- *zoznam verzií:*– zoznam verzií konkrétneho zadania konkrétneho študenta

Údajové entity:

- *zadanie:* súbor(y) s vypracovaným zadaním, má vzťah k študentovi, predmetu, a k číslu zadaniu

4.6.2 Opis funkcií

Ulož zadanie	
Popis	Funkcia uloží odovzdané súbory do databázy zadaní.
Vstup	• súbory so zadaním • identifikátor predmetu • identifikátor študenta • číslo zadania
Výstup	• číslo verzie zadania, príp. 0 pre chybovú správu

Vráť zoznam verzií	
Popis	Funkcia poskytne zoznam všetkých verzií odovzdaných zadaní pre daného študenta a číslo zadania, predmetu.
Vstup	• identifikátor predmetu • identifikátor študenta • číslo zadania

Výstup	• zoznam verzií

Vyber zadanie	
Popis	Funkcia poskytne súbory patriace ku konkrétnemu zadaniu konkrétneho študenta.
Vstup	• identifikátor predmetu • identifikátor študenta • číslo zadania • požadovaná verzia zadania
Výstup	• súbory patriace k danému zadaniu

Zmaž zadanie	
Popis	Funkcia zmaže odovzdané zadanie.
Vstup	• identifikátor predmetu • identifikátor študenta • číslo zadania • verzia zadania
Výstup	• potvrdenie zmazania

Doplň zadanie	
Popis	Funkcia pridá súbory k už odovzdanému zadaniu.
Vstup	• identifikátor predmetu • identifikátor študenta • číslo zadania • verzia zadania • pridávané súbory
Výstup	• potvrdenie doplnenia súborov k už odovzdaným súborom

4.6.3 Opis správania sa

4.6.3.1 Ulož zadanie

Súbory poskytnuté modulu sa uložia do databázy zadaní (adresárový strom, vytvorený v nasledujúcej štruktúre: /identifikátor_predmetu /identifikator_zadania/ identifikátor_studenta/ verzia), čas odovzdania zadania (resp. verzia), identifikátor študenta a zoznam odovzdaných súborov sa zapíšu do databázy odovzdaných zadaní.

4.6.3.2 Zoznam verzií

Poskytne zoznam odovzdaných verzií daného zadania pre vybraného študenta. Zoznam verzií sa prečíta z databázy odovzdaných zadaní.

4.6.3.3 Vyber zadanie

Poskytne súbory patriace k požadovanej verzii zadania odovzdaného konkrétnym študentom. Zoznam súborov, ktoré budú poskytnuté sa prečíta z databázy odovzdaných zadaní. Pri nešpecifikovaní verzie zadania je vrátená najaktuálnejšia verzia

4.6.3.4 Zmaž zadanie

Vymaže súbory patriace k určenej verzii zadania odovzdaného konktrétnym študentom, a zaeviduje kto a kedy dané zadanie zmazal. Pri nešpecifikovaní verzie zadania sa vymažú všetky.

4.6.3.5 Doplň zadanie

K už odovzdanému zadaniu pridá súbory poskytnuté modulu a aktualizuje databázu odovzdaných zadaní (aktualizuje odovzdané súbory a čas odovzdania). Pokiaľ nie je špecifikovaná verzia, dopĺňa sa k najaktuálnejšej verzii.

4.7 Modifikované zadania

Úlohou modulu na zisťovanie modifikovaných zadaní bude zistenie vzájomnej zhodnosti množiny programov s rovnakou funkciou napísaných v JSI. Ako algoritmus porovnávania bude použitý YAP verzia 3, upravený pre programy napísané v JSI a jazyku C. Pri zisťovaní modifikácií v zdrojových textoch sa modul bude zameriavať na:

- zmenu komentárov (pridanie/vymazanie/modifikácia),
- zmenu zobrazených textov,
- výmenu poradia procedúr,
- rozdelenie jedného zdrojového súboru do viacerých súborov a naopak,
- premenovávanie premenných, názvov procedúr a návestí.

Zisťované zmeny sú najčastejšími zmenami, ktoré vykonávajú študenti pri modifikácii programov. Takéto zmeny dokáže urobiť aj študent neznalý problematiky programovania. Naopak, ak bude študent schopný vykonať také zmeny v programe, aby nebol odhalený týmto modulom, bude potrebovať hlbšie znalosti a takéto zmeny sa vlastne rovnajú opätovnému naprogramovaniu väčšiny programu odznova, čo sa už nemusí považovať za modifikáciu.

4.7.1 Opis informácií

Vstupom modulu bude teda množina zdrojových textov programov a hodnota miery podobnosti, ktorej prekročenie indikuje vysokú pravdepodobnosť kopírovania (modifikovania) programu. Výstupom modulu bude označenie skupiny zdrojových textov, u ktorých miera podobnosti prekročí stanovenú hranicu a konkrétna percentuálna hodnota podobnosti týchto zadaní.

V systéme bude vytvorených viacero úložísk údajov potrebných pre modul zisťovania modifikovaných zadaní

- *zadania* - Jeden alebo viacero súborov so zdrojovým textom zadania, všetky musia byť identifikované príponou označujúcou v akom programovacom jazyku je zadanie naprogramované (v prípade JSI je to *.asm).
- *tokeny* - Pre každé odovzdané zadanie sa vytvorí jeden súbor (bez ohľadu na počet zdrojových súborov zadania), ktorý obsahuje tokeny vygenerované z príslušných zdrojových textov daného zadania.
- *výsledky porovnania* - Toto úložisko údajov obsahuje výsledky porovnania zdrojových súborov programov všetkých študentov pre konkrétne zadanie.
- *slovník* - Obsahuje priradenie tokenov jednotlivým príkazom daného programovacieho jazyka. Slovníkov môže byť vytvorených viacero, pričom každý musí mať označenie, pre ktorý programovací jazyk je vytvorený.

Do styku s týmto modulom budú prichádzať nasledovné externé entity:

- *pedagóg* - Bude zadávať texty úloh, ktoré sa budú odovzdávať ako zadania, bude môcť spustiť porovnanie podobnosti zadaní a zadať hranicu podobnosti, po prekročení ktorej sa budú zadania považovať za kopírované. Po vykonaní analýzy podobnosti zadaní si bude môcť prezrieť jej výsledky.
- *študent* - Do termínu stanoveného pedagógom bude môcť odovzdávať vypracované zadanie vo forme zdrojových textov programov.

4.7.2 Opis funkcií

Modul bude pozostávať z troch častí:

Prvá časť bude umožňovať zadávanie úloh pedagógom, ktoré budú riešené študentmi ako zadania a taktiež odovzdávanie vypracovaných zadaní študentmi.

Druhá časť bude vykonávať analýzu podobnosti zadaní, ktorá bude prebiehať v dvoch fázach:

1. Vytvorenie súboru s tokenmi - tento sa vytvorí vždy, keď študent odovzdá zadanie.

2. Vytvorenie súboru s výsledkami analýzy podobnosti zadaní - súbor sa vytvorí na požiadanie pedagóga, alebo automaticky po dosiahnutí termínu, do ktorého je možné odovzdávať konkrétne zadanie.

Tretia časť zobrazuje výsledky analýzy podobnosti zadaní. Tieto údaje sú prístupné len pedagógovi.

Vytvorenie zadania	
Popis	Funkcia vytvorí nové zadanie
Vstup	• identifikátor predmetu • číslo zadania • text zadania
Výstup	• potvrdenie vytvorenia zadania

Vytvorenie tokenov	
Popis	Funkcia vytvorí súbor s tokenmi k odovzdanému zadaniu
Vstup	• identifikátor predmetu • číslo zadania • súbory so zadaním
Výstup	• súbor s tokenmi

Analýza podobnosti zadaní	
Popis	Funkcia vykoná analýzu podobnosti nad všetkými zadaniami s rovnakým identifikátorom predmetu a číslom zadania
Vstup	• identifikátor predmetu • číslo zadania • súbory s tokenmi
Výstup	• súbor s výsledkami analýzy podobnosti zadaní

Zobrazenie výsledkov	
Popis	Funkcia zobrazí výsledky analýzy podobnosti zadaní
Vstup	• identifikátor predmetu • číslo zadania
Výstup	• výsledky analýzy podobnosti zadaní

4.7.3 Opis správania sa

4.7.3.1 Zadávanie a odovzdávanie zadaní

Každé zadanie bude identifikované svojím textom zadania, dátumom do ktorého je ho možné odovzdať a jedným alebo viacerými zdrojovými textami odovzdanými ako vypracovanie zadania študentom.

- *vytvorenie textu zadania* - Túto činnosť môže vykonať len pedagóg (súčasťou textu zadania je i termín, do ktorého je možné zadanie odovzdať).
- *odovzdanie zadania* - Túto činnosť môže vykonať len študent.

4.7.3.2 Analýza podobnosti zadaní

Vykoná sa pre každé riadne odovzdané zadanie každého študenta, ktorý zapísaný pre daný predmet. Vykonanie analýzy podobnosti zadaní je buď automatické, alebo spustené pedagógom ako je spomenuté v časti 3.4.

4.7.3.3 Zobrazenie výsledkov porovnania

Výsledky vykonanej analýzy podobnosti zadaní sa uložia do súboru spolu s časom vykonania a číslami študentov, ktorých zadania boli porovnávané. Študenti, ktorých zadania prekročili hranicu podobnosti stanovenú pedagógom budú vypísaní na zvláštnom zozname.

4.8 Kontrola funkčnosti zadaní

Cieľom tohto modulu je otestovať funkčnosť odovzdaných zadaní. Ma poskytnúť celkový pohlaď na funkčnosť zadaní ako aj na správnosť výsledkov. Modul umožní zjednodušenie vyhodnocovania zadaní. Pri testovaní treba testovať aj funkčnosť aj správnosť produktu, t.j. otestovať či dané program je funkčný (dá sa preložiť, skompilovať, spustiť a dáva požadované výstupy) a na koľko je správne (zhodnosť výstupu s očakávanými výstupmi pre zadané vstupy).

4.8.1 Opis informácií

Modul bude pracovať s týmito údajmi:

- *Zadanie* - zdrojový kód testovaného programu (zadania). Nemusí pozostávať iba z jedného súboru. Súbory musia byť jednoznačne identifikovateľné ku ktorému zadaniu patria. Obsahuje aj všetky doplnkové súbory, knižnice a iné súbory potrebné pri preklade a spustení programu. Pri použití neštandardných knižníc programovacieho jazyka je potrebné ich dodať spolu zo zadaním, aby mohlo byť dané zadanie preložené a otestované. Sem spadajú aj pomocné

súbory s ktorými pracuje program ako konfiguračné a dátové súbory programu (nie súbory z kódom programu). Patria sem aj konfiguračné súbory (Makefile), ktoré slúžia na nastavenie prekladača, ak by sme nevedeli ako zadanie prekladať nemohli by sme ho preložiť a tím by bolo zadanie nefunkčné.

- *testovacie vzorky* - množina testovacích vstupov a k ním priradených výstupov, pre ktoré má byť dané zadanie otestované. Tieto informácie sú použité pri samotnom testovaní preloženého programu a slúžia aj na vyhodnotenie vrátených výsledkov programu. Nie je možné robiť testovanie bez týchto informácií ak chceme aby bol modul univerzálny a schopný testovať rôzne typy programov.
- *kód programu* - úložiško pre pre uloženie programu a súbory generované počas prekladu programu, ktorý sa ďalej používa v procesu testovania. Toto sú dočasné informácie a po skončení testu zanikajú.
- *výsledky testovania* - obsahuje výsledky a výpisy z jednotlivých etáp testovania. Tieto informácie sú na konec spracované a na ich základe je zostavený celkový výsledok testovania, ktorý je výstupom tohto modulu.

Vstupom bude zadanie na testovanie a k nemu príslušné testovacie vzorky a výstupom výsledky otestovania zadania, ktoré budu pozostávať z výsledkov prekladu a výsledkov funkcionálneho testovania.

S modulom môže komunikovať každý používateľ, ale túto akciu bude vyvolávať najmä učiteľ.

4.8.2 Opis funkcií

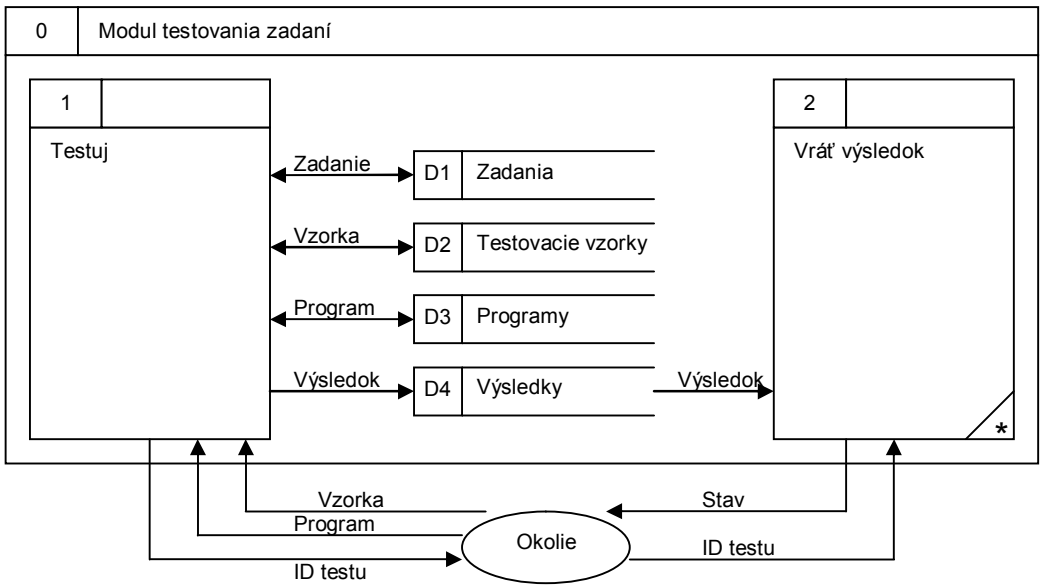
Modul bude pozostávať z dvoch funkcií. Prvá je samotné testovanie a druhá informuje o priebehu a výsledkoch testovania.

Testuj	
Popis	Funkcia spustí proces testovania programu
Vstup	• Zadanie • Testovacie vzorky
Výstup	• ID testu

Vráť výsledok	
Popis	Funkcia vráti stav alebo výsledok testovania.

Vstup	• ID testu
Výstup	• Stav, výsledok testovania

Diagram toku údajov je znázornený na Obr. 10. Okolie reprezentuje používateľov, môžu to by aj ostatné moduly, keďže modul sa chová ako služba, ktorá je vyvolávaná podľa potreby a teda samotný modul nerozlišuje medzi používateľmi. Rozlíšenie medzi používateľmi robí manažmentový modul.



Obr. 10: DFD diagram modulu testovania.

4.8.3 Opis správania sa

4.8.3.1 Testuj

Táto funkcia spustí proces testovania programu, čo je úlohou tohto modulu. Toto testovanie pozostáva z nasledujúcich krokov:

1. nastavenie,
2. kompilácia,
3. linkovanie,
4. funkcionálne testovanie,
5. vyhodnotenie výsledkov.

Tieto časti na seba nadväzujú a ak nejaká neprebehne správne, ostatné sa nevykonajú až na "vyhodnotenie výsledkov" ktoré sa vždy vykoná na záver tejto funkcie a sumarizuje výsledky jednotlivých časti.

Fáza nastavenia : Najskôr analyzuje vstup a pripraví (rozbalí) vstupné súbory pre ďalšie použitie a tiež pripraví testovací systém.

Fáza kompilácie: Pokúsi sa skompilovať zdrojový kód programu podľa konfigurácie. Výsledkom je skompilovaný program, ktorý je vstupom do ďalšej fáze spracovania, alebo chyba, pričom sa už v ďalšej fáze nepokračuje.

Fáza linkovania: Táto fáza je podobná predchádzajúcej len s tým rozdielom že je to posledná fáza prekladu tj. zostavenie strojového kódu programu. Výsledkom tejto fáze je buď preložený program alebo chyba.

Funkcionálne testovanie: Vstupom tejto fázy je preložený program a testovacie vzorky. Program sa spustí a predávajú sa mu jednotlivé vstupné údaje, pričom sa zaznamenáva odozva programu na tieto vstupy. Po ukončení programu sú porovnané odozvy programu z jednotlivými testovacími vzorkami. Program sa môže ukončiť z vlastnej činnosti, alebo násilne po chybe programu, vyčerpaním testovacích vstupov, dlhej nečinnosti programu a ďalších príčinách. Výstupom tejto fázy je výsledok porovnania testovacích vzoriek s odozvou programu a spôsobu ukončenia programu.

Vyhodnotenie výsledkov: Táto fáza sa vykoná vždy aj po chybovom ukončení niektorej z predchádzajúcich fáz. Spracováva výsledky jednotlivých fáz a na ich základe zostaví výstup funkcie, ktorý pozostáva z výsledkov prekladu (fázy kompilácia a linkovania) a z výsledkov testovania tj. zhodnosť testovacích vzoriek a výstupu programu a správanie programu pri a po testovaní (chyby, spôsob ukončenia, návratové hodnoty programu).

4.8.3.2 Vráť výsledok

Táto funkcia vracia výsledok testovania ak bolo toto testovanie už ukončené, inak stav prebiehajúceho testovania. Vstupom funkcie je identifikátor testu, na základe neho sa určí testovanie, pre ktoré požadujeme výsledky (stav). Stav testovania sa dá ľahko zistiť z výstupov jednotlivých fáz procesu testovania, ktorý spúšťa funkcia Testuj.

4.9 Správa používateľov

Tento modul by mal poskytovať autentifikačné a autorizačné služby pre ostatné moduly vrámci ich vlastného interného manažmentu. Okrem týchto funkcií pre náš projekt by mal mať univerzálne rozhranie, ktoré by mu umožňovalo poskytovať tieto funkcie aj mimo projekt. Veľkou výhodou je možnosť jednotného prihlasovania sa na fakultné služby, poskytovaná týmto modulom.

Modul by mal pod správou základné autentifikačné, autorizačné a identifikačné údaje o používateľoch. Jednotlivým systémom by mal ale poskytovať len relevantné údaje alebo len symbolické identifikátory

používateľa, ktoré neumožnia zistiť priamo fyzickú identitu používateľa a tak zvýšiť objektivitu hodnotenie pedagógom.

Modul by mal poskytovať možnosť autentifikácie pre jednotlivých používateľov a zároveň by pôsobil ako autorizačný modul pre ostatné moduly. Všetky tieto údaje by mal čerpať zo spoločného úložiska autentifikačných údajov na fakulte (LDAP). Prípadne by priamo využíval možnosti autentifikačného servra fakulty.

Ohľadom na analýzu je požiadavkou aby tento modul ako komunikačné rozhranie používal rozhranie na báze SOAP. Ako správy zasielané pomocou SOAP by mali byť správy SAML.

Požadované SOAP funkcie:

SAMLRequest - zaslanie poziadavky

Vstupy:

SAMLRequestMessage - SAML správa

Výstupy

SAMLResponseMessage - SAML odpoved

SAML správa prenáša všetky relevantné informácie ohľadom používateľa viď analýza. Výhodou je vlastnosť jednotného prihlasovania používateľov k systému, prípadne v súčinnosti s ostatnými systémami.

4.10 Manažment predmetu a prezentačná časť

Manažment predmetu tvorí integračný prvok jedného alebo viacerých modulov. Zabezpečuje riadiacu logiku predmetu, udržiava informácie o jednotlivých krokoch používateľov a transformuje ich požiadavky zo vstupu na požiadavky na jednotlivé moduly. Manažment tvorí portál pre používateľov.

Pri využívaní iba niektorých málo modulov môže manažmentový modul zabezpečovať transformáciu všeobecných dát vo forme XML z jednotlivých modulov, do výstupného formátu pre používateľa.

4.10.1 Opis informácií

Modul prijíma požiadavky od používateľov prostredníctvom HTTP rozhrania. Modul ich identifikuje a prostredníctvom SOAP protokolu nechá vykonať požadovanú funkcionalitu na jednom z predchádzajúcich modulov.

4.10.1.1 Údaje na strane používateľov

Údaje sa z modulu manažmentu prenášajú v HTML forme, v ktorej sú už upravené pre prezentačné účely. Údaje od používateľov sa do modulu dostávajú prostredníctvom transakcie HTTP Request ako POST alebo GET metóda (podľa potreby) .

Údaje sú rovnaké ako na kontextovom diagrame systému na Obr. 8, keďže tento modul transformuje požiadavky od používateľov pre jednotlivé moduly, ktoré vystupujú ako „Web-services“.

4.10.1.2 Údaje na strane modulov

Je to vlastne zjednotenie vstupno-výstupných údajov od všetkých modulov opísaných vyššie.

4.10.2 Opis funkcií

Na strane modulov sú funkcie orientovanie na vyvolanie funkcií jednotlivých modulov, ktoré boli popísané vyššie v špecifikácii.

Na strane používateľov sa vyvolanie nasledujúcich funkcií zrealizuje prostredníctvom HTTP protokolu a príslušného skriptu, ktorý HTTP server následne vykoná.

Zobraz informáciu	
Popis	Zobrazí používateľovi informáciu uloženú v báze znalostí (konceptov).
Vstup	• ID informácie • ID používateľa
Výstup	• informácia vo formáte HTML

Zadaj informáciu	
Popis	Uloží používateľovu informáciu do bázy znalostí (konceptov).
Vstup	• ID informácie • ID používateľa • informácia v DocBook dokumente
Výstup	• výsledok operácie

Zobraz diskusiu	
Popis	Zobrazí používateľovi požadovanú diskusiu, alebo jej časť.

Vstup	• ID diskusie • ID používateľa • Rozsah príspevkov.
Výstup	• Diskusia, resp. jej časť ako HTML dokument.

Zadaj príspevok	
Popis	Pridá príspevok do diskusie.
Vstup	• ID diskusie • ID používateľa • Príspevok • ID kde pripojiť príspevok
Výstup	• Výsledok operácie

Vytvor zadanie	
Popis	Vytvorí zadanie.
Vstup	• Text zadania • „deadline“ • ID predmetu
Výstup	• Stav operácie

Odovzdaj zadanie	
Popis	Umožní študentovi odovzdať zadanie.
E	• ID predmetu • ID používateľa • ID Zadania • Súbor (-y)
Výstup	• Výsledok operácie

Zobraz stav odovzdávania

Popis	Poskytne zoznam mien a ich odovzdaných zadaní spolu s dátumami a aj stavom testovania, ak sú jeho výsledky dostupné.
Vstup	• ID predmetu • ID používala • ID Zadania Voliteľné • ID používateľa (študenta)
Výstup	• Zoznam vo forme HTML

Poskytni zadanie	
Popis	Poskytne používateľovi odovzdané zadanie.
Vstup	• ID predmetu • ID používateľa • ID Zadania Voliteľné • ID používateľa (študenta)
Výstup	• Súbory

Testuj zadanie	
Popis	Príkaz pre systém, aby vykonal testovanie vybraných zadaní.
Vstup	• ID predmetu • ID používateľa • ID Zadania • Zoznam zadaní, ktoré sa majú otestovať.
Výstup	• Stav operácie

Zisti plagiátorstvo	
Popis	Príkaz pre systém, aby vykonal zisťovanie plagiátorstva vybraných zadaní.
Vstup	• ID predmetu • ID používateľa • ID Zadania • Zoznam zadaní, ktoré sa majú otestovať.
Výstup	• Stav operácie

Zobraz výsledok hodnotenia zadania	
Popis	Zobrazí výsledok hodnotenia zadania pre daného používateľa.
Vstup	• ID predmetu • ID používateľa • ID Zadania
Výstup	• Výsledok hodnotenia vo forme HTML

Edituj výsledok hodnotenia zadania	
Popis	Dovolí používateľovi upraviť výsledok hodnotenia zadania pre daného používateľa.
Vstup	• ID predmetu • ID používateľa • ID Zadania • Zmené údaje
Výstup	• Stav operácie

Zobraz stav systému	
Popis	Zobrazí používateľovi stav systému. Zobrazená informácia záleží od práv používateľa.
Vstup	• ID používateľa
Výstup	• Stav vo forme HTML

Pridaj používateľa	
Popis	Pridá do databázy používateľov nového používateľa.
Vstup	• Meno • Osobné číslo • Heslo • Rola
Výstup	• Stav operácie

Zruš používateľa	
Popis	Vymaže používateľa z databázy používateľov systému.
Vstup	• ID používateľa
Výstup	• Stav operácie

Edituj používateľa	
Popis	Zmení informácie o používateľovi.
Vstup	• ID používateľa • Zmenené informácie
Výstup	• Stav operácie

Zobraz používateľov	
Popis	Zobrazí používateľov podľa zadaného kritéria (prihlásených na predmet, všetkých, iba určitých, ...)
Vstup	• Kritérium • ID používateľa
Výstup	• Zoznam používateľov

Importuj meta-informácie	
Popis	Importuje od prednášajúceho metainformácie o predmete vo forme XML dokumentu.
Vstup	• XML • ID používateľa
Výstup	• Stav operácie

Zobraz meta-informácie	
Popis	Zobrazí meta-informácie o predmete
Vstup	• ID používateľa
Výstup	• Meta-informácie v HTML

Zobraz predmety

Popis	Zobrazí predmet(-y) podľa zadaného kritéria.
Vstup	• Kritérium
Výstup	• Zoznam predmetov v HTML forme.

Vytvor test	
Popis	Vytvorí pre daní predmet test.
Vstup	• ID predmetu • Test • Zoznam študentov
Výstup	• ID testu

Zobraz test	
Popis	Zobrazí pre daní predmet test.
Vstup	• ID predmetu • ID testu
Výstup	• Test vo forme HTML

Odovzdaj test	
Popis	Odovzdanie odpovedí pre daný test.
Vstup	• ID predmetu • ID testu • ID používateľa • Odpovede
Výstup	• Stav operácie

Vyhodnoť test	
Popis	Vyhodnotí testy od vybraných (aj všetkých) používateľov.
Vstup	• ID predmetu • ID testu • Zoznam používateľov

Výstup	• Stav operácie

Zobraz výsledok testu	
Popis	Zobrazí výsledok hodnotenia testu pre daných používateľov.
Vstup	• ID predmetu • ID testu • Zoznam používateľov
Výstup	• Výsledok vo forme HTML.

Edituj výsledok testu	
Popis	Dovolí učiteľovi upraviť výsledok testu, resp. pridať komentár.
Vstup	• ID predmetu • ID testu • ID používateľa • Zmené informácie
Výstup	• Stav operácie

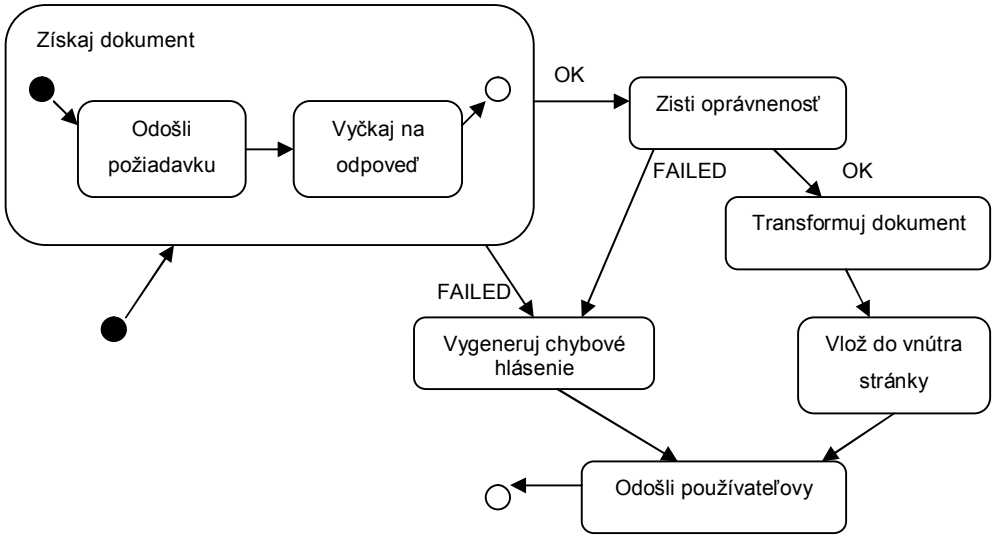
Zmeň tému prezentácie	
Popis	Zmení tému a štýl prezentovaných informácií (farby, písmo a atď.) a dovolí vybrať inú z ponúknutého zoznamu.
Vstup	• ID témy • ID používateľ
Výstup	• Stav operácie

4.10.3 Opis správania sa

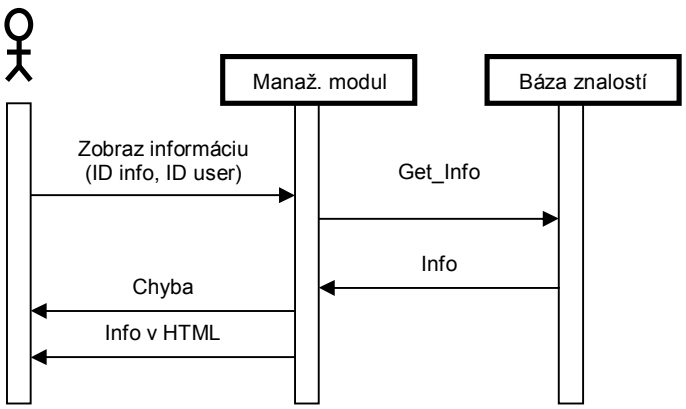
4.10.3.1 Zobraz informáciu

Zobrazí používateľovi požadovanú informáciu z bázy znalostí (konceptov). Môže to byť informácia o predmete, oznamye, ale aj učebný materiál. Informácia prichádza do tohoto modulu vo forme DocBook, ktorá sa prostredníctvom XSLT pretransformuje do HTML formy. Pri používaní CSS štýlov sa potom používateľovi zobrazí dokument v danej téme.

Stavový diagram tejto činnosti je na Obr. 11.



Obr. 11: Stavový diagram funkcie Zobraz informáciu.



Obr. 12: Sekvenčný diagram funkcie Zobraz informáciu.

Sekvenčný diagram, ktorý ozrejmí vzťahy medzi , modulom manažmentu predmetu a bázy ználosí znalostí je na Obr. 12. Ulož informáciu

Najprv sa uverí oprávnenosť požiadavky, teda prístupové práva a potom uloží do bázy znalostí (konceptov) informáciu vo forme DocBook.

4.10.3.2 Zobraz diskusiu

Požiada modul fóra aby poskytol danú diskusiu, ktorá je určená ID diskusie. Modul vykoná kontrolu, či má používateľ na prezeranie danej diskusie. Po získaní diskusie modul následne tento „surový“ formát prevedie do HTML dokumentu.

4.10.3.3 Zadaj príspevok

Po overení oprávnenosti pridá do danej diskusie príspevok od používateľa.

4.10.3.4 Vytvor zadanie

Pre daný predmet vytvorí zadanie. Text zadania sa zadá vo forme DocBook a uloží sa do bázy znalostí.

4.10.3.5 Odovzdaj zadanie

K danému zadaniu a používateľovi si overí hraničný termín a oprávnenosť požiadavky, teda či ma daný používateľ právo odovzdať vypracované zadanie k danej úlohe.

4.10.3.6 Zobraz stav odovzdávania

Oprávnenému používateľovi (prednášajúci, cvičiaci) zobrazí príslušnú tabuľku k danému zadaniu, kde bude pri každom šetudentovi informácia kedy odovzdal zadanie, výsledok automatického testovania a aj výsledok zisťovania plagiátorstva, pokiaľ sa vykonali. Pri zadaniach, ktoré sa ešte nestihli spracovať a boli zadané na spracovanie, sa zobrazí i táto informácia.

4.10.3.7 Poskytni zadanie

Funkcia slúži pedagógovi, ktorý si chce stiahnuť odovzdané zadanie. Overí sa oprávnenosť požiadavky a z databázy odovzdaných zadaní sa vyberie požadovaný súbor(-y).

4.10.3.8 Testuj zadanie

Táto funkcia pridá do radu čakajúcich zadaní na testovanie nové zadanie. Funkcia vráti informáciu o úspešnom, resp. neúspešnom zaradení do frontu. Testovanie prebieha automaticky a jeho výsledky sa potom zobrazia pri vyvolaní funkcie Zobraz stav odovzdávania.

4.10.3.9 Zisti plagiátorstvo

Presne ako funkcia testuj zadanie zaradí odovzdané zadanie (súbor) do radu zadaní, čakajúcich na takéto spracovanie.

4.10.3.10 Zobraz výsledok hodnotenia zadania

Táto funkcia je určená pre študentov, ktorým sa zobrazí výsledok hodnotenia. K danému zadaniu a príslušného študenta sa mu zobrazí jeho hodnotenie.

4.10.3.11 Edituj výsledok hodnotenia zadania

Dovolí pedagógovi zmeniť výsledok hodnotenia. Vstupom sú informácie, ktoré sa majú zmeniť. Modul potom tieto zmeny prenesie do zodpovedajúcej položky v databáze.

4.10.3.12 Zobraz stav systému

Poskytne komplexné informácie o stave systému. Najprv sa podľa práv prístupu používateľa zozbierajú relevantné informácie, ktoré sa mu potom zobrazia.

4.10.3.13 Pridaj používateľa

Táto funkcia prináleží len správcovi systému, preto sa najprv vykoná kontrola oprávnenosti požiadavky. Funkcia vytvorí v module správy používateľov systému novú položku, do ktorej naplní potrebné informácie.

4.10.3.14 Zruš používateľa

Táto funkcia prináleži len správcovy systému, preto sa najprv vykoná kontrola oprávnenosti požidavky. Funkcia zmaže používateľa z databázy používateľov systému.

4.10.3.15 Edituj používateľa

Táto funkcia prináleži správcovy systému. Heslo si však môže zmeniť aj samotný používateľ. Pred vykonanim zmien sa vykoná autentifikácia.

4.10.3.16 Zobraz používateľov

Poskytne zoznam používateľov podľa zadaného kritéria. Podľa roly používateľa sa mu zobrazia releventné informácie. Študentovy a pedagógovy sa môže zobraziť iba zoznam mien študentov bez informácií o ich prístupe a informácii, ktoré by boli v rozpore zo zákonom o ochrane osobných údajov.

4.10.3.17 Importuj meta-informácie

Do modulu sa nahrajú meta-informácie o predmete. Informácie sú v XML forme a v takejto forme sú aj uložené.

4.10.3.18 Zobraz meta-informacie

XML dokument s meta-informáciami sa pretransormuje do prezentovateľnej formy v HTML a pošle klientovi.

4.10.3.19 Zobraz predmety

Podľa zadaného kritéria zobrazí z databázy predmetov predmety.

4.10.3.20 Vytvor test

Pedagóg vytvorí v module testovania znalostí nový test, ktorý naplní danými otázkami a ich hodnotením a aj časom uskutočnenia testu Okrem toho zadá zoznam študentov, ktorý sa ho majú zúčastniť. Pre tým sa najprv pedagóg autorizuje.

4.10.3.21 Zobraz test

Podľa načasovania testu ho pri prihlásení zobrazí študentom. V tomto momente začína prebiehať test.

4.10.3.22 Odovzdaj test

Odovzdá vypracovaný test od študenta, priradí mu jedinečné čislo, ktoré sa odpamätá.

4.10.3.23 Vyhodnoť test

Vyhodnotí všetky odovzdané testy a výsledky uloží k jednotlivých odovzdaným odpovediam.

4.10.3.24 Zobraz výsledok testu

V prípade učiteĺa zobrazí celkové hodnotenie pre každého študenta vrátane celkovej štatistiky. V prípade študenta zobrazí iba jeho hodnotenie a štatistiku.

4.10.3.25 Edituj výsledok testu

Zmení hodnotenie testu. Toto je dovolené len učiteľovy.

4.10.3.26 Zmeň tému prezentácie

Dovolí zmeniť tému, akou sa používateľovy zobrazujú dokumenty. Zmena sa vykoná použitím ineho CSS štýlu. ID štýlu sa uloží v databáze používateľov k príslušnému používateľovy po autorizácii tejto operácie.

5 Návrh

V návrhu sme vychádzali z techniky návrhu zdola nahor. Primárne sme si stanovili určitú základnú architektúru celku a základné vzťahy medzi modulmi a potom sme sa snažili pre jednotlivé moduly navrhnúť čo najvhodnejšie riešenie s využitím existujúcich už vytvorených funkcií a aplikácií.

5.1 Celková architektúra

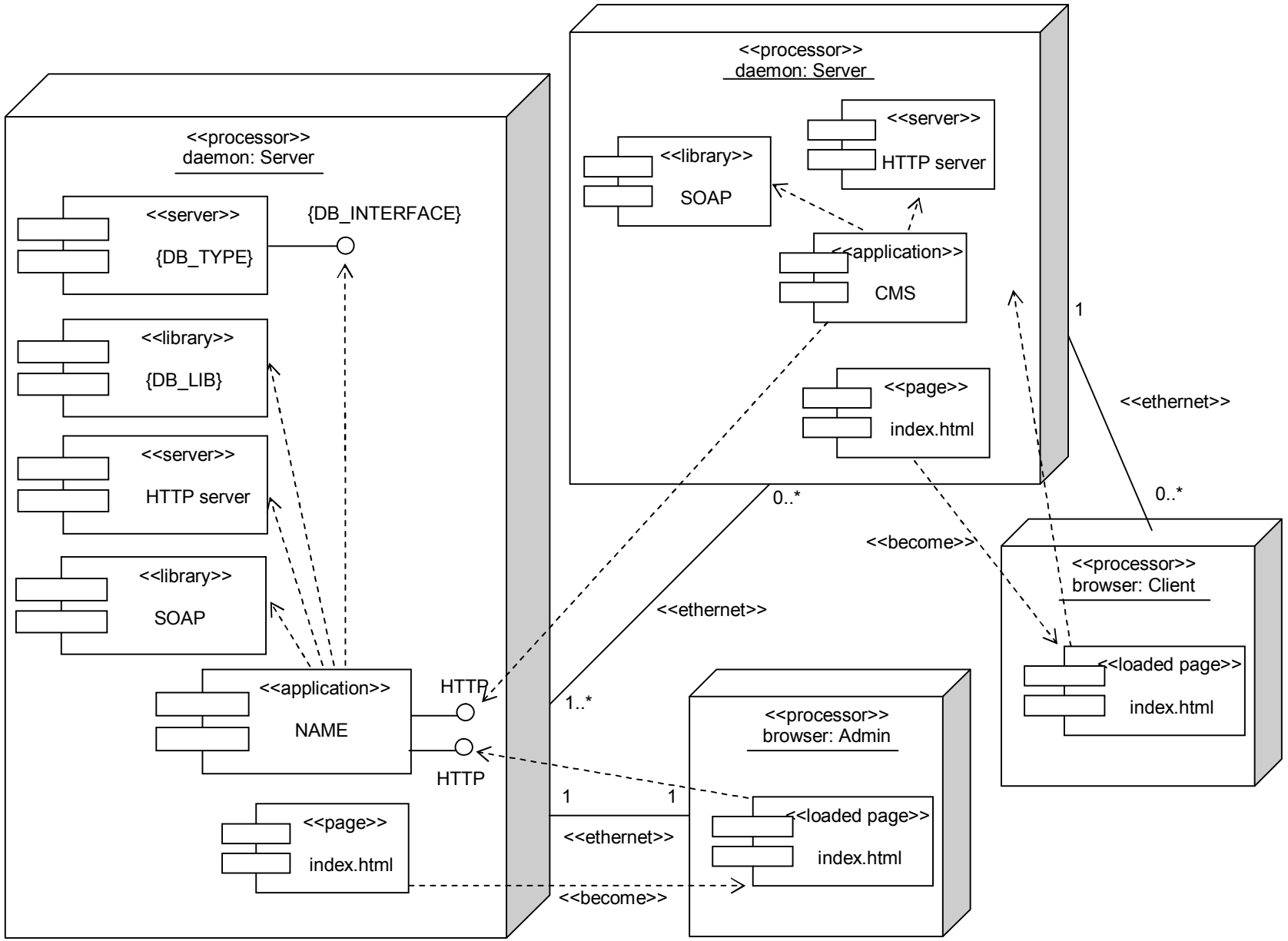
Celková architektúra pozostáva so samostatných modulov. Tieto moduly sú vlastne samostatnými aplikáciami poskytujúcimi svoju funkčnosť prostredníctvo rozhrania na báze „web“ služieb. Na komunikáciu medzi jednotlivými modulmi sa využíva protokol SOAP nad protokolom HTTP. Každý modul má vlastnú špecifikáciu rozhrania publikovanú pomocou WSDL opisu rozhrania.

Moduly sú rozdelené do štyroch skupín podľa ich činnosti:

- Úložiská – Charakteristickou vlastnosťou týchto modulov je, že uchovávajú a poskytujú údaje pre ostatné časti systému. Typickým reprezentantom je modul bázykonceptov. Každé úložisko má implementované aj základné operácie nad uloženými dátami. (vkladanie, čítanie, vyhľadávanie, mazanie), čo vlastne tvorí manažment údajov.
- Vstupno/výstupné – Tieto moduly sú samostatné alebo sú súčasťou manažmentového modulu predmetu. Zabezpečujú komunikáciu medzi e-vzdelávacou aplikáciou a používateľom. Unifikujú požiadavky z rôznych vstupných kanálov tak, aby vstupy boli transparentné pre manažmentový modul.
- Funkčné moduly – Implementujú špecifické operácie nad údajmi z iných modulov. Príkladom je testovanie podobnosti alebo funkčnosti zadaní.
- Manažment predmetu – Zabezpečuje riadenie výkonnej logiky toku dát medzi modulmi. Jeho úlohou je spracovávať požiadavky prijaté od vstupno/výstupných modulov a prideľovať ich vykonávanie príslušným modulom.

Ako výmenný formát medzi modulmi sa využívajú XML dokumenty. Konkrétny formát správ je spoločný pre všetky moduly (SOAP), líši sa v type prenášaného dokumentu (DocBook, MathML, QuizML, SCORM). Definícia typu dokumentu je deklarovaná pomocou DTD.

Na Obr. 13 je znázornená všeobecná architektúra systému prostredníctvom diagramu rozmiestenia (Deployment Diagram) kombinovaného s diagramom súčiastok (Component Diagram).



Obr. 13: Architektúra celého systému.

Uzol so stereotypom „processor“, typom prehliadač (browser) a inštanciou klient (Client) predstavuje používateľa systému a jeho počítač s internetovým prehliadačom. Tento uzol môže naraz komunikovať iba s jedným manažmentovým modulom (processor, daemon, server). Manažmentový modul môže v tom istom čase byť dostupný pre mnoho používateľov. Fyzicky sú prepojené technológiou „ethernet“. Logická komunikácia prebieha prostredníctvom HTTP protokolu. Používateľ sa do prehliadača načíta „web“ stránku z manažmentového modulu, prostredníctvom ktorej komunikuje s týmto modulom.

Manažmentový modul (Manažment predmetu) komunikuje aspoň s jedným modulom „processor, daemon, server“ prostredníctvom technológie „ethernet“ a protokolu HTTP.

Moduly „processor, daemon, server“ muožu byť buď úložiská, vstupno/výstupné alebo funkčné. Názvy v zložených zátvorkach sa nahradia podľa úlohy modulu nasledovne:

	DB_LIB	DB_INTERFACE	DB_TYPE
XML	libXML	XML	XML
Databáza	libSQL	OBDC	SQL
Adresárová služba	libLADP	LDAPv3	LDAP(X.500)

Modul nemusí byť dočasne využívaný žiadnym manažmetovým modulom. Konfigurácia modulu sa vykonáva prostredníctvom „web“ stránky a teda cez HTTP protokol autorizovanou osobou, ktorá je administrátor (processor, browser, Admin) cez vlastné rozhranie. Modul muože byť konfigurovaný naraz iba jedným administrátorom a jeden uzol administrátora môže konfigurovať naraz tiež iba jeden modul.

6 Prototyp

Uvádzame tu popis prototypu časti systému, ktoré sme sa rozhodli zprototypovať.

6.1 Cieľ prototypovania

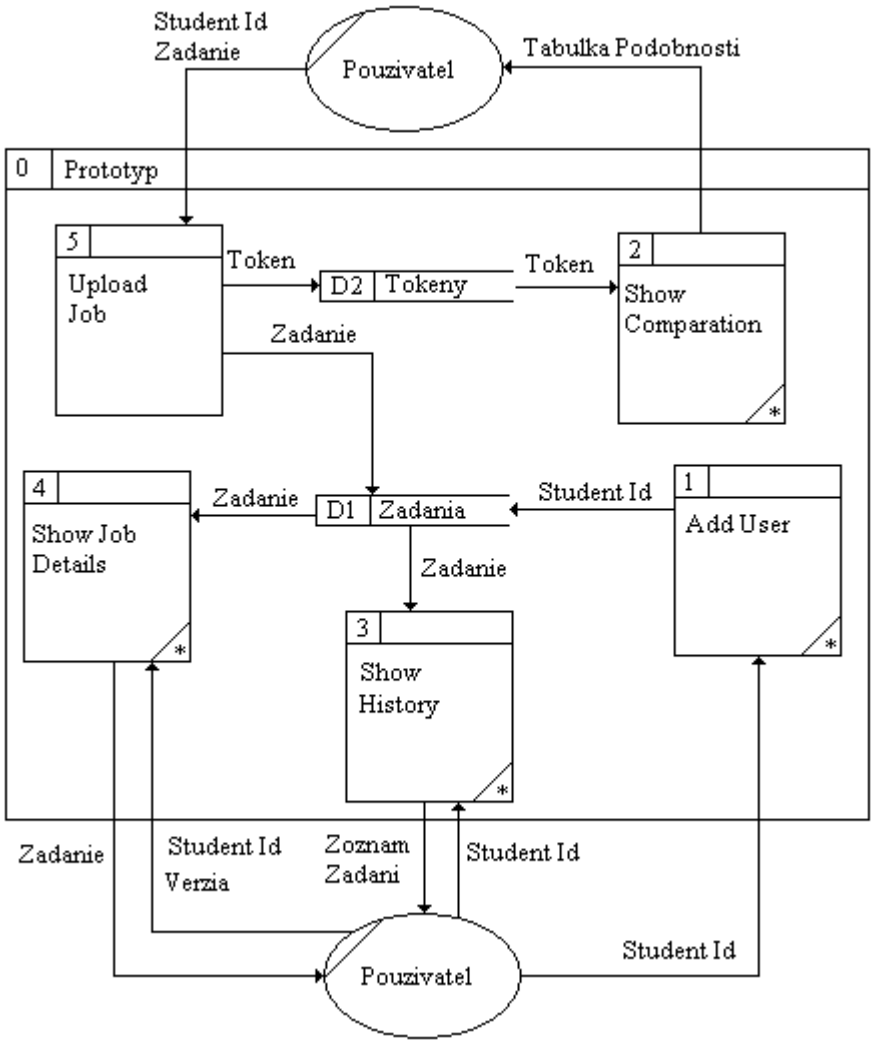
Vytváraný prototyp je primárne určený na tieto ciele:

- experimentálne overiť navrhnú architektúru,
- možnosť integrovania s existujúcim riešením,
- praktické skúšky technológie SOAP,
- študijný materiál pre členov tímu.

Prototyp má byť realizovaný ako samostatný program zatiaľ bez integrácie na nejaký bežne používaný WEB server. Prototyp si komunikáciu pomocou HTTP musí realizovať samostatne. Výsledkom prototypovania bude prototyp na zahodenie.

6.2 Návrh prototypu

Diagram toku údajov prototypu je na Obr. 14.



Obr. 14: DFD prototypu.

6.3 Zhodnotenie

Cieľ prototypovania splnil požiadavky na prototyp kladené. Vzhľadom na oboznamovanie sa jednotlivých členov s praktickou aplikáciou a použitím SOAP protokolu bola stanovená aj náročnosť prototypu.

7 Riešenie

V tejto kapitole opisujeme riešenie, ktoré sme navrhli a realizovali v letnom semestri. Najprv zdôvodníme zmeny špecifikácie, ktoré následne opíšeme. Za nimi nasleduje technická dokumentácia častí produktu, ktoré sme bytvorili, alebo modifkovali. V letnom semestri sme prehodnotili názvoslovie popisu systému, aby nedošlo ku zmetku, keďže niektoré použité existujúce súčasti (Moodle) používajú rovnaké pomenovania.

Náš produkt sa volá Dagwood, pozostáva z niekoľkých podsystémov (subsystémov), ktoré boli pôvodné pomenované ako moduly. Moduly sa odteraz budú rozumieť rozšírujúce prvky systému Moodle.

7.1 Ohraničenia, zmeny špecifikácie, priority riešenia

Oproti zámerom a návrhu v letnom semestri došlo k prehodnoteniu priórit a následne zmeny pôvodnej špecifkácie. V tejto podkapitole uvádzame najmarkantnejšie dôvody zmeny a potom uvedieme nové špevifikácie. Dôvody zmien (mimo poradia dôležitosti a váhy):

- Po zimnom semestri vypadol z tímu Bc. Peter Procházka, ktorý bol súčasne jeho vedúcim. Okrem toho zastával aj úlohu hlavného architekta. Jeho bohaté skusenosti s technológiami, ktoré sme chceli použiť sa dajú ťažko nahradiť.
- V pôvodnej špecifikácii a návrhu sme rátali s využiťím XML databázy. Vzhľadom na to, že budúci správca systému odmietol možnosť použiť inú ako Postgre databázu, museli sme upustiť aj od tohto zámeru.
- Nezanedbateľná je aj komplexnosť takého systému zrejmá z predchádajúcich kapitol. Keďže sa nám navyše tím zmenšil, museli sme vážnejšie zohladniť tento fakt.
- Dohoda s pedagógom mala tiež nemalý podiel na uskutočnení zmien. Pedagóg podporil naše úsilie zreálniť pôvodný plán a radšej spravíme niečo ucelené a menšie ale použiteľné ako zanechať nedokončený produkt. Takto sa bude môcť hneď nasadiť do prevádzky a otestovať niektoré pôvodné myšlienky.
- Rozhodli sme sa nezačať stavať na zelenej lúke a za týmto účelom sme si vybrali už existujúci systém na podporu dištančného vzdelávania – Moodle (2.1.1)[1].

[1] Celý náš produkt nesie názov Dagwood. Systém Dagwood pozostáva s niekoľkých submodulov. Jedným z nich je naša modifikovaná verzia Moodle v1.1.1.

7.1.1 Zmena z pohľadu systému

Zásadnou zmenou špecifikácie z pohľadu systému (systémovej urovni) je úprava architektúry. Keďže ideme použiť systém Moodle, ktorý má niektoré nami navrhované moduly obsiahnuté priamo v sebe (báza konceptov, testovanie znalostí, správa používateľov a manažment) nie je dôvod (a ani čas) ich zásadne zmeniť a teda ich iba upravíme podľa našej špecifikácie. Dôsledkom tejto zmeny budú ako „web“ služby implementované len subsystémy na testovanie funkcionality a zisťovanie plagiátorstva.

Na prepojenie našich subsystémov (funkčnosť a plagiátorstvo) a teda odovzdávanie si súborov, výsledkov testovania a iných informácií s Moodle použijeme jednoduchý princíp založený na „spool“ démone[2]. Takýto démon sa používa napr. pri používaní tlačiarne procesmi kde úlohu nemožno elegantne riešiť cez zámky, pretože procesy sa môžu zaseknuť vo fronte ak pri momentálne tlačiacom procese nastane hardvérová chyba alebo nedostatok papiera a pod. Nepríjemnosť takejto situácie znepríjemňuje fakt, že zostanú „visieť“ aj procesy nepatriace momentálne tlačiacej aplikácie a teda možno aj iného používateľa. Pri používaní démona procesy nahrajú do dohodnutého adresára dokument na tlač a jeho samotnú tlač (a teda vybavenie požiadavky) zaobstará samotný démon. Proces môže teda ďalej pokračovať v behu bez rizika nečakaného zaseknutia pri tlači dokumentu.

7.1.2 Báza konceptov

Rozhodli sme sa ako základ bázy použiť tú, ktorú už v sebe obsahuje Moodle s tým, že je ju nevyhnutné vylepšiť. Vymýsľať a implemtovať úplne novú bázu je pre nás neefektívne a preto použijeme to, čo už existuje ale s našimi vylepšeniami. Zoznam nami navrhovaných vylepšení:

- Zaviesť v Moodle tzv. pohľady (perspektívy, náhľady), ktoré uľahčia orientáciu v prezentácii predmetu. Predmet je prezentovaný ako postupnosť sekcií (reprezentujú buď týždeň alebo tému), ku ktorým sa dali pridávať inštancie rôznych modulov (fórum, diskusia, zadanie, zdroj atď.). Pri voľbe príslušného pohľadu sa zobrazia iba tie inštancie modulov, ktoré spolu logicky súvisia, pričom jedna inštancia sa môže nachádzať v niekoľkých pohľadoch. Vytváranie pohľadov a prideľovanie inštancií modulov do nich má na starosti učiteľ. Ide teda o rôzne pohľady na tú istú bázu konceptov. Každý predmet môže mať ľubovoľný počet náhľadov.
- Zaviesť nasledovné typy pohľadov:
    - Strom: do uzlov sa budú dať pridávať buď inštancie modulov alebo iba textový názov. Vhodné, na prezentovanie hierarchického prepojenia použitých konceptov.

[2] Démon je systémový proces, ktorý je neustále prítomný za behu systému. Ak má prázdny rad požiadaviek na obsluhu „odpočíva“ a neuberá bežiacim procesom strojový čas (je čakajúcim).

- Jedna: iba jedna sekcia. Budú sa priraďovať moduly, ktoré majú všebecný charakter a nedajú sa jednoznačne priradiť do konkrétneho týždňa semestra ako napr. základné informacie o predmete, výsledky písomiek a pod.
- Mnoho: jedna sekcia na týždeň semestra t.j. každý týždeň má vlastnú sekciu. Vhodné najmä tam, kde je treba moduly postupne usporiadať do jednolivých týždňov semestra (napr. cvičenia, prednášky).
- Celok: spojenie Single a Multi.

- Pri type náhľadu strom umožniť vzájomné referencovanie medzi HTML dokumentmi.
- Vložiť nové bloky ako kalendár, blížiace sa udalosti a často kladené otázky.

7.1.2.1 Opis údajov

V pohľadoch sa bude pracovať s modulmi systému Moodle, konkrétne ich inštanciami v jednotlivých kurzoch. Moduly budú môcť byť priradené do pohľadov a teda sa im rozšíria pôvodné vlastnosti o informáciu, do ktorých pohľadov patria. Pohľad má tieto vlastnosti:

- Viditeľnosť: či je daný pohľad dostupný aj pre študentov.
- Typ: jeden zo štyroch vyššie spomínaných.
- Názov: charakteristika modulov, ktoré su v ňom priradené.
- Kurz: ktorého kurzu (id) je súčasťou.

Pri práci s typom Strom sa pracuje s jeho uzlami. V uzle sa môže nachádzať buď iba nadpis alebo inštancia modulu. V pohľade bude existovať iba jeden strom. Uzol bude mať minimaĺne tieto vlastnosti:

- Rodič: identifikátor na rodiča.
- Sused: identifikátor na suseda vpravo.
- Pohľad: identifikátor pohľadu, ktorý strom reprezentuje.
- Nadpis: názov uzla.
- Modul: identifikátor inštancie modulu, ktorý sa v uzle nachádza (pokiaľ nie je definovaný nadpis).

7.1.2.2 Opis funkcií

Vytvor pohľad	
Popis	Vytvorí sa v danom predmete nový pohľad, ktorý bude mať dané vlastnosti. Pohľad bude implicitne na začiatku nastavený na neviditeľný a priradené unikátne id v rámci systému.
Vstup	• typ pohľadu • id kurzu • názov
Výstup	• potvrdenie vytvorenia

Zmaž pohľad	
Popis	Výmaz pohľadu zo systému. Súčasne sa zabezpečí vnútorná konzistencia tabuliek databázy systému a teda dereferencovanie pohľadu v moduloch, ktoré ho používajú.
Vstup	• id pohľadu
Výstup	• potvrdenie výmazu

Nastav viditeľnosť	
Popis	Nastaví požadovanú viditeľnosť pohľadu (viditeľny/skrytý).
Vstup	• id pohľadu • viditeľnosť
Výstup	• potvrdenie nastavenia

Načítaj pohľady v kurze	
Popis	Vráti zoznam pre používateľa dostupných pohľadov v kurze. Funkcia si overí rolu používateľa. Pri učiteľovi zobrazí všetky vytvorené pohľady v systéme. Pri študentovi sa skontroluje atribút viditeľnosti.
Vstup	• id predmetu • id používateľa
Výstup	• id zoznam dostupných pohľadov

Priraď modul do pohľadu	
Popis	Priradí do pohľadu modul. Funkcia zapíše (ak ešte nebol priradený) do atribútu prítomnosť v pohľadoch id pohľadu.
Vstup	• id pohľadu • id inštancie modulu
Výstup	• potvrdenie o pridani

Odstráň modul z pohľadu	
Popis	Ak modul bol do pohľadu priradený, tak sa id daného pohľadu zmaže zo zoznamu pohľadov, do ktorých je modul priradený.
Vstup	• id predmetu • id inštancie modulu
Výstup	• potvrdenie o odstránení

Vráť moduly v pohľade	
Popis	K danému id pohľadu vráti id zoznam inštancií modulov.
Vstup	• id pohľadu
Výstup	• id zoznam inštancií modulov

Pridaj dieťa k uzlu	
Popis	Vytvorí k danému uzlu potomka s danými vlastnosťami. Nové dieťa bude pripojené na koniec zoznamu detí daného rodiča.
Vstup	• id uzla (rodiča) • pohľad • nadpis alebo id inštancie modulu
Výstup	• potvrdenie o vykonaní operácie

Vráť potomkov	
Popis	K danému id uzla vráti usporiadaný zoznam jeho detí.
Vstup	• id uzla (rodiča)
Výstup	• potvrdenie o vykonaní operácie

Vráť koreň	
Popis	K danému pohľadu vráti id koreňa.
Vstup	• id pohľadu
Výstup	• id koreňa

Posuň uzol vpravo/vpravo	
Popis	Posunie daný uzol (pokiaľ sa dá) o jedno miesto doprava/doľava.
Vstup	• id uzla
Výstup	• potvrdenie o vykonaní operácie

Zmaž uzol	
Popis	Vymaže daný uzol vrátane jeho podstromu.
Vstup	• id uzla
Výstup	• potvrdenie o vykonaní operácie

Zmeň nadpis uzla	
Popis	Ak je to možné (ak k uzolu nie je priradená inštancia modulu), zmení nadpis uzla.
Vstup	• id uzla • nadpis
Výstup	• potvrdenie o vykonaní operácie

7.1.3 Správa používateľov

Moodle má už v sebe implementovanú správu používateľov vrátane podpory rôznych autentifikačných mechanizmov. Pôvodne sme chceli v produkte nasadiť projekt Liberty Alliance (3.10.4). Vzhľadom na nedostatok času na jeho implementáciu sme sa obzreli po jeho voľne dostupnej implementácii. Momentálne bol voľne dostupný iba projekt SourceID (www.sourceid.org), ktorý je kompletne implementovaný v jazyku Java a spúšťa sa pod HTTP serverom Jakarta-TomCat. Toto je však nezlúčiteľné so systémom Moodle, ktorý je kompletne naprogramovaný v PHP. Museli sme preto upustiť od autentifikácie používateľov založenej na Liberty Alliance.

Budúci správca systému plánuje na FIIT zaviesť autentifikáciu cez LDAP. Vzhľadom k tomu, že včase riešenia projektu nebola táto služba presne špecifkovaná tak isto sme ju nezahrnuli do správy používateľov. Moodle má však v sebe čiastočne implemetovanú podporu LDAP a teda do budúcnosti sa dá bez problémov rozšíriť.

Keďže nemôžme naplniť pôvodnú špecifikáciu rozhodli sme sa implementovať podporu osobných čísel a hromadného pridávania (importovania) používateľov do Moodle. V dátovom modeli Moodle nastane zmena iba pri entite *User*, kde pridáme textový atribút s názvom *Os_cislo*. Špecifikácie funkcie import:

Hromadný import používateľov	
Popis	Funkcia bude dostupná iba administrátorovy Moodle. Daný súbor s informáciami o používateľoch rozparsuje a vytvorí nových používateľov v systéme Moodle.
Vstup	• Súbor: riadok na jedného používateľa s loginom, menom, priezviskom, heslom (sifrovanym, nesifrovanym), osobným číslom a email adresou.
Výstup	• potvrdenie o vykonaní operácie

7.1.4 Odovzdávanie zadaní

Oproti zimnému semestru sa špecifikácia tohto modulu z funkčného hľadiska zmenila minimálne, pričom hlavnou príčinou menšej úpravy oproti pôvodnému návrhu bolo využitie existujúceho systému Moodle, ktorého súčasťou už bola jednoduchá správa zadaní.

Pôvodné požiadavky, ktoré boli kladené na modul, súviseli len so samotnou správou zadaní. Boli to nasledovné požiadavky:

- prijatie a uloženie zadania na základe identifikátora (študenta),
- poskytnutie vybraného zadania, resp. zadaní podľa požiadaviek,
- odstránenie zadania,
- evidencia rôznych verzií zadania.

Oproti uvedeným požiadavkám bola špecifikácia doplnená o ďalšie funkcie, ktorými tento modul prebral časť funkcionality pôvodného modulu manažmentu predmetu:

- vytváranie a definícia nového zadania v predmete,
- rozhranie pre aktiváciu kontroly funkčnosti,
- rozhranie pre aktiváciu kontroly plagiátorstva,
- zobrazenie výsledkov kontrol.

Okrem prvého z menovaných bolo je potrebné do existujúceho systému Moodle doimplementovať tieto dodatočné funkcie podľa nasledovnej špecifikácie.

7.1.4.1 Popis údajov

Modul pracuje s odovzdanými zadaniami (súbor) a so samotnými inštanciami modulu zadanie (zadanie ako znenie úlohy a akcie nad nim t.j. odovzdanie, testovanie a posd.).

7.1.4.2 Popis funkcií

Odovzdaj zadanie	
Popis	Prijatie a uloženie zadania na základe identifikátora (študenta) a časovej pečiatky (verzie).
Vstup	• ID kurzu, • ID zadania predmetu, • ID študenta • Súbor s vybracovaným zadaním
Výstup	• potvrdenie uloženia

Poskytni zadania	
Popis	Poskytnutie všetkých odovzdaných zadaní študenta.
Vstup	• ID kurzu, • ID zadania predmetu, • ID študenta
Výstup	• zoznam všetkých odovzdaných zadaní študenta

Poskytni verziu zadania	
Popis	Poskytnutie konkrétneho odovzdaného zadania (verzie zadania) študenta.
Vstup	• ID kurzu, • ID zadania predmetu, • ID študenta • ID zadania
Výstup	• požadované odovzdané zadanie

Vymaž zadanie	
Popis	Vymazanie odovzdaného zadania.
Vstup	• ID kurzu, • ID zadania predmetu, • ID študenta • ID zadania
Výstup	• potvrdenie vymazania

Otestuj zadanie na funkcionalitu	
Popis	Zaradí zadanie do testu funkcionality a poskytne výsledok tohto testu.
Vstup	• ID kurzu, • ID zadania predmetu, • ID študenta • ID zadania
Výstup	• výsledok testu funkcionality

Otestuj zadania na plagiátorstvo	
Popis	Zaradí zoznam zadaní do testu plagiátorstva, otestuje ich na plagiátorstvo, a poskytne výsledky tohto testu.

Vstup	• ID kurzu,
	• ID zadania predmetu,
	• zoznam zadaní:
	  ○ ID študenta
	  ○ ID zadania

Výstup	• výsledok testu plagiátorstva

7.1.5 Testovanie vedomostí

Oproti zimnému semestru sa špecifikácia tohto modulu zmenila do formy, v akej je tento modul implementovaný v systéme Moodle. Keďže existujúci modul testovania vedomostí dostatočne pokrýva požiadavky na neho kladené, rozhodli sme sa tento modul neupravovať, a ponechať ho v pôvodnom tvare. Existujúce riešenie neobsahuje nasledovné možnosti, definované v zimnom semestri:

- podpora adaptívnych testov,p
- podpora adaptívneho testovania,
- import a export testov vo formáte QML,
- podpora odpovede typu „text“,
- postupné zobrazovanie otázok.

7.1.5.1 Ďalšie možnosti úprav

Vzhľadom na nedostatok časových prostriedkov sme nezaradili do požiadaviek ďalšie možnosti, ktoré by mohol tento modul obsahovať. Jedná sa vlastne o požiadavky, stanovené v zimnom semestri:

- Nami navrhovaný modul testovania znalostí by mal obsahovať i možnosť otázky typu 'program', čo však vyžaduje prítomnosť odladovacieho nástroja, a nástroja na otestovanie funkčnosti. Pokiaľ nie je potrebné testovať funčnosť, je postačujúce doplniť typy otázok o 'text', t.j. viacriadková odpoveď.
- Ďalšou požiadavkou je možnosť postupného zobrazovania otázok, t.j. nie celého testu naraz. Táto požiadavka vyžaduje vytvoriť mechanizmus navigácie medzi otázkami, a skript pre vyhodnocovanie a ukladanie čiastkových odpovedí.
- Špeciálnym typom testov sú tzv. adaptívne testy, kedy je nasledujúca otázka vyberaná zo skupiny otázok podľa úspešnosti zodpovedania predchádzajúcej otázky. Toto možno dosiahnuť úpravou (resp. vytvorením podobnej) otázky typu 'náhodný výber zo skupiny otázok', kde by bola každej z tejto otázok priradená váha, a pri požiadavke na ďalšiu otázku by sa vybrala otázka s príslušnou vypočítanou váhou.
- V návrhu nášho modulu testovania sme uvažovali i s komplexným testovaním, t.j. s vytváraním skupín testov, ktoré sa vykonávajú po sebe, pričom i tu by bola možnosť adaptívnosti na úrovni testov. Realizácia tejto možnosti by vyžadovala vytvorenie nového typu otázky, a síce 'otázka typu test'. Takýto test, obsahujúci subtesty, musí mať nastavené postupné zobrazovanie otázok.
- I keď existujúci modul podporuje import testov v rôznych formátoch, žiadny z týchto formátov nie je dostatočne komplexný, preto je vhodné doplniť možnosť importu a export údajov z/do formátu QML (Quzit Markup Language), alebo XML.

7.1.6 Démon

Dagwood Daemon (ďalej DD) bude slúžiť na prepojenie webového rozhrania systému Dagwood s modulmi. Je určený na nepretržitý beh na hostiteľskom systéme. Ďalšou podmienkou funkčnosti v systéme je to, aby bežal na tom istom systéme, ako aj webové rozhranie systému. Dôvodom je to, že tieto dve časti systému si budú odovzdávať údaje prostredníctvom súborov vytváraných na súborovom systéme.

DD bude používať na komunikáciu so svojim okolím niekoľko spôsobov. Prvým spôsobom je jednoduchý protokol špeciálne na tento účel. Malo by ísť o jednoduchý protokol, v ktorom „web“ rozhranie systému (PHP skripty) zadáva požiadavky, ktorých vybavenie zabezpečuje DD. DD bude

počúvať na TCP porte na „loopback“ adrese 127.0.0.1 (štandardne by malo ísť o port 23000). Po nadviazaní TCP spojenia a zadaní požiadavky odpovie DD cestou k súboru, do ktorého očakáva odovzdanie vstupných údajov požiadavky. Pri niektorých požiadavkách (porovnanie zadaní) môže ísť aj o viac súborov.

Po prijatí požiadavky DD bude kontaktovať modul zodpovedný za vykonanie požadovanej úlohy. V súčasnosti ide o modul porovnávania zadaní a modul testovania funkčnosti. Komunikácia s týmito modulmi je bude bežať protokolom SOAP. Toto umožňuje vytvoriť konfigurácie, kedy je „jadro systému“ („web“ rozhranie a DD) na jednom počítači, moduly na iných počítačoch. Výhodou tohto modelu je, že priamo počíta s možnosťou rozloženia záťaže na viac počítačov. Po odovzdaní požiadavky modulu DD bude čakať na jej vyriešenie. Aby bolo zabránené blokovaniu DD počas vykonávania nejakej úlohy, DD vytvorí na vykonanie požiadavky vždy nový proces, ktorý danú požiadavku obsluhuje. Po prijatí odpovede od modulu uloží DD výsledok do súboru na definované miesto. Pri nasledovnej požiadavke z „web“ rozhrania vráti cestu k súboru s odpoveďou.

7.1.7 Testovanie funkčnosti a plagiátorstva

Oproti zimnému semestru nedošlo k žiadnej zmene špecifikácie týchto dvoch subsystémov.

7.2 Rozdelenie projektu

Rozdelenie dlhodobých úloh na implementácii projektu bolo nasledovné:

Bc. Ľudovít Fülöp:

- Testovanie znalostí (zrušená pre nedostatok času a použiteľnosť súčasne implementácie v Moodle, možné vylepšenia sú opísané v kap. 7.1.5).
- Podpora osobných čísel v Moodle.
- Hromadný import používateľov.
- Modifikácie odovzdávania zadaní za účelom prepojenia s démonom a podpori viac verzií zadaní.

Bc. Martin Lacko:

- Zisťovanie plagiátorstva.

Bc. Ivan Malich:

- démon na prepojenie Moodle s podsystémami na testovanie funkčnosti zadaní a zisťovania plagiátorstva,
- inštalačný skript produktu.

Bc. Dalimír Orfánus:

- Vloženie Liberty Alliance do Moodle (zrušená pre nedostatok času a technické problémy – existujúca voľne dostupná implementácia je postavená ako Java Servlet zatiaľ čo Moodle na PHP)
- Vylepšenie a úpravy bázy konceptov.

Bc. Ivan Straka:

- podsystém testovania funkčnosti zadaní,
- podpora osobných čísel v Moodle,
- hromadný import používateľov do Moodle.

7.3 Báza konceptov

Báza konceptov je jedným z niekoľkých zdrojov informácií v navrhovanom systéme a súčasne je zodpovedná za ich prezentáciu používateľovy. Oproti pôvodnej báze v Moodle je naša rozšírená o tzv. pohľady, cez ktoré sa dá na ňu pozerať z rôznych „uhlov“.

Pracovali sme na nad vezrziou Moodle 1.1.1 a všetky modifikované súbory sa nachádzajú v adresári „course“ pokiaľ nebude v ďalšom texte uvedené inak.

7.3.1 Opis návrhu

7.3.1.1 Základňa pre podporu pohľadov

Pre podporu pohľadov bolo nutné vykonať najprv tieto úlohy:

1. Vložiť do databázy novú tabuľku, ktorá udržiava informácie o typoch pohľadoch (meno, ktorý súbor ho implementuje a či je v systéme dostupný pre učiteľov).
2. Pridať daľsiu tabuľku, ktorá reprezentuje väzobnú entitu medzi typom pohľadu a predmetom. Riadok predstavuje jeden pohľad v danom predmete.

3. Do tabuľky mapujúcej inštanciu modulu k predmetu vložiť nový stĺpec, kde bude uložené, v ktorých pohľadoch sa inštancia modulu nachádza.
4. Ďalej naslodalo pridanie nového bočného bloku navrch do ľavého stĺpca, v ktorom je menu na výber pohľadov.
5. Vytvorenie stránky, kde sa pohľady manažujú, ktorá bude dostupná len pre učiteľa.
6. Úprava *view.php* aby sa pri generovaní stránky kurzu podľa „id“ pohľadu zistilo, ktorý súbor ho implementuje a po predspracovaní údajov sa vyvolá a odovzdá sa mu ďalšie riadenie. Pri nezadaní „id“ pohľadu (prvé načítanie stránky predmetu) sa vyberie implicitný pohľad a to *Celok* (názov a je súčasne aj typom *celok* a jeho „id“ je 0).
7. Vytvorenie tabuľky *tree_view*, kde budú uložené všetky stromy.

Pohľad *Celok* obsahuje v sebe všetky moduly, ktoré sa pridali do predmetu. V režime editovania je ich možné odtiaľto pridávať do pohľadov. Ak sa prezerá nie implicitný pohľad v režime editovania, potom je možné moduly iba uberať. Diagram údajov –fyzický model ako existoval pred našou zmenou (je zachytené iba jadro kurzu, každý modul má svoj vlastný), je na Obr. 15. Pre prehladnosť a zachytenie podstaty myšlienky neuvádzame atribúty. Presný zoznam atribútov je možné dostať napr. SQL príkazom `describe nazov_tabulky;` keďže každá entita v modeli je implementovaná ako tabuľka databázy.

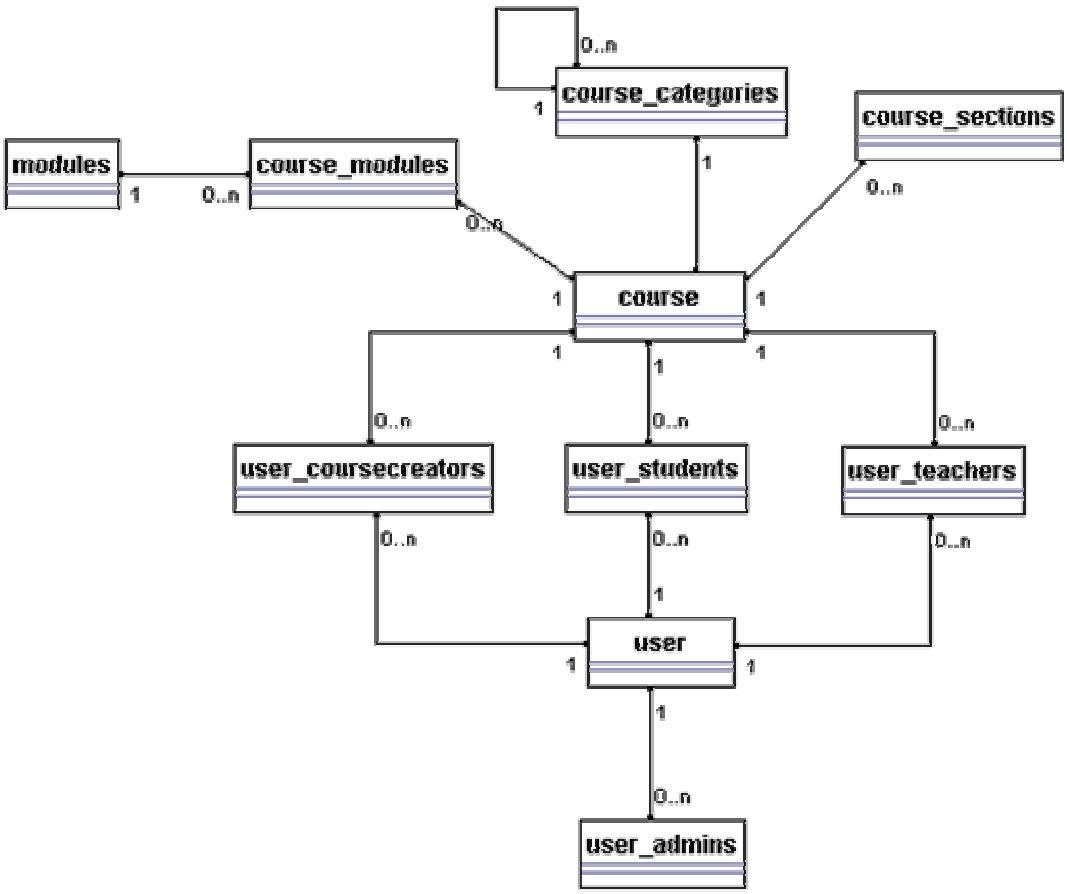
7.3.1.2 Typy pohľadov

Typy pohľadov mono, multi a celok netreba špecialne navrhovať. Stačí iba do *weeks.php* implementovať filtre pre zobrazované sekcie, t.j.:

- mono – iba sekcia 0,
- multi – sekcie 1 až n (n je počet týždňov kurzu),
- celok – sekcie 0 až n.

Typ strom je nutné navrhnúť úplne od začiatku. Pridávanie modulov do stromu sa bude realizovať tak, že sa v implicitnom pohľade priradí modul do pohľadu typu strom. Potom si učiteľ vyvolá zvolený pohľad. Obrazovka bude rozdelená na dve časti pod sebou. V prvej bude strom vrátane funkcií na jeho editovanie. V spodnej časti bude zoznam modulov, ktoré neboli asociované k uzlom stromu vráte funkcií na ich asociovanie. Teraz možno návrh rozdeliť na tieto časti:

- návrh abstraktného dátového typu (ADT) strom ,
- grafické rozhranie k stromu vrátane jeho editovania,
- mechanizmuz a rozhranie na úpravu uzlov stromu (mazanie, vytváranie, posúvanie, zmena nadpisu).

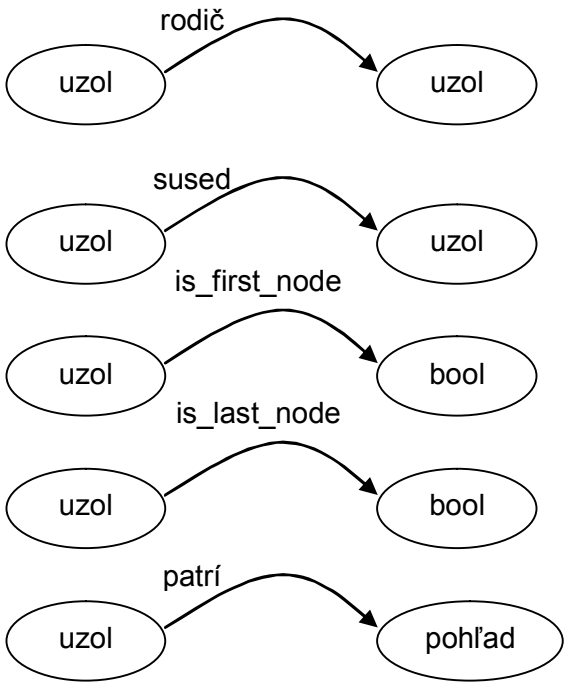


Obr. 15: Fyzický model údajov jadra Moodle pred modifikáciou

Návrh ADT strom vychádza z jednoduchej špecifikácie na Obr. 16. Špecifikácia zohľadnuje fakt, že strom bude uložený v databáze a teda napr. zoznam potomkov uzlov sa dá získať správnou otázkou, v našom prípade: vráť všetky uzly, ktoré maju daný uzol ako rodiča. Určiť prvého potomka je tiež jednoduché, keď sa otázka rozšíri o podmienku aby súčasne platilo, že uzol je *is_first_node*. Koreňový element bude mať nastavený identifikátor rodiča na nulu. Uzol bude rozšírený aj o informáciu, ku ktorému pohľadu patrí. Algoritmus zostavenia stromu z databázy je navrhnutý tak, že akceptuje aj možnost, že na jeden uzol ukazuje niekoľko iných ako na suseda. Dôvodom je zabránenie vzniku nekonzistentnosti databázy ako počas vykonávania operácií nad ňou (a s „id“ uzlov) padol systém, alebo by vlákno HTTP servera bolo prerušené. Tak isto sú navrhnuté aj ostatné algoritmy,

ktoré pracujú s atribútom „id“, aby sa nikdy nestalo, že priradí do neho hodnotu, ktorá by po nedokončení vykonávanej operácie znamenala nekonzistenciu údajov.

Grafické rozhranie bude rešpektovať fakt, že do jedného uzla je možné pridať iba jeden modul. Na zobrazovanie stromu sme použili JavaScript Tigra Tree Menu (http://www.softcomplex.com/products/tigra_menu/), ktorý bol voľne dostupný. Je napísaný univerzálne a sa iba správne inicializuje. Uzly sa dajú vybalovať a zbalovať. Za týmto účelom sme vložili do adresára *course/* tri súbory z tohto projektu (*tree_items.js*, *tree_tpl.js* a *tree.js*) a vytvorili adresár *course/img/*, kde sú ikonky použité v strome.



Obr. 16: Použitá špecifikácia ADT strom.

Rozhranie na manažovanie uzlov sa nachádza pod stromom ako „radio“ menu. Používateľ si zvolí požadovanú akciu a potom klikne na obrázok šípky (vykonaj zvolenú akciu) vedľa uzla. JavaScript najprv skontroluje podľa typu akcie parametre, prišom používateľa upozorní a potom sa príkaz na akciu spolu s parametrami odošle.

7.3.2 Súčasti

7.3.2.1 lib.php

Meno súboru	**lib.php**
Popis	Knižnica spoločných funkcií pre bázu znalostí. Zodpovedná za generovanie blokov, sekcií a pod.
Programovací jazyk	PHP + HTML

Modifikované a doplnené funkcie:

Názov	**print_section**
Popis	Vypisuje sekciu vrátane jej modulov a rôznych ovládacích prvkov. Vložil sa kod, ktorý za menom modulu vypiše zoznam pohľadov, v ktorých sa nachádza. V prípade režimu „Upráva kurzu“ vygeneruje vysúvacie „popup“ menu, kde je zoznam kompatibilných pohľadov, do ktorých sa modul môže priradiť.
Argumenty	Oproti pôvodnej funkcii nezmené.
Výstup	HTML kód sekcie

Názov	**get_modules_in_per**
Popis	Vráti zoznam inštancií modulov, ktoré patria do daného pohľadu
Argumenty	$course : object reprezentujúci daný kurz $per : id pohľadu
Výstup	Pole objektov v ktorom sú moduly v danom pohľade.

Názov	**show_views_name**
Popis	Vypíše zoznam pohľadov, v ktorých sa modul nachádza.
Argumenty	$ids : zoznam id pohľadov, vktorých je daná inštancia modulu. Je to stĺpec „view“ v tabuľke „course_modules“. Je to string, kde id pohľadov sú oddelené čiarkami. $courseid : id kurzu
Výstup	HTML kód pre zoznam mien pohľadov.

Názov	**print_perspective**
Popis	K danému kurzu predpripravý údaje na zavolanie funkcie „print_side_block“. Je to zoznam pohľadov, ktoré sú v danom kurze dostupné pre daného používateľa.
Argumenty	$course: object reprezentujúci daný kurz
Výstup	HTML kód pre zoznam mien pohľadov.

Názov	**add_to_comma_list**
Popis	Do daneho zoznamu prida dalsi prvok, pokial uz v nom nie je. Zoznam je reprezentovany ako string, kde polozky su oddelene ciarkami.
Argumenty	$list : zoznam $item : nova polozka zoznamu
Výstup	Zoznam s pridanou polozkou.

Názov	**remove_from_comma_list**
Popis	Z daneho zoznamu odoberie prvok, pokial v nom je. Zoznam je reprezentovany ako string, kde polozky su oddelene ciarkami.
Argumenty	$list : zoznam $item : nova polozka zoznamu
Výstup	Zoznam bez danej polozky.

Názov	**delete_mod_from_tree**
Popis	Vymaze instanciu modulu zo stromu ak bol modul priradený do pohľadu typu strom a bol aj priradený do uzla stromu.
Argumenty	$coursemodule : id instancie modulu
Výstup	-

Názov	**tree_delete_node**
Popis	Zmazanie celého podstromu. Najprv vsetky uzly oznacime rekurzivne ako deleted a potom zmazeme vsetky uzly z daneho pohladu nastavene ako deleted. Je to preto, aby sme nemuseli zamknúť a odomknúť tabuľku a aby si pri súčasnom načítavaní mazaného stromu iným používateľov nezobrazili nekonzistentné údaje.
Argumenty	$nodeid: id korena podstromu $per: id pohľadu
Výstup	-

Názov	**set_deleted_tree_node**
Popis	Rekurzivne oznacenie podstromu ako deleted.
Argumenty	$nodeid: id korena podstromu
Výstup	-

7.3.2.2 tree.php

Meno súboru	**course/format/tree.php**
Popis	Skript zodpovedný za kompletné vygenerovanie HTML kódu stredného stĺpca pri pohľade typu strom.
Programovací jazyk	PHP + HTML

Modifikované a doplnené funkcie:

Názov	**print_tree**
Popis	Funkcia si nechá vygenerovať a potom vloží do stránky JavaScript kód, ktorý zobrazuje strom a je zodpovedný aj za vyvolávanie a ošetrovanie akcií, ktoré požaduje vykonať používateľ. Vykonáva prvotnú kontrolu argumentov pri každej používateľovej požiadavke na modifikáciu stromu a keď je všetko v poriadku, potom vykaná „submit“ daných údajov.
Argumenty	$per : id pohľadu $course : objekt kurz
Výstup	HTML a JavaScript kód

Názov	**array_PHP2JS**
Popis	Rekurzívna funkcia, ktorá premení dané pole v PHP do stringu, ktorý reprezentuje inicializáciu identického pola v jazyku JavaScript.
Argumenty	$arr : PHP pole
Výstup	String, kde je inicializácia JavaScript pola

Názov	**construct_tree**
Popis	Rekurzívna funkcia. Z údajov v databáze vygeneruje PHP pole, ktoré repzentuje strom.
Argumenty	$nodeid : id uzla
Výstup	PHP pole

Názov	**find_sibling**
Popis	Rekurzívna funkcia. Najde suseda k danému uzlu a pridá ho do PHP pola, ktoré reprezentuje strom.
Argumenty	$tree : pole $node : id uzla, ku ktorému sa hľadá sused
Výstup	PHP pole

Názov	**find_sibling**
Popis	Rekurzívna funkcia. Najde suseda k danému uzlu a pridá ho do PHP pola, ktoré reprezentuje strom.
Argumenty	$tree : pole $node : id uzla, ku ktorému sa hľadá sused
Výstup	PHP pole

Názov	**is_first_node, is_last_node**
Popis	Funkcie zistia, či je daný uzol prvý alebo posledný.
Argumenty	$node : id uzla
Výstup	Boolean

Názov	**make_tree_editing_buttons**
Popis	K danému uzlu vygeneruje string, kde je HTML kód pre zobrazenie ovládacích prvkov. Momentálne je tam iba jeden prvok, ktorý vyvolá JavaScript funkciu PerformAction().
Argumenty	$node : id uzla
Výstup	String s HTML kódom.

Názov	**print_action_menu**
Popis	Zobrazí menu s dostupnými akciami pre modifikovanie stromu. Menu používa „radio“ prvky na výber požadovanej akcie.
Argumenty	-
Výstup	HTML kód pre menu.

Názov	**tree_add_child**
Popis	K danému uzlu stromu vytvorí jeho potomka. Potomok môže byť buď nadpis, alebo modul. Ak id modulu je 0, potom je to nadpis, ináč bude v uzle inštancia modulu.
Argumenty	$nodeid : id uzla (rodiča) $nodename : nadpis uzla $module : id inštancie modulu $per :
Výstup	HTML kód pre menu.

Názov	**tree_movedown_node, tree_movedup_node**
Popis	Posunie uzol vľavo/vpravo (pokiaľ je to možné).
Argumenty	$nodeid : id uzla
Výstup	Posunutý uzol.

Názov	**print_modules_not_assoc_in_tree**
Popis	Vypíše moduly, ktoré sú v danom pohľade typu strom, ale nie sú priradené do stromu.
Argumenty	$course : objekt kurz $per : id pohľadu
Výstup	HTML kód pre zoznam neasociovaných modulov.

7.3.2.3 view.php

Meno súboru	**view.php**
Popis	Skript, ktorý zobrazí domovskú stránku predmetu. Vygenruje hlavičku, zavolá skript na generovanie jadra a potom dovygeneruje koniec stránky.
Programovací jazyk	PHP + HTML

Modifikácie:

Skript je bez funkcií. Vložili sme do tela skriptu pridávanie a uberanie modulu z pohľadu a logiku na výber skriptu, ktorý je zodpovedný za generovanie typu zvoleného pohľadu.

7.3.2.4 weeks.php

Meno súboru	**view.php**
Popis	Skript, ktorý generuje jadro kurzu t.j. bočné bloky a sekcie. Zodpovedný sa zobrazovanie pohľadov typu: mono, multi a celok.
Programovací jazyk	PHP + HTML

Modifikácie:

Skript je bez funkcií. Podľa zvoleného typu pohľadu sa preskočí generovanie sekcií, ktoré do typu pohľadu nepatria (napr. sekcia 0 nepatrí do typu multi). Pridali sme generovanie bočného bloku – menu pre výber dostupných pohľadov.

7.3.2.5 perspective.php

Meno súboru	**perspective.php**
Popis	Nami pridaný skript na manažovanie (pridávanie, mazanie a nastavovanie viditeľnosti) pohľadov v kurze.
Programovací jazyk	PHP + HTML

7.3.2.6 mod.php

Meno súboru	**mod.php**
Popis	Mazanie, pridávanie, aktualizovanie alebo mazanie modulov.
Programovací jazyk	PHP + HTML

Modifikácie:

Skript je bez funkcií. Pri mazaní inštancie modulu sa vložilo vyvolanie funkcie *delete_mod_from_tree* (nachádza sa v „course/lib.php") aby sa po zmazaní modulu v strome neobjavil uzol bez pridaného modulu.

7.3.3 Modifikácie v databáze

7.3.3.1 Modifikované tabuľky

Názov	**course_modules**		
Popis	Väzobná entita medzi kurzom a modulom. Riadok predstavuje jednu inštanciu daného modulu v danom predmete.		
Doplnený parameter	view	text	Testový atribút, kde je v zozname uložený zoznam id pohľadov, v ktorých je modul asociovaný. Položky sú oddelené čiarkov.

7.3.3.2 Nové tabuľky

Názov	**view_types**		
Popis	Tabuľka s informáciami o typoch pohľadov inštalovaných v systéme.		
Parametre	id	int(3), unsigned, primary key, not null, auto increment	Identifikátor.
	name	varchar(20)	Názov typu pohľadu.
	file	varchar(20)	Názov súboru v adresári „course/format/*.php“, kde „*“ jeho meno (bez koncovky a typu PHP), zodpovedného za výpis daného typu pohľadu.
	visible	int(1), unsigned, default 1	Či je typ pohľadu dostupný na používanie v systéme. Momentálne sa nevyužíva.

Názov	**course_views**		
Popis	Väzobná entita medzi kurzom a typom pohľadu. Jeden riadok predstavuje jeden pohľad v danom predmete s daným typom.		
Parametre	id	int(10), unsigned, primary key, not null, auto increment	Identifikátor.
	course	int(10), unsigned, default 0	Cudzí kľuč tabuľky „course“.
	type	int(3), unsigned, default 0	Cudzí kľúč tabuľky „view_types“.
	name	varchar(20), default ‘changeme’	Meno pohľadu.
	visible	tinyint(1), unsigned, default 1	Či je pohľad viditeľný.

Názov	**tree_view**		
Popis	Väzobná entita medzi kurzom a typom pohľadu. Jeden riadok predstavuje jeden pohľad v danom predmete s daným typom.		
Parametre	id	int(10), unsigned, primary key, not null, auto increment	Identifikátor.
	parent	int(10), unsigned, default null	Id rodiča.
	sibling	int(3), unsigned, default null	Id suseda.
	last	tinyint(1), default 0	Či je posledný v úrovni. Nepoužíva sa.
	first	tinyint(1), default 0	Či je prvý v úrovni. Nepoužíva sa.
	per	int(10), default null	Cudzí kľúč do „course_views“.

	heading	text, default null	Nadpis.
	module	int(10), default null	Cudzí kľúč do „course_modules“.
	deleted	tinyint(1), default 0	Či sa má uzol zmazať.

7.3.4 Popis vybraných vnútorných algoritmov

Zobrazenie kurzu /(view.php)

1. Skontroluj, či používateľ môže prezerať kurz. Ak nie, presmeruj na login.
2. Ošetri príkaz zadaný cez GET a POST
   - 2.1. Ak skryť sekciu, skry.
   - 2.2. Ak zobraziť sekciu, zobraz
   - 2.3. Ak pridať modul do pohľadu, pridaj
   - 2.4. Ak ubrať modul z pohľadu, uber
   - 2.5. Ak posunúť sekciu, posuň
3. Vypíš hlavičku
4. Načítaj informácie o všetkých inštanciach modulov v kurze
5. Podľa typu zvoleného pohľadu vyvolaj daný skript
6. Vygeneruj pätu stránky

Generovanie pohľadov mono, multi a celok /(weeks.php)

1. Skontroluj, či používateľ môže prezerať stránku, ak nie, presmeruj na login
2. Vygeneruj ľavu časť obrazovky
   - 2.1. Zoznam pohľadov
   - 2.2. Odkazy na aktivity
   - 2.3. Formulár na prehľadávanie
   - 2.4. Odkazy na administrovanie
   - 2.5. Zoznam kurzov, v ktorých používateľ participuje
3. Vypíš hlavičku pre stredný stĺpec
4. Ak je pohľad typu celok a mono, generuj sekciu 0
   - 4.1. Predpriprav údaje
   - 4.2. Pre všteky sekcie zavolaj print_section()
     - 4.2.1. Vypíš ľavý rám
     - 4.2.2. Pre všetky moduly v sekcii:
       - 4.2.2.1. Získaj všteky informácie o moduloch v sekcii
       - 4.2.2.2. Zobraz ikonku inštancie, mena modulu
       - 4.2.2.3. Ak v režime upravovania, pridaj ovládacie ikony
       - 4.2.2.4. Ak to nie je implicitný pohľad (Celok), ikonka na zmazanie z tohto pohľadu
       - 4.2.2.5. Zisti pohľady, v ktorých sa nachádza a vypíš ich názvy
       - 4.2.2.6. Ak to nie je implicitný pohľad (Celok) a je v režime upravovania
         - 4.2.2.6.1. Zostroj zoznam možných (ak je v sekcii 0 nemôže patriť do multi a pod.) pohľadov na priradenie
         - 4.2.2.6.2. Vygeneruj popup menu
   - 4.3. Ak môže upravovať, vygeneruj bočné menu k sekcii
5. Ak je pohľad typu celok a multi, generuj sekcie 1..n
   - 5.1. Predpriprav údaje
   - 5.2. Zavolaj print_section()
   - 5.3. Ak môže upravovať, vygeneruj bočné menu k sekcii
6. Vygeneruj pätu stredného stĺpca stránky
7. Vygeneruj pravý stĺpec
   - 7.1. Odkazy k účastníkom
   - 7.2. Posledné novinky
   - 7.3. Poslednú aktivitu používateľa

Generovanie pohľadu typu strom /(tree.php)

1. Skontroluj, či používateľ môže prezerať stránku, ak nie, presmeruj na login
2. Vygeneruj ľavu časť obrazovky
   - 2.1. Zoznam pohľadov
   - 2.2. Odkazy na aktivity
   - 2.3. Formulár na prehľadávanie
   - 2.4. Odkazy na administrovanie
   - 2.5. Zoznam kurzov, v ktorých používateľ participuje
3. Vypíš hlavičku pre stredný stĺpec
4. Ak boli zadané akcia cez POST a GET a je učiteľ
   - 4.1. Ak pridať potomka k uzlu a sedia parametre, pridaj
   - 4.2. Ak zmazať podstrom a sedia parametre, zmaž
   - 4.3. Ak upraviť nadpis a sedia parametre, uprav
   - 4.4. Ak posunúť uzol vyššie/nižšie a sedia parametre, posuň
5. Vygeneruj strom
   - 5.1. Načítaj strom do pola z databázy
   - 5.2. Prekonvertuj strom z PHP pola do reťazca na inicializáciu JavaScript pola
   - 5.3. Vlož do stránky JavaScript kód na strom a funkcie na podporu dynamiky a kontrolu zadan
6. Ak je stránka v režime editovania, vygeneruj menu pre úpravy stromu
7. Vypíš zoznam inštancií modulov nepriradených do stromu
8. Vygeneruj pätu stredného stĺpca stránky
9. Vygeneruj pravý stĺpec
   - 9.1. Odkazy k účastníkom
   - 9.2. Posledné novinky
   - 9.3. Poslednú aktivitu používateľa

7.3.5 Ďalšie možnosti úprav

Vzhľadom na nedostatok času sme nezrealizovali nasledovné veci:

- Časová optimalizácia kódu generujúceho stránky bázy.
- Vytváranie log záznamov z práce s pohľadmi.
- Implementovanie viditeľnosti typov pohľadov v systéme. Toto je použiteľné napr. aj pri vyvoji nového typu pohľadu, kedy s daným pohľad pracuje iba administrátor.
- Vylepšenie manažovania stromu.
- Vymyslieť a implementovať nové typy pohľadov (podľa potrieb používateľov).

7.4 Zisťovanie plagiátorstva

Zmena nastala iba v ich architektúre tohto podsystému, kde sme ho rozdelili na dve časťi. Prvá časť zabezpečuje rozhranie a teda komunikáciu s vonkajším svetom. Druhá časť je binárny súbor, ktorý vykonáva zisťovanie plagiátorstva a všteky potrebné údaje su muzadávane cez prepínače. Samotné fungovanie je veľmi jednoduché. Rozhranie príjme cez SOAP protokol príkaz na vykonanie akcie. Ak treba vykonať testovanie, spustí binárny súbor s vhodnými prepínačmi. Ak treba načítať výsledky testovania, rozhranie ich načíta z vlastných súborov.

Takéto elegantné oddelenie komunikácie od spracovania je veľmi flexibilné a najmä bezpečné. V budúcnosti možno bez problémov preprogramovať samotnú funkcionalitu bez zmeny komunikácie.

Modul plagiátorstva je naprogramovaný v jazyku C s použitím príkazov podľa normy ANSI C, čím bola dosiahnutá jeho platformová nezávislosť. Samotný modul porovnávania zadaní je implementovaný v dvoch nezávislých submoduloch. Vstupy pre oba moduly sa zadávajú pomocou parametrov pre každý modul.

V prvej fáze sa z textového súboru obsahúceho vypracované zadanie vytvorí zoznam tokenov podľa zvolenej tabuľky tokenov. Toto tokenizovanie sa vykoná pre každé zadanie len jeden raz a ďalej sa už pracuje len s výsledným súborom tokenov. Samotné porovnanie zadaní sa vykonáva v druhej fáze, pričom porovnávanie sa vykonáva po dvojiciach zadaní (súborov tokenov) - pre každú dvojicu súborov tokenov je výsledkom číslo v rozsahu od 0 do 100, ktoré vyjadruje precentuálnu podobnosť dvoch zadaní.

7.4.1 Submodul vytvárania tokenov (token)

Meno súboru	**token.c**
Popis	Zdrojový kód submodulu vytvárania tokenov (súčasť modulu plagiátorstva).
Programovací jazyk	C (norma ANSI C)
Spúšťanie	Je spúštaný obslužným démonom pre plagiátorstvo – používateľ nemá k nemu priamy prístup
Prostredie	UNIX/Linux, Windows, ... je úplne systémovo nezávislý

Vstupom pre tento submodul je textový súbor s vypracovaným zadaním, výstupom je textový súbor so zoznamom tokenov. Vstupný súbor sa načítava po riadkoch a každý riadok sa spracúva podľa pravidiel definovaných v slovníku pre daný programovací jazyk. V spracovávanom riadku sa vyhľadávajú príkazy zo slovníka, ktoré sú zamenené za ekvivalentné tokeny. Z tokenov sa vytvárajú n-tice, t.j určený počet tokenov spoločne vytvorí jednu hodnotu a tá sa uloží do súboru. Tieto n-tice vlastne definujú granularitu, s ktorou sa porovnávajú zdrojové súbory. Ak je zvolená granularita nízka,

budú si i rozdielne zadania dosť podobné, ak je naopak veľká, program odhalí len veľmi málo kopírovaných zadaní. V praxi sa ukázalo pre programy napísané v JSI je najlepšie vytvárať a porovnávať trojice tokenov. Pri spracovávaní sa ignorujú komentáre a pre priradenie tokenu nie je rozhodujúci názov premennej, ale len jej typ, čo umožňuje odhaliť programy, ktoré boli modifikované premenovaním premenných a modifikáciou komentárov. Nastavenie parametrov tokenizácie a definovanie príkazov daného jazyka sa vykonáva pomocou slovníka. Pre odhalenie modifikácie programov zámenou príkazov za ich ekvivalenty sa odporúča rovnocenným príkazom priradiť rovnakú hodnotu tokenu. Slovník je textový súbor s nasledovnou štruktúrou:

```
NticeTokenov = <koľko príkazov má vytvárať jednu n-ticu>
TerminalneZnaky = <počet terminálnych znakov>
<zoznam znakov, ktoré ukončujú príkaz oddelených ENTERom>
KometareRiadkove = <počet riadkových komentárov>
<zoznam znakov, za ktorými nasleduje komentár do konca riadku>
KomentareBlokove = <počet blokových komentárov>
<zoznam znakov, ktoré ohraničujú viacriadkový komentár>
PocetPrikazov = <počet príkazov, ktorým sú priradené tokeny>
<zoznam tokenov a príkazov štruktúre: token,príkaz>
```

Za údajmi sa môžu nachádzať poznámky, ale musia začínať znakmi '//', príklad:

```
NticeTokenov = 3
TerminalneZnaky = 7
32   //medzera
44  //ciarka
9 //tabulator
10 //enter
58 //dvojbodka
91 //zatvorka: [
93 //zatvorka: ]
KometareRiadkove = 1
; //bodkociarka
KomentareBlokove = 0
PocetPrikazov = 7
1,segment
2,ends
3,start
4,dw
5,assume
6,db
7,dq
```

Spustenie tokenizovania sa vykoná zadaním názvu submodulu a parametrov do príkazového riadka nasledovne:

```
token <slovník> <vstupný_súbor> <vystupný_súbor>
```

V prípade úspešného vykonania program vráti hodnotu 0, v prípade chyby hodnotu -1 a na štandardný chybový výstup je vypísaná príčina chyby.

Popis funkcií:

Názov	**main**
Popis	Kontrola vstupných parametrov a spustenie tokenizovania.
Argumenty	int argc,char*argv[]
Návr. hodnota	int

Názov	**PrehladajSubor**
Popis	Funkcia vykonáva prehľadávanie zdrojového súboru zadania a pre každý reťazac volá funkciu Porovnaj.
Argumenty	char *NazovSlovnika, char *MenoSuboru, char *CelaCesta
Návr. hodnota	-

Názov	**Porovnaj**
Popis	Funkcia zistí, či je načítaný reťazec príkaz daného jazyka.
Argumenty	char* Retazec
Návr. hodnota	int

Názov	**PridajToken**
Popis	Vytvára a ukladá n-tice tokenov do výstupného súboru.
Argumenty	int NajdenyToken, FILE *ZnakovySubor
Návr. hodnota	-

Názov	**Hashfun**
Popis	Vypočíta pozíciu pre načítaný reťazec v hash tabuľke.
Argumenty	char* Vstup
Návr. hodnota	int

7.4.2 Submodul porovnávania dvoch súborov tokenov (compare)

Meno súboru	**compare.c**
Popis	Zdrojový kód submodulu porovnávania dvoch súborov tokenov (súčasť modulu plagiátorstva)
Programovací jazyk	C (norma ANSI C)
Spúšťanie	Je spúšťaný obslužným démonom pre plagiátorstvo – používateľ nemá k nemu priamy prístup.
Prostredie	UNIX/Linux, Windows, ... je úplne systémovo nezávislý

Porovnávaním súborov tokenov sa zisťuje koľko n-tíc tokenov (t.j. príkazov) je v oboch zadaniach zhodných. Na základe porovnaní trojíc a ich výskytu v oboch súboroch tokenov sa určí percentuálna podobnosť zadaní. Toto porovnanie ja zamerané na odhalenie zadaní, ktoré boli modifikované preusporiadaním príkazov alebo funkcií. Samotné porovnávanie je implementované pomocou hash tabuľky, do ktorej sa načíta jeden súbor s tokenmi a následne sa z druhého súboru čítajú tokeny a zaznamenáva sa ich rovnaký výskyt.

Spustenie submodulu sa vykoná zadaním názvu submodulu a parametrov do príkazového riadka nasledovne:

```
<compare> <súbor_s_tokenmi1> <súbor_s_tokenmi2>
```

Výstupom je percentuálna podobnosť dvoch zadaní, ktorá sa vypíše na štadardný výstup.

V prípade vyskytnutia sa chyby, je vypísaná hodnota -1 a na štandardný chybový výstup je vypísaná príčina chyby.

Popis funkcií:

Názov	**main**
Popis	Kkontrola vstupných parametrov a spustenie tokenizovania.
Argumenty	int argc,char*argv[]
Návr. hodnota	int

Názov	**NacitajTokeny**
Popis	Načíta tokeny zo zdrojových súborov do pamäte.
Argumenty	char* MenoSuboru,long *PoleTokenov
Návr. hodnota	int

Názov	**Porovnaj2Subory**
Popis	Vykonáva vlastné porovnávanie súborov s tokenmi. Vracia percentuálnu podobnosť zadaní.
Argumenty	long *Subor1,int Pocet_Subor1,long *Subor2,int Pocet_Subor2
Návr. hodnota	float

7.5 Testovanie funkčnosti

Aj tu sme aplikovali rovnakú myšlienku ako pri podsystéme zisťovania plagiátorstva a rozdelili sme tento modul presne ako predchádzajúcom prípade na dve časti a oddelili ta komunikáciu od spracovania.

Hlavnou úlohou modulu je testovanie zadaní po funkčnej stránke. Na dosiahnutie najpresnejšieho testovania, je potreba testovať program v prostredí, pre ktorý bol napísaný, preto najlepším riešením je testovať programy písané pod OS-MSDOS tiež pod OS MSDOS. Ale z hľadiska bezpečností sme navrhli a zvolili riešenie používané emulované prostredie toho operačného systému. Toto riešenie umožňuje vierohodné testovanie v bezpečnom testovacom prostredí. Kedže pracujeme nad platformou OS GNU/Linux zvolili sme emulátor prostredia MSDOS *dosemu*. Tento emulátor nám umožňuje spúšťať programy pre MSDOS pod GNU/Linux pričom samostatné testovanie bude prebiehať pod týmto emulovaným prostredím.

Kedže modul treba aj testovať pre tieto účely poskytuje rôzne výstupy a to predovšetkým normálny a chybový výstup z emulátora a výstupy jednotlivých fáz testovania (tieto sa vykonávajú v emulovaným prostredí). V prípade chyby sa preto dá zistiť či nastala nejaká chyba pri testovaní, alebo to bola chyba spôsobená emulátorom.

Celé testovanie prebieha pod emulovaným prostredím, tj. pre programy sa javí ako keby bol spustený pod OS MSDOS. Preto sa použili programy na skompilovanie, linkovanie a porovnávanie napísané tiež pre túto platformu.

7.5.1 Popis jednotlivých súborov

7.5.1.1 Testuj

Meno súboru	**bin/testuj.sh**
Popis	Spúšťací skript testovania. Najskôr skontroluje predané parametre, následne pripraví pracovný adresár, skopíruje potrebné súbory a spustí program rundos s potrebnými parametrami. Nakoniec upraví konečný výstup testovania.
Programovací jazyk	Shell

Modifikácie:

- Pridanie a spracovanie argumentov.
- Pridanie konfiguračného súboru (voliteľné nastavovanie ciest k jednotlivým súborom).
- Pridanie nápovedy.
- Zmena prípravy testovania (zmenená štruktúra adresárov, generovanie spúšťacieho dos-batch súboru).
- Pridanie testovania návratu programu „rundos“ - spracovanie chyby pri vypršaní času.
- Upravené generovanie výstupu (možnosť výstupu aj na štandardný výstup aj do súbora).

7.5.1.2 Rundos

Meno súboru	**bin/run_dos, src/run_dos.c**
Popis	Program slúži na spúšťanie programov a kontrolu ich ukončenia(umožňuje ho ukončiť po určitom čase). Spustenie programu je vyvolané v dcérskom procese tohto programu, ktorý je ukončený buď svojou činnosťou alebo násilne po uplynutí určitého časového intervalu.
Programovací jazyk	C
Využitie	Používané na spustenie dosemu. Takto je ošetrená situácia, že testovaný program sa sám neukončí.

Modifikované funkcie:

Názov	**main**
Popis	Hlavná funkcia programu. Vypustilo sa alokovanie tty-čiek (alloc_tty) a pridané rúry na komunikáciu so spúšťaným programom. Tieto rúry nahradili deskriptory tty zariadenia.
Argumenty	timeout, program, parametre spúšťaného programu
Výstup	-1 : chyba pri vytváraní dcérskeho procesu 0 : program sa spustil a ukončil svoju činnosť v stanovenom čase 1 : program sa spustil ale bol násilne ukončení po vypršaní zadaného času

7.5.1.3 Todos

Meno súbor	**bin/todos.sh**
Popis	Formátuje názov testovaného programu (odstraňuje príponu).
Programovací jazyk	C-Shell
Využitie	Pri generovaní spúšťacieho dos-batch súboru.

7.5.1.4 Asmtst

Meno súbor	**var/dos/asmtst.bat**
Popis	Dávkový súbor, ktorý riadi a spúšťa jednotlivé programy pod dosemu. V podstate riadi cele testovanie. Najskôr spúšťa proces skompilovania, následne linkovania, testovania testovaného programu a vyhodnotenia výstupu testovaného programu. Ak neprebehne niektorá z týchto fáz úspešne ďalšie fázy budú preskočené a výsledkom tejto fáze bude chyba. Nakoniec upraví výsledky jednotlivých fáz testovania.
Programovací jazyk	DOS/batch
Využitie	Riadi samotné testovanie pod prostredím DOS.

Modifikácie:

- upravené generovanie výstupov jednotlivých fáz aj celkové zostavenie výstupu.

7.5.2 Nemodifikované programy modulu

Meno súbor	**var/dos/tasm.exe**
Popis	Turbo asembler. Slúži na skompilovanie testovaného programu. Výstupom je skompilovaný program.
Použité parametre	Názov testovaného programu (meno.asm)

Meno súbor	**var/dos/tlink.exe**
Popis	Turbo link. Slúži na linkovanie testovaného programu. Výstupom je zlinkovaný program (spustiteľný súbor).
Použité parametre	Názov skompilovaného programu (meno.obj)

Meno súbor	**var/dos/asmtst.com**
Popis	Slúži na otestovanie testovaného programu. Emuluje stláčanie klávesnice a zaznamenáva výpisy na obrazovku.
Použité parametre	keys.def – súbor s klávesmi, ktoré sa majú emulovať. (vstupná testovacia vzorka) screen.log – výstupný súbor (záznam obrazovky) %1.exe – názov programu (testovaný spustiteľný program, %1 – obsahuje názov testovaného programu bez prípony)

Meno súbor	**var/dos/rslttst.exe**
Popis	Porovnáva výsledky testovaného programu s očakávanými výsledkami.
Použité parametre	pattern.def – žiadaný výstup programu (výstupná testovacia vzorka) screen.log – výstup testovaného programu (záznam obrazovky) rslt.log – názov súboru pre výsledok porovnávania.

7.6 Správa používateľov

Spravá používateľov slúži na administráciu používateľov ich pridávaniu, mazaniu, editovaniu údajov o používateľoch systému. Je tvorená z viacerých zdrojových súborov ako výpis, pridávanie a editovanie používateľov. Všetky zdrojové súbory využívajú spoločnú knižnicu funkcií pre prácu s databázou, resp. s údajmi prislúchajúcimi k modulu.

Verzia Dagwood tejto správy je založená na pôvodnom jadre systému Moodle verzie 1.1.1. Toto je doplnené o ďalšiu funkcionalitu (možnosť importovania používateľov zo súboru, podpora osobného čísla pre používateľov). Zdrojové súbory sú umiestnené v najmä adresári „user“ inštalácie systému Moodle, s výnimkou súborov funkcií určených pre administrátora (sú v adresári „admin“).

7.6.1 Popis formátu vstupného súboru pre import používateľov

Položky v riadku sú oddelené znakom „;“, formát riadku:

```
login;heslo;oscislo;meno;priezvisko;E-mail
```

- login – prihlasovacie meno, nesmie obsahovať biele znaky.
- heslo – prihlasovacie heslo, môže byť zapísané dvoma spôsobmi:
  - „x:HHHHHH“ - zašifrovaný zápis, HHHHHH predstavuje zašifrovanú hodnotu hesla md5 algoritmom.
  - „hhhhhh“ - nezašifrovaný, hhhhhh – heslo v textovej nezašifrovanej podobe.
- meno – krstné meno používateľa.
- priezvisko – priezvisko používateľa.
- E-mail – E-mailová adresa používateľa

Meno a priezvisko môžu obsahovať aj medzeru a prípadne ďalšie mená. Každý riadok obsahuje jeden záznam o používateľovi, ktorí sa má importovať do databázy. Komentár sa označuje na začiatku riadku znakom “#“, tento riadok sa spolu s prázdnymi riadkami ignorujú.

7.6.2 Modifikové časti správy používateľov

7.6.2.1 edit.html

Meno súboru	**edit.html**
Popis	Webová stránke pre formulár na editáciu používateľov. Zobrazuje formulár a jednotlivé jeho položky, povoľuje ich v závislosti na prihlásenom používateľovi.
Programovací jazyk	PHP + HTML
Využitie	Editácia používateľov.

Modifikácie:

- Pridaná položka pre editáciu/zobrazenie osobného čísla.

7.6.2.2 edit.php

Meno súboru	**edit.php**
Popis	Skript pripravuje a spracováva formulár pre editovanie používateľov. Testuje povinné položky a updatuje databázu používateľov.
Programovací jazyk	PHP
Využitie	Editácia používateľov

Modifikácie:

- Doplnenie logiky pre podporu a spracovanie osobných čísel.

7.6.2.3 import.php

Meno súboru	**import.php**
Popis	Skript pre importovanie používateľov so súboru, zobrazuje formulár pre zadanie súboru, spracováva načítaní súbor aj zobrazuje výsledky importovania používateľov (či boli úspešne pridaný do databázi)
Programovací jazyk	PHP
Využitie	Hromadné pridávanie nových používateľov, pridávanie používateľov zo súboru.

Modifikácie:

- Vytvorený skript.

7.6.2.4 user.php

Meno súboru	**admin/user.php**
Popis	Skript slúžiaci na prácu s používateľmi.
Programovací jazyk	PHP
Využitie	Administrátor pri pridávaní a editovaní používateľov

Modifikácie:

- Doplnenie spracovanie (presmerovanie) import používateľov.

7.6.3 Modifikácie v databáze

Názov tabuľky	**user**	
Popis	Tabuľka pre uchovávanie informácií o používateľoch	
Doplnené parametre	oscislo	Položka obsahujúca osobné číslo používateľa

7.6.4 Popis vnútorných algoritmov spracovania

Importovanie používateľov

Importovanie používateľov
1. Načítanie súboru. 1.1. Načítanie riadku. 1.2. Parsovanie riadku do jednotlivých premenných. 1.3. Kontrola premennej username (login) na výskyt bielych znakov. 1.4. Pridanie do databázy. 2. Výpis výsledku načítania

7.7 Odovzdávanie zadaní

Modul Assignment, resp. modul správy zadaní slúži na definovanie, príjem, prezeranie, a spracovávanie zadaní, pozostávajúcich z programu v niektorom z programovacích jazykov, alebo z iného súboru (PDF, DOC a pod).

Modul pozostáva z viacerých zdrojových súborov, reprezentujúcich jednotlivé časti modulu, ako zobrazovanie zadaní, odovzdávanie zadaní, kontrola zadaní a pod. Všetky zdrojové súbory využívajú spoločnú knižnicu funkcií pre prácu s databázou, resp. s údajmi prislúchajúcimi k modulu.

Verzia Dagwood tohto modulu je založená na pôvodnom jadre systému Moodle verzie 1.1.1. Táto je doplnená o ďalšiu funkcionalitu, ako možnosť odovzdávania viacerých verzií zadania a možnosť mazania neohodnotených zadaní. Najpodstatnejšou zmenou je doplnenie modulu o možnosť testovania funkčnosti a plagiátorstva odovzdaných zadaní. Testovanie je realizované prostredníctvom Dagwood démona a externých modulov.

Nasleduje popis zmien a rozšírení vo verzii Dagwood oproti verzii Moodle 1.1.1. a popis komunikačného protokolu s obslužným démonom pre testovanie zadaní. Všetky uvedené zdrojové súbory sú umiestnené v adresári „mod/assignment“ inštalácie systému Moodle, s výnimkou súborov, u ktorých je explicitne uvedená iná cesta.

7.7.1 Súčasti modulu

7.7.1.1 lib.php

Meno súboru	**lib.php**
Popis	Jedná sa o knižnicu funkcií pre modul Assignment, zabezpečujúci prístup k údajom v databáze, komunikáciu s démonom a ďalšie funkcie.
Programovací jazyk	PHP + HTML

Modifikované a doplnené funkcie:

Názov	**assignment_file_area_name_timestamp**
Popis	Funkcia vytvorí názov adresára pre verziu zadania odovzdaného včase $timestamp
Argumenty	$assignment, $user, $timestamp
Výstup	cesta k adresáru pre uloženie zadania

Názov	**assignment_file_area_timestamp**
Popis	Funkcia vytvorí adresár pre uloženie verzie zadania odovzdaného v čase $timestamp
Argumenty	$assignment, $user, $timestamp
Výstup	cesta k adresáru pre uloženie verzie zadania odovzdaného v čase $timestamp

Názov	**assignment_dir_area**
Popis	Funkcia vráti adresár zadania predmetu, do ktorého sa ukladajú odovzdávané zadania študentov
Argumenty	$assignment
Výstup	plná cesta k adresáru zadania predmetu

Názov	**assignment_dir_area_name**
Popis	Funkcia vráti adresár zadania predmetu, do ktorého sa ukladajú odovzdávané zadania študentov
Argumenty	$assignment, $user, $timestamp
Výstup	cesta k adresáru zadania predmetu relatívna vzhľadom dátový adresár Moodle

Názov	**assignment_get_submission**
Popis	Funkcia vráti záznam o vybranom odovzdanom zadaní študenta
Argumenty	$assignment, $user, $subid
Výstup	pokiaľ nie je zadané $subid, vráti najaktuálnejšiu verziu zadania, inak verziu požadovanú parametrom $subid

Názov	**assignment_get_submissions**
Popis	vráti všetky odovzdané zadania daného študenta a zadania predmetu
Argumenty	$assignment, $user
Výstup	všetkých odovzdaných verzií zadania daného študenta

Názov	**assignment_get_submission_last**
Popis	Funkcia vráti najaktuálnejšie odovzdané zadanie študenta
Argumenty	$assignment, $user
Výstup	objekt reprezentujúci posledne odovzdané zadanie

Názov	**assignment_get_submission_file_last**
Popis	vráti odkaz na súbor s poslednou verziou zadania
Argumenty	$assignment, $user
Výstup	odkaz pre WWW prístup k súboru s najaktuálnejšou verziou odovzdaného zadania

Názov	**assignment_get_submission_file**
Popis	Funkcia vráti odkaz na súbor s vybranou verziou zadania
Argumenty	$assignment, $user, $submission
Výstup	odkaz pre WWW prístup k súboru s vybranou verziou odovzdaného zadania

Názov	**assignment_get_submission_file_fullpath**
Popis	Funkcia vráti úplnú fyzickú cestu k súboru odovzdaného zadania
Argumenty	$assignment, $user, $submission
Výstup	úplná fyzická cesta k súboru daného odovzdaného zadania

Názov	**assignment_print_submission**
Popis	Modifikácia tejto funkcie vypíše informácie o danom odovzdanom zadaní, zároveň získava výsledky funkcionálneho testovania
Argumenty	$assignment, $user, $submission, $teachers, $grades, $testfunccheckhandler=""
Výstup	HTML výpis informácii o danom zadaní

Názov	**assignment_print_user_files_submission**
Popis	Funkcia vypíše zoznam súborov daného odovzdaného zadania (v Moodle je podpora zatiaľ len jedného súboru)
Argumenty	assignment, $user, $submission
Výstup	výpis zoznamu súborov daného zadania

Názov	**assignment_socket_open**
Popis	socketu pre komunikáciu s démonom
Argumenty	-
Výstup	Ukazovateľ na handle socketu

Názov	**assignment_socket_close**
Popis	Zatvorenie špecifikovaného socketu
Argumenty	$socket
Výstup	-

Názov	**assignment_functionality_test**
Popis	Spustenie testu funkcionality pre dané odovzdané zadanie
Argumenty	$assignment, $user, $submission
Výstup	Chybová správa, alebo výsledok testovania

Názov	**assignment_plagiarism_test_file**
Popis	Zaradenie súboru do testu plagiátorstva
Argumenty	$socket, $assignment, $user, $submission
Výstup	-

Názov	**assignment_plagiarism_test_start**
Popis	Spustenie testu plagiátorstva pre zaradené súbory
Argumenty	$socket, $assignment
Výstup	-

Názov	**assignment_plagiarism_test_result**
Popis	Získanie výsledku testu plagiátorstva
Argumenty	$socket, $assignment
Výstup	Chybová správa, alebo „ok“, ak bol výsledok získaný a uložený

Názov	**assignment_daemon_status**
Popis	Zistenie status démona a subdémonov pre kontrolu zadaní
Argumenty	-
Výstup	HTML text obsahujúci statusy démona a subdémonov

Názov	**assignment_print_daemon_status**
Popis	Výpis status démona a subdémonov pre kontrolu zadaní
Argumenty	-
Výstup	HTML výpis statusov démona a subdémonov

7.7.1.2 mod.html

Meno súboru	**mod.html**
Popis	Formulár pre vytváranie nového zadania v rámci predmetu, resp. pre zadávanie vlastností zadania
Programovací jazyk	HTML + PHP
Využitie	Vytváranie nového zadania, modifikácia vlastností existujúceho zadania predmetu

Modifikácie:

- Doplnenie parametra povoľujúceho možnosť odovzdávania viacerých verzií zadania.
- Doplnenie parametra povoľujúceho možnosť študentovi mazať odovzdané ešte neohodnotené verzie zadania.

7.7.1.3 submissions.php

Meno súboru	**submissions.php**
Popis	Skript zobrazujúci odovzdané zadania, obsluhujúci proces hodnotenia zadaní a ich kontroly.
Programovací jazyk	PHP + JavaScript
Využitie	Učiteľ – prezeranie odovzdaných zadaní, spúšťanie kontrol zadaní

Modifikácie:

- Doplnenie možnosti označenia zadaní, ktoré sa majú otestovať na plagiátorstvo, alebo funkčnosť (pridanie „checkboxov“).
- Doplnenie spúšťania testov plagiátorstva a funkčnosti pre vybrané zadania, a zároveň získavanie výsledkov pre ukončené testy funkčnosti.
- Doplnenie možnosti zobrazenia všetkých, alebo len posledných verzií odovzdaných zadaní študentov.
- Doplnené funkcie pre hromadné označovanie zadaní určených pre testovanie (JavaScript – funkcie *checkallfunc()* a *checkallplag()*).
- Doplnenie zadávania vstupnej a výstupnej vzorky pre test funkčnosti (HTML + JavaScript funkcia *showhidefileform()*).

7.7.1.4 upload.php

Meno súboru	**upload.php**
Popis	Skript pre prevzatie a uloženie odovzdaného zadania, vymazanie verzie odovzdaného zadania
Programovací jazyk	PHP
Využitie	Študent – odovzdanie, vymazanie zadania

Modifikácie:

- Doplnenie podpory viacerých verzií odovzdaného zadania.
- Vymazávanie vybraných ešte neohodnotených verzií zadaní.

7.7.1.5 view.php

Meno súboru	**view.php**
Popis	Skript pre prehliadanie študentom odovzdaných zadaní a ich hodnotení, pre odovzdanie, príp. vymazanie verzie zadania. Zobrazenie vykonaných testov plagiátorstva a statusu démona
Programovací jazyk	PHP + HTML
Využitie	Študent – odovzdanie, vymazanie zadania, prezeranie zadaní a ich hodnotení. Učiteľ – zobrazenie statusu démona a vykonaných testov plagiátorstva

Modifikácie:

- Doplnenie podpory viacerých verzií odovzdaného zadania.
- Vymazávanie vybraných ešte neohodnotených verzií zadaní.
- Doplnenie zobrazovania statusu démona.
- Doplnenie zoznamu vykonaných testov plagiátorstva

7.7.1.6 index.php

Meno súboru	**index.php**
Popis	Zobrazenie zoznamu zadaní daného predmetu, statusu démona
Programovací jazyk	PHP
Využitie	Študent, učiteľ – zobrazenie zoznamu zadaní predmetu. Učiteľ – zobrazenie statusu démona

Modifikácie:

- Doplnenie podpory viacerých verzií odovzdaného zadania.
- Vymazávanie vybraných ešte neohodnotených verzií zadaní.

7.7.1.7 moodle/config.php

Meno súboru	**moodle/config.php**	
Popis	Konfiguračný súbor moodle	
Programovací jazyk	PHP	
Využitie	Systém – vnútorné systémové nastavenia a premenné	
Doplnené parametre	$CFG->daemonhost	Hostiteľský počítač, na ktorom beží démon
	$CFG->daemonport	Port na hostiteľskom počítači, na ktorom beží démon

7.7.2 Komunikácia s démonom

Nasleduje popis komunikačného rozhrania medzi modulom zadaní a obslužným démonom zabezpečujúcim vykonanie kontroly plagiátorstva, resp. funkčnosti.

Funkcia: odovzdanie na testovanie

```
telnet localhost port
        klient> func_test <predmet> <user> <zadanie> <timestamp>
        daemon> store /var/spool/adresar/subor.asm
        klient> store_ok
```

Funkcia: zistenie stavu testovania

```
telnet localhost port
        klient> func_result <predmet> <user> <zadanie> <timestamp>
```

```
        daemon> result /var/spool/adresar/vysledok.txt
```
alebo
```
        daemon> none
```
alebo
```
        daemon> not found
```

Funkcia: testovanie plagiátorstva

```
telnet localhost port
        klient> compare <predmet> <user> <zadanie> <timestamp>
        daemon> store /cesta/subor.asm
        klient> compare ....
        daemon> store ...
        ....
        klient> store_ok <predmet> <zadanie>
```

Funkcia: zistenie stavu porovnávania

```
telnet localhost port
        klient> compare_result <predmet> <zadanie>
        daemon> none
```
alebo
```
        daemon> not found
```
alebo
```
        daemon> result /cesta/file.csv
```

Funkcia: stav modulov

```
telnet localhost port
        klient> status_plag
        daemon> ok
```
alebo
```
        daemon> down
        ...
        klient> status_func
        daemon> ok
```
alebo
```
        daemon> down
```

Výstupom z porovnávania zadaní je CSV súbor takejto štruktúry:

```
user1;timestamp
user2;timestamp
user3;timestamp
user4;timestamp
results
res(1 vs. 1);res(1 vs. 2);res(1 vs. 3);res(1 vs. 4)
res(2 vs. 1);res(2 vs. 2);res(2 vs. 3);res(2 vs. 4)
res(3 vs. 1);res(3 vs. 2);res(3 vs. 3);res(3 vs. 4)
```

```
res(4 vs. 1);res(4 vs. 2);res(4 vs. 3);res(4 vs. 4)
```

7.7.3 Modifikácie v databáze

Názov tabuľky	**assignment**	
Popis	Tabuľka pre uchovávanie zadaní predmetov a ich vlastností	
Doplnené parametre	Versioning	Príznak povolenej podpory viacerých verzií odovzdávaných zadania
	allow_func_test	Príznak povolenej možnosti pre testovanie funkčnosti študentom – zatiaľ nevyužité
	plag_test	Príznak signalizujúci prebiehajúci test plagiátorstva
	plag_test_time	Čas spustenie testu plagiátorstva
	allow_delete	Príznak povolenej možnosti vymazávania neohodnotených odovzdaných zadaní

Názov tabuľky	**assignment_submissions**	
Popis	Tabuľka pre uchovávanie odovzdaných zadaní študentov a ich vlastností	
Doplnené parametre	func_test	Príznak signalizujúci prebiehajúci test funkčnosti zadania
	func_test_result	Výsledok testovania funkčnosti daného zadania
	func_test_input	Vstupná vzorka pre testovanie funkčnosti daného odovzdaného zadania
	funct_test_output	Výstupná vzorka pre testovanie funkčnosti daného odovzdaného zadania

7.7.4 Popis vnútorných algoritmov spracovania

Podpora rôznych verzií zadania
3. Povolenie verzií pri definícií zadania predmetu 4. Odovzdávanie zadania 4.1. Kontrola povolenia verzií 4.2. Získanie cieľového adresára pre uloženie zadania 4.3. Uloženie nového zadania / nahradenie existujúceho zadania 5. Prehliadanie zadaní 5.1. Zobrazenie všetkých verzií zadania 5.2. Vymazanie neohodnoteného zadania

Kontrola funkčnosti
1. Výber zadaní pre kontrolu funkčnosti (Učiteľ) 2. Spustenie kontroly zadania 2.1. Nadviazanie komunikácie s démonom 2.2. Poskytnutie zadania na kontrolu 2.3. Skok na 2.1, kým sú v rade ďalšie zadania na kontrolu 3. Zobrazenie zadaní 3.1. Kontrola prebiehajúceho testu 3.2. Získanie výsledku (ak je 3.1 pozitívne) 3.3. Zobrazenie výsledku testu

Kontrola plagiátorstva
1. Výber zadaní pre kontrolu plagiátorstva (Učiteľ) 2. Spustenie kontroly plagiátorstva 2.1. Kontrola prebiehajúceho testu 2.2. Nadviazanie komunikácie s démonom (ak je 2.1 negatívne, inak koniec) 2.3. Poskytnutie zadania na kontrolu 2.4. Skok na 2.2, kým sú v rade ďalšie zadania na kontrolu 2.5. Spustenie kontroly a nastavenie príznaku kontroly 3. Zobrazenie realizovaných testov 3.1. Kontrola prebiehajúceho testu 3.2. Získanie výsledku (ak je 3.1 pozitívne) 3.3. Zobrazenie zoznamu realizovaných testov

7.7.5 Ďalšie možnosti úprav

Vzhľadom na nedostatok časových prostriedkov sme nezaradili do požiadaviek ďalšie možnosti, ktoré by mohol tento modul obsahovať. V tejto časti teda aspoň stručne načrtneme ďalšie možné úpravy:

- Vykonávanie viacerých testov plagiátorstva súčasne – vyžaduje doplnenie mechanizmu pre uchovávanie zoznamu bežiacich testov, a úpravu obslužného démona a komunikačného rozhrania medzi démonom a modulom zadaní.
- Možnosť testovania funkčnosti študentom – vytvoriť mechanizmus pre vykonávanie testov bez zápisu výsledkov, resp. mechanizmus pre nezávislé testovanie zadania.
- Zvýšiť konfigurovateľnosť modulov testovania – napr. možnosť definovať vzorky zdrojových kódov, ignorovaných pri teste plagiátorstva, a pod.

7.8 Démon

Dagwood Démon (ďalej DD) slúži na prepojenie webového rozhrania systému Dagwood –Moodle so subsystémami. Na jeho funkčnosť je potrebné, aby bežal na tom istom systéme, ako aj „web“ rozhranie systému (Moodle).

Hlavnou časťou DD je samotný zdrojový súbor, ktorý po spustení vykonáva požadovanú činnosť. Na spustenie je potrebný ešte konfiguračný súbor, ktorý rešpektuje potrebné nastavenia.

Konfigurácia DD je možná prostredníctvom konfiguračného súboru. Umiestnenie tohto súboru je napísané priamo v tele zdrojového súboru démona. Keďže je DD napísaný v jazyku Perl, po zmene umiestnenia súboru stačí iba jednoduchý reštart démona. Presný popis konfiguračných parametrov je v dokumentácii ku konfiguračnému súboru. V popise konfiguračného súboru je napísané aj štandardné umiestnenie jednotlivých adresárov využívaných na dočasné ukladanie výsledkov činnosti modulov.

Na jemnejšie nastavenie sieťových a výkonnostných vlastností DD je možné preštudovať dokumentáciu k Perlovému modulu Net::Server::Fork, ktorý je použitý pri implementácii DD.

Vstup DD je typicky z časti Moodle. Po pripojení sa klienta na TCP port démona, klient oznámi svoju požiadavku. Podľa komunikačného protokolu si dohodnú parametre požiadavky a ukončia spojenie. Počas spojenia je možné, že klient bude musieť odovzdať démonovi súbor so zadaním, alebo si prečíta výsledky predchádzajúcej požiadavky. Deje sa to cez súbory, ktoré určí démon.

7.8.1 Súčasti

7.8.1.1 dagwood-daemon.pl

Meno súboru	**dagwood-daemon.pl**
Popis	Tento súbor obsahuje samotný kód démona. Je naprogramovaný ako UNIXový daemon – jeho úlohou je bežať nepretržite.
Programovací jazyk	Perl
Spúšťanie	Pri spustení beží na popredí. Naspustenie je potrebné použiť niektorú z možností spustenia na pozadí (napr. nohup, daemontools atď.)
Prostredie	UNIX/Linux, pravdepodobne Windows
Požadované moduly	use Net::Server::Fork use SOAP::Lite

Funkcie:

Názov	**delete_file**
Popis	Funkcia vymaže súbory zadané ako argumenty v „taint“ prostredí (perl –T).
Argumenty	@filename
Návr. hodnota	-
Výstup	Žiadny

Názov	**do_compare**
Popis	Funkcia volá modul plagiátorstva cez SOAP rozhranie. Výsledok uloží na dohodnuté miesto.
Argumenty	$task
Návr. hodnota	-
Výstup	Výstup porovnávania, zmazanie dočasného súboru porovnávania

Názov	**do_compare_result**
Popis	Funkcia slúži na zistenie stavu porovnávania zadaní z časti Moodle.
Argumenty	$task
Návr. hodnota	-
Výstup	Podľa stavu porovnávania „not_yet“, „result menosuboru” alebo “not_found”.

Názov	**do_func_result**
Popis	Funkcia zistí stav testovania funkčnosti zadania pre požiadavky časti Moodle.
Argumenty	$task
Návr. hodnota	-
Výstup	Podľa stavu testovania „not_yet“, „result menosuboru” alebo “not_found”.

Názov	**do_func_test**
Popis	Funkcia volá modul testovania funkčnosti cez rozhranie SOAP.
Argumenty	$task
Návr. hodnota	-
Výstup	Výstup testovania, zmazanie dočasného súboru pre zistenie stavu.

Názov	**generate_filename**
Popis	Funkcia sa snaží vygenerovať jedinečné meno pre súbor.
Argumenty	-
Návr. hodnota	-
Výstup	Reťazec použiteľný v názve súboru pozostávajúci z časovej značky (timestamp), PID procesu a náhodného čísla.

Názov	**get_configuration**
Popis	Funkcia číta konfiguračný súbor a nastavuje globálne parametre.
Argumenty	$config_file_name
Návr. hodnota	-
Výstup	Globálne parametre – cesty k dočasným adresárom, URL modulov atď.

Názov	**logging**
Popis	Funkcia zabezpečujúca jednoduché logovanie v jednotnom formáte.
Argumenty	@log_messages
Návr. hodnota	-
Výstup	Funkcia zapíše všetky svoje parametre do definovaného súboru.

Názov	**prepare_compare**
Popis	Funkcia pripraví operáciu porovnávania zadaní komunikáciu s časťou Moodle.
Argumenty	@log_messages
Návr. hodnota	$predmet, $user, $zadanie, $timestamp, $line
Výstup	Parametre následného porovnávania zadaní.

Názov	**prepare_compare_result**
Popis	Funkcia pripraví argumenty pre zistenie stavu porovnávania zadaní.
Argumenty	$predmet, $zadanie
Návr. hodnota	$task
Výstup	Parametre pre zistenie stavu porovnávania zadaní.

Názov	**prepare_func_result**
Popis	Funkcia pripraví parametre na zistenie stavu testovania funkčnosti zadania.
Argumenty	$predmet, $user, $zadanie, $timestamp
Návr. hodnota	$task
Výstup	Parametre pre zistenie stavu testovania funkčnosti.

Názov	**prepare_func_test**
Popis	Pripravuje parametre pre testovanie testovanie funkčnosti zadania.
Argumenty	$predmet, $user, $zadanie, $timestamp
Návr. hodnota	$task
Výstup	Pripravené parametre na otestovanie funkčnosti zadania.

Názov	**prepare_status**
Popis	Funkcia zistí, či moduly testovania funkčnosti zadaní a porovnávania zadaní počúvajú na svojich portoch.
Argumenty	$line_from_socket
Návr. hodnota	$task
Výstup	Výstupom sú reťazce „OK“ alebo „down“ podľa skutočného stavu.

Názov	**process_request**
Popis	Funkcia číta socket zo siete a podľa prvého riadku volá príslušné prípravné funkcie (prepare_*).
Argumenty	-
Návr. hodnota	-
Výstup	Výstupom je vykonanie požadovanej funkcie.

Názov	**read_file**
Popis	Funkcia je určená na načítanie obsahu súboru do premennej.
Argumenty	$filename
Návr. hodnota	$data_from_file
Výstup	Údaje načítané zo súboru.

Názov	**read_line**
Popis	Funkcia prečíta jeden riadok zo socketu a znormalizuje ho.
Argumenty	$timeout
Návr. hodnota	$line
Výstup	Načítaný riadok s orezanými bielymi znakmi na začiatku a na konci.

7.8.1.2 dagwood-daemon.conf

Meno súboru	**dagwood-daemon.conf**	
Popis	Ide o konfiguračný súbor pre DD, cesta k nemu musí byť zapísaná priamo v súbore s DD.	
Parametre	port	Adresa a port, na ktorom démon počúva. Formát address:port
	CLIENT_TIMEOUT	Komunikačný timeout v sekundách
	PREFIX_INCOMING	Cesta ku vstupným súborom z časti Moodle
	PREFIX_PLAGIAT	Cesta k adresáru, kde sa ukladajú výsledky z porovnávania zadaní
	PREFIX_TF	Cesta k adresáru, kde sa ukladajú výsledky z testu funkčnosti

	PROXY_PLAGIAT	URL k modulu plagiátorstva
	PROXY_TF	URL k modulu testovania funkčnosti

7.8.1.3 dagwood-client-TF.pl

Meno súboru	**dagwood-client-TF.pl**
Popis	Jednoduchý skript, ktorý simuluje činnosť časti Moodle a testuje funkčnosť modulu testovania funkčnosti zadaní. Spúšťa sa z príkazového riadku s príslušnými parametrami. Parametre spojenia je potrebné napísať priamo do skriptu.
Programovací jazyk	Perl
Spúšťanie	dagwood-client-TF.pl <source_file> <input_pattern> <output_pattern>
Prostredie	UNIX/Linux, pravdepodobne Windows
Požadované moduly	IO::Socket

Funkcie:

Názov	**read_file**
Popis	Funkcia je určená na načítanie obsahu súboru do premennej.
Argumenty	$filename
Návr. hodnota	$data_from_file
Výstup	Údaje načítané zo súboru.

7.8.1.4 dagwood-client-plagiat.pl

Meno súboru	**dagwood-client-plagiat.pl**
Popis	Jednoduchý skript, ktorý simuluje činnosť časti Moodle a testuje funkčnosť modulu plagiátorstva. Spúšťa sa z príkazového riadku s príslušnými parametrami. Parametre spojenia je potrebné napísať priamo do skriptu.
Programovací jazyk	Perl
Spúšťanie	dagwood-client-plagiat.pl <input_file_1> <input_file_2> [<input_file_3> [...]]

Požadované moduly	IO::Socket

Funkcie:

Názov	**read_file**
Popis	Funkcia je určená na načítanie obsahu súboru do premennej.
Argumenty	$filename
Návr. hodnota	$data_from_file
Výstup	Údaje načítané zo súboru.

7.8.2 Komunikačný protokol

Komunikčný protokol medzi démonom a Moodle je opísaný v časti 7.7.2 Komunikácia s démonom.

Na komunikáciu démona s modulmi je využívaný protokol SOAP. Tento protokol sme vybrali hlavne kvôli jednoduchosti jeho používania a dostatočnej rozšírenosti. Vďaka implementácii pomocou modulu SOAP::Lite sme dosiahli to, že celá komunikácia s modulmi je ukrytá za volaniami dvoch funkcií. Prvou funkciou je funkcia *compare_this*, ktorá zabezpečí porovnanie súborov zadaní odovzdaných ako parametre funkcie. Druhou funkciou je *test_this*, ktorá zabezpečí otestovanie funkčnosti zdrojového súboru v module testovania funkčnosti.

8 Testovanie systému

Keďže sme stavali na už existujúcom systéme, implementácia prebiehala zdola-nahor a teda testovanie mohlo prebiehať priebežne pri každej zmene. Vďaka rozdeleniu systému na podsystémy a dobrému definovaniu rozhraní sme pri integrácii nemali žiadne zásadné problémy a nebolo treba vykonávať špecialne testovania na odhalenie príčiny pozorovanej chybovosti. V prípade nastania chyby, opravila sa iba v podsystéme, kde sa vyskytla a nemuseli sme vykonať revíziu aj v ostatných častiach.

Podsystémy vyhodnocovania zadaní (funkčnosť a plagiátorstvo) boli postavné na už existujúcich riešeniach, ktoré boli odskúšané a testované a preto neuvádzame podrobný záznam o ich testovaní. Testovanie častí systému prebiehalo ako „biela skrinka“ pričom testovanie systému ako celok vyzeralo ako „čierna skrinka“.

Modul správy zadaní je kľúčový v celom systéme, pretože predstavujej naplnenie jednej zo zásadných vlastností systému: „Funkcie podporujúce automatizované preverovanie vedomostí vrátane generovania a vyhodnocovania testov“. Predstavuje prepojenie bázy konceptov so subsystémami na vyhodnocovanie zadaní prostredníctvom nášho démona. Záznam o testovaní tejto dôležitej súčasti je opísaný nižšie.

8.1 Testovanie modulu správy zadaní

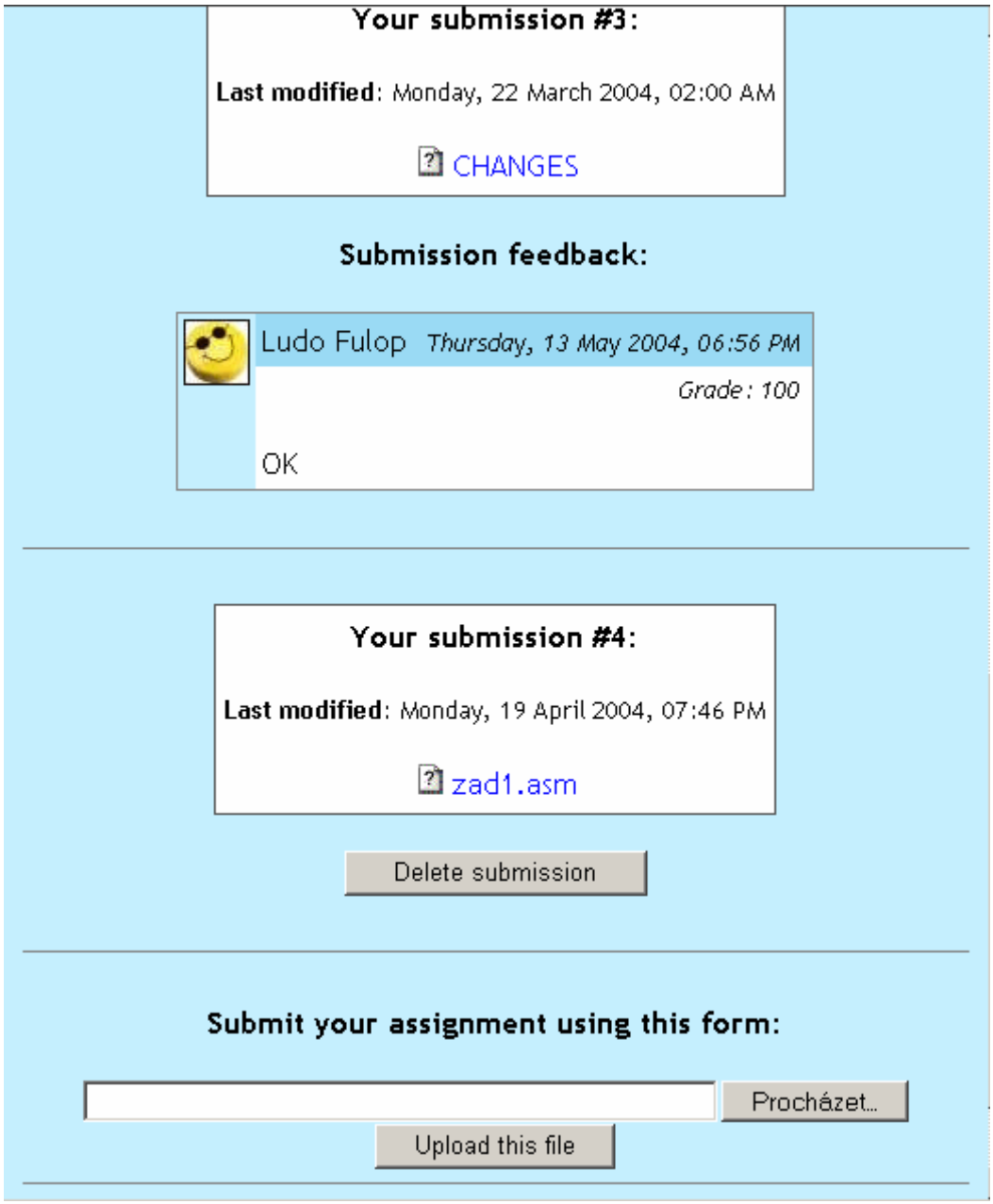
Testovanie doplnenej funkcionality a modifikácií prebiehal v troch krokoch, a to testovanie podpory verzií, testovanie kontroly plagiátorstva a testovanie kontroly funkčnosti.

Testovanie prebiehalo priebežne počas implementácie jednotlivých častí modulu, a taktiež bola funkcionalita testovaná aj po dokončení všetkých úprav.

8.1.1 Testovanie podpory verzií

Na Obr. 17 sa nachádza ukážka priebehu testovania podpory viacerých verzií zadaní. Pri testovaní bolo nutné overiť nasledovnú funkčnosť:

1. vplyv povolenia, resp. zakázania podpory viacerých verzií v nastaveniach zadania predmetu,
2. vplyv povolenia, resp. zakázania podpory vymazania ešte neohodnoteného zadania,
3. samotné odovzdávanie zadaní a ich zobrazovanie študentovi a učiteľovi.



Obr. 17: Testovanie podpory viacerých verzií zadaní.

8.1.1.1 Vplyv povolenia, resp. zakázania podpory viacerých verzií v nastaveniach zadania predmetu

- Testovanie spočívalo v porovnaní funkcionality pri povolenej podpore odovzdávania zadaní a pri zakázanej podporu odovzdávania zadaní.
- Funkcionalita sa podľa predpokladu líšila v závislosti od daného nastavenia, t.j. študentovi bolo, resp. nebolo povolené odovzdávanie viacerých verzií zadania.

- Testovanie prebehlo úspešne a neboli zistené žiadne chyby.

8.1.1.2 Vplyv povolenia, resp. zakázania podpory vymazania ešte neohodnoteného zadania

- Testovanie spočívalo v porovnaní funkcionality pri povolenej podpore vymazávania ešte neohodnotených zadaní.
- Funkcionalita sa podľa predpokladu líšila v závislosti od daného nastavenia, t.j. študentovi bolo, resp. nebolo povolené vymazávanie ešte neohodnotených verzií zadania.
- Testovanie prebehlo úspešne a neboli zistené žiadne chyby.

8.1.1.3 Samotné odovzdávanie zadaní a ich zobrazovanie študentovi a učiteľovi

- Testovanie spočívalo v niekoľkých pokusoch o odoslanie súboru (odovzdanie zadania) a sledovania reakcií systému. Testovanie prebiehalo pri povolenej podpore verzií a mazania zadaní.
- Systém sa správal podľa predpokladov, uchovávané boli všetky odovzdané súbory (verzie zadania). Taktiež bol zachovaný prístup k zoznamu verzií zadaní a aj možnosť zobraziť ľubovolnú verziu zadania.
- Testovanie prebehlo úspešne a neboli zistené žiadne chyby.

8.1.2 Testovanie kontroly plagiátorstva

Pri testovaní bolo nutné overiť nasledovnú funkčnosť:

1. zaradenie zadaní do testu a spustenie testu,
2. sledovanie priebežného stavu a výsledkov testovania.

8.1.2.1 Zaradenie zadaní do testu a spustenie testu

- Testovanie spočívalo vo výbere zadaní zo zoznamu odovzdaných zadaní, a v následnom potvrdení testovania. Po takomto spustení testovania sa sledovala odozva systému.
- Po spustení testovania bol automaticky zablokovaný výber zadaní na zaradenie do testovania plagiátorstva.

- Testovanie tohto kroku prebehlo úspešne na odlišných vzorkách a neboli zistené žiadne chyby.

8.1.2.2 Sledovanie priebežného stavu a výsledkov testovania

- Po spustení testovania podľa kroku 1. sa sledovala odozva systému v rozličných časoch. Po ukončení testu sa výsledky testu kontrolovali s odhadovanými hodnotami (napr. podobnosť rovnakých zadaní, podobnosť zadaní, ktoré sme mierne upravili, podobnosť rozličných zadaní).
- Po spustení testu v zozname realizovaných testov pribudol záznam o práve prebiehajúcom teste. Neskôr, po ukončení testu, sa tento záznam odkazoval na výsledok tohto testu. Výsledky testu boli kontrolované podľa odhadovaných výsledkov. Zároveň bol na viacerých systémoch testovaný export výsledkov do XLS formátu.
- Testovanie prebehlo úspešne na odlišných vzorkách a neboli zistené žiadne chyby. Problémy vznikali len pri exporte do XLS formátu v prípade, že na klientskej stanici nebola s týmto formátom asociovaná žiadna aplikácia (čo je problém lokálnej stanice, nie systému).

Obr. 18: Výsledky testovaní plagiátorstva.

Výsledky testovaní plagiátorstva možno vidieť na Obr. 18.

8.1.3 Testovanie kontroly funkčnosti

Pri testovaní bolo nutné overiť nasledovnú funkčnosť:

1. zaradenie zadaní do testu a spustenie testu,
2. sledovanie priebežného stavu a výsledkov testovania.

8.1.3.1 Zaradenie zadaní do testu a spustenie testu

- Testovanie spočívalo vo výbere zadaní zo zoznamu odovzdaných zadaní, výbere vstupných a výstupných vzoriek, a v následnom potvrdení testovania. Po takomto spustení testovania sa sledovala odozva systému.
- Po spustení testovania bol automaticky zablokovaný výber zadaní na zaradenie do testovania funkčnosti (zadaní, na ktorých prebiehalo testovanie).
- Testovanie tohto kroku prebehlo úspešne na odlišných vzorkách a neboli zistené žiadne chyby.

8.1.3.2 Sledovanie priebežného stavu a výsledkov testovania

- Po spustení testovania podľa kroku 1. sa sledovala odozva systému v rozličných časoch. Po ukončení testovania sa prezeral výsledok testu podľa výsledkov získaných z testovania v klasickom prostredí (kompilácia, ručné zadávanie vstupov a sledovanie výstupu zadania).
- Po spustení testu bola zablokovaná možnosť opätovne spustiť test (až do doby kým test nebol ukončený). V rôznych časových momentoch sa postupne zobrazovali výsledky testu podľa poradia v akom boli zadania testované systémom.
- Testovanie prebehlo úspešne na odlišných vzorkách a neboli zistené žiadne chyby.

Výsledky testovania funkčnosti možno vidieť na Obr. 19.

Obr. 19: Testovanie funkčnosti.

8.1.4 Záver

V terminológii softvérového inžinierstva možno záverečné testovanie považovať za neúspešné, keďže neodhalilo žiadne chyby v systéme. Všetky chyby boli teda odhalené počas testovania jednotlivých častí modulu, resp. systému priamo v čase implementácie.

9 Čo sme sa naučili a čo nestihli

9.1 Čo sme sa naučili

Odhliadnuc od riadenia a manažovania projektu a prihliadnuc na praktické skúsenosti s novými technológiami, sa náš tím naučil a zlepšil vo využívaní technológií založených na jazyku XML a najmä SOAP protokol. V prvotných fázach projektu šlo aj napr. o DocBook a SAML. Tak isto sme solídne množstvo skúseností nadobudli pri implementácii vo vnútri Moodle v jazyku PHP, kde sme celkovo odpracovali približne 80 hodín čistého programovania. Precvičili sme si aj skriptovacie jazyky „shell-u“ vrátane Perl. Keďže niektoré podsystémy boli už pred nami naimplementované v jazyku C, tak sme oprášili staré skúsenosti z programovania podľa ANSI štandartu jazyka C.

9.2 Čo sme nestihli

Čo sme nestihli vzhľadom na zmenenú špecifikáciu je zachytené v podkapitolách s názvom „Ďalšie vylepšienia“ v kapitole 7.

10 Používateľská príručka

Systém Dagwood slúži na riadenie výučby predmetov a komunikáciu medzi učiteľom a študentami (informačnú, testovaciu). Systém bol vyvinutý najmä pre jeho použitie v predmete Systémové programovanie a asemblery ale môže sa použiť na riadenie ľubovoľného predmetu bez závislosti na forme štúdia a type predmetu. Celý systém z fyzického pohľadu pozostáva z týchto subsystémov:

- pavučinové rozhranie (postavené na systéme Moodle v1.1.1 a jeho modifikácií),
- subsystém Testovanie zadaní (TZ),
- subsystém Zisťovanie plagiátorstva (ZP),
- subsystém Odovzdávanie zadaní (OZ).

Používateľ (okrem administrátora celého systému) sa stretne iba s pavučinovým rozhraním. V ďalšom texte je opísanie používanie systému. Niektoré časti (najmä tie, ktoré sme nemodifikovali) vychádzajú aj z originálu príručky Moodle, ktorá je dostupná každému používateľovi po prihlásení sa do systému. Vo všeobecnosti platí pravidlo, že pri používaní sa nebojte ničoho a kľudne sa pohybujte po systéme aj vrámci učenia sa ho používať.

10.1 Základné pojmy a používanie rozhrania Dagwood

Blok –zobrazovacia časť systému na zobrazenie rôznych menu.

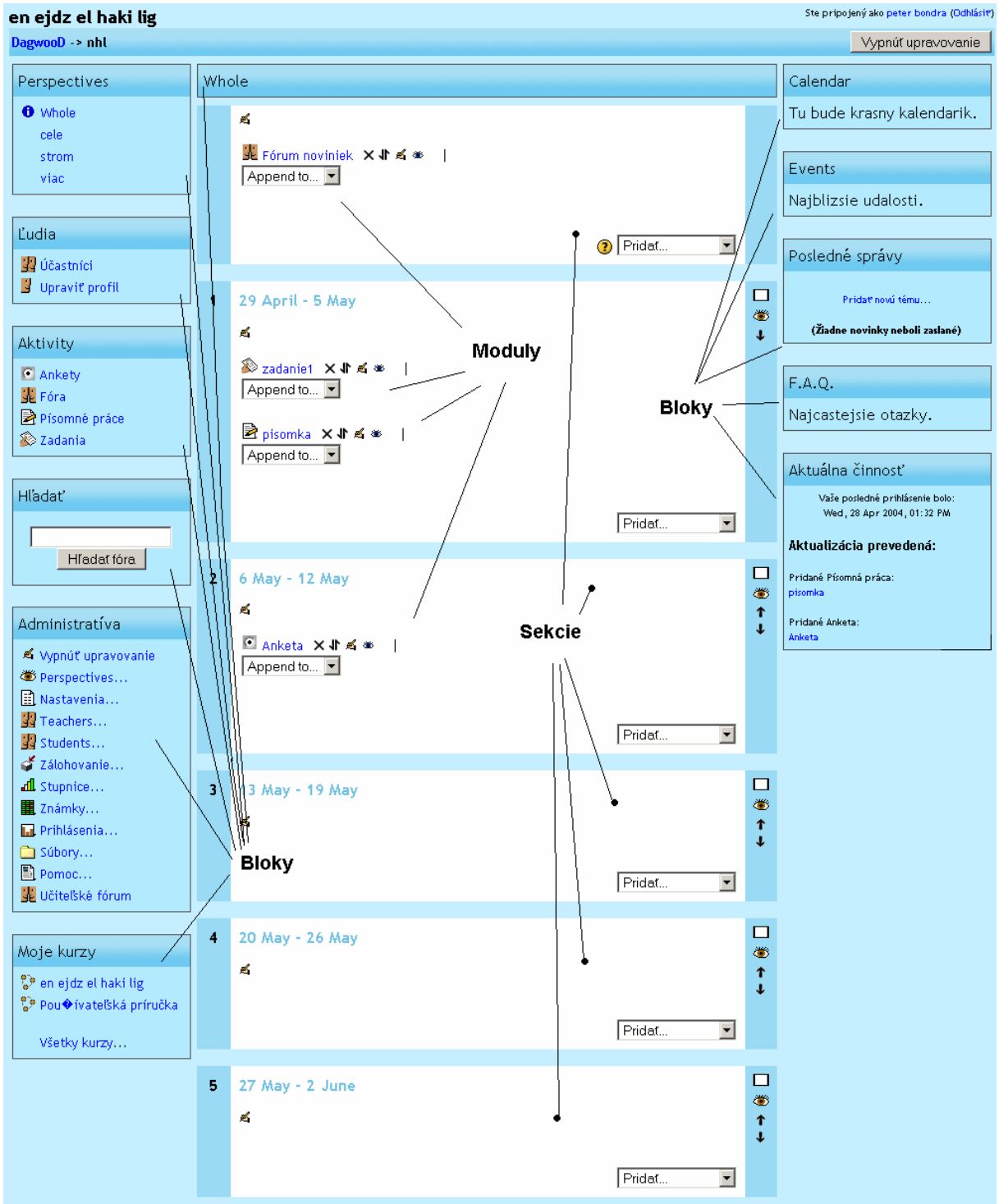
Sekcia –kurz sa skladá zo sekcií, ktoré môžu byť buď týždne, alebo témy (záleží od nastavenia kurzu - či je základnou jednotkou kurzu týždeň, alebo téma). Sekcie sú zobrazované na hlavnej stránke kurzu v strede obrazovky.

Modul –je základom bázy konceptov. Každý modul má svoju špecifickú funkcionalitu a účel. V systéme sa nachádzajú napr. tieto moduly: zdroj, zadanie, diskusia, tvorivá dieľna, dotazník a pod.

Pohľad –je filtrované zobrazenie bázy koncpetov v predmete. Filtrovať znamená zobraziť iba určité moduly. Takto je možné zoskupiť určité moduly do logick´zch celkov –pohľadov. Samotná prezentácia pohľadu závisí od jej typu. Možné typy pohľadov:

- mono – zobrazí sa iba vybrané moduly zo sekcie 0.
- multi –zobrazia sa vybrané moduly zo sekcií 1 až n (n –počet týždňov kurzu).
- celok –zobrazia sa všetky týždne/témy kurzu + horná spoločná sekcia (sekcia 0). Je to kombinácia typu mono a multi.
- strom –vybrané moduly sa zobrazia vo forme stromu.

Názorné vysvetlenie pojmov je zachytené aj na obr. Obrázok 1.



Obrázok 1: Základná stránka kurzu a vysvetlenie pojmov.

Zobrazenie jednotlivých sekcií v perspektívach typu *celok* a *multi* je možné dodatočne upraviť pomocou menu nachádzajúceho sa pri ľavom okraji každej sekcie (s výnimkou hornej spoločnej sekcie) obr. Obrázok 2.

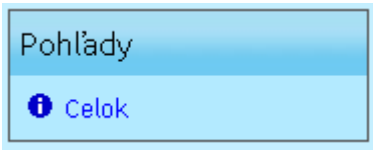
Obrázok 2: Menu na ovládanie zobrazenia sekcie.

 -zobrazenide všetkých sekcií/jednej sekcie.

 - zobrazenie alebo skrytie tejto sekcie pred študentmi.

 - presnunie danú sekciu o jednu sekciu hore/dole.

Každý kurz má vždy aspoň pohľad s menom *Celok,* typu *celok* (obr.Obrázok 3). Je to základný a úplný pohľad na bázu konceptov. Z tohto pohľadu sa potom priraďujú moduly do pohľadov, zatiaľ čo v iných pohľadoch sa moduly iba uberajú.



Obrázok 3: Menu s implicitným pohľadom

10.2 Kategórie používateľov

Používatelia, resp. ich právomoci sú rozdelené do nasledovných kategórií:

- administrátor - plný prístup ku všetkým častiam systému.
- tvorca kurzov - možnosť vytvárania nových kurzov, v rámci svojich vytvorených kurzov má plné právomoci prístupu (t. j. je automaticky evidovaný v týchto kurzoch ako učiteľ).
- učiteľ - má plné právomoci vo svojom kurze (t. j. vytvárať nové témy v diskusnom fóre, vytvárať zadania pre študentov, hodnotiť výsledky študentov atď.), nemá však možnosť sám vytvoriť nový kurz.
- študent - v rámci daného kurzu má právomoci prezerania si údajov na fóre noviniek pre študentov, odpovedania na príspevky a odovzdávania zadaní do termínu stanoveného učiteľom.

10.2.1 Administrátor

Administrátor má plné právomoci prístupu k systému, v tejto časti je popísaná administratíva tých častí systému, ku ktorým má výhradný prístup. Blok administratíva sa skladá z nasledujúcich častí:

- konfigurácia (nastavenie jazyka, vzhľadu okien atď),
- užívatelia (administratíva používateľských účtov, zadeľovanie používateľov do kategórií),
- kurzy (administratíva kurzov),
- prihlásenia (sledovanie prihlásení používateľov),
- miesto so súbormi (administratíva uploadovania súborov),
- administrátor (zobrazí vymenované položky bloku administrátor spolu s ich položkami na celú obrazovku).

10.2.1.1 Rozloženie prvkov na obrazovke

Obrázok 4: Stránka po prihlásení administrátorova

V **strednej časti** obrazovky sa vypisujú príspevky do diskusných fór, prípadne ak si administrátor prezerá stránky niektorého kurzu, obsah zvolených položiek tohto kurzu.

V **pravej časti** sú vypisované dôležité upozornenia ako oznamy o nových príspevkoch v diskusných fórach, zmeny od posledného prihlásenia atď.

V **ľavej časti** obrazovky nachádzajú bloky, pomocou ktorých je možné prezeranie a administrácia systému Dagwood. Tieto bloky sú členené nasledovne:

Pohľady

Administrátor si muože zvoliť pohľad (náhľad) na bázu konceptov v predmete. Vždy je dostupný pohľad *Všetko*. Manažovanie pohľadov (vytváranie, mazanie, ich typ a obsah) je povolený len pre učiteľov daného kurzu.

Miestne správy

Pomocou miestnych správ je možné vytvárať oznamy, resp. diskusné fóra, ktoré slúžia na informovanie používateľov.

Kurzy

Táto časť slúži na informovanie o činnosti prebiehajúcich kurzoch a na správu kurzov.

Administratíva

Konfigurácia

Slúži na definovanie premenných rozhrania Dagwood, prípadne na nastavenie jeho základného výzoru:

- nastavte premenné - umožňuje nastavenie premenných, ktoré ovplyvnia základné operácie rozhrania Dagwood,
- miestne nastavenia - umožňuje definovanie výzoru úvodná stránka,
- témy - umožňuje výber výzoru stránky (farby, písma, atď.),
- jazyk - umožňuje kontrola a úprava jazykovej verzie,
- administrácia modulov - umožňuje usporiadanie nainštalovaných modulov a ich nastavenia,
- filters - umožňuje výber textových filtrov a ich nastavenie,
- zálohovanie - umožňuje konfiguráciu automatického zálohovania a jeho pravidelného spúšťania.

Užívatelia

Umožnuje administrátorovi vytváranie/rušenie používateľských kônt a prideľovanie právomocí používateľom:

- overovanie - nastavenie spôsobu overovania nových používateľských kônt,
- pridať nového užívateľa - manuálne vytvorenie nového používateľského konta,
- upload users - importovanie nových používateľských kônt z textového súboru,
- upraviť užívateľské kontá - prezeranie zoznamu používateľských kônt a ich upravovanie,
- prideliť administrátorov - umožňuje pridelenie administrátorkých právomocí vybraným používateľom,
- prideliť tvorcov kurzov - určenie, ktorí používatelia majú právo vytvárať nové kurzy,

- prideliť učiteľov - pridelenie učiteľov do kurzov,
- zapísať študentov - umožňuje administrátorovi zápis študentov do kurzov.

Kurzy

Slúži na vytváranie/rušenie kurzov a taktiež zadeľovanie kurzov do tématických skupín (jeho právomoci sú takmer totožné v tomto ohľade s právomocami *tvorca kurzov* s tým rozdielom, že zmazať existujúci kurz môže len administrátor).

Prihlásenia

Zobrazí prihlásenia používateľov do rozhrania Dagwood podľa zadaných kritérií (kurzy, používatelia, dátum, vykonané aktivity v systéme).

Miesto so súbormi

Dovoľuje administrátorovi vytváranie stromovej štukturúry v adresári určenom na odkladanie nahraných súborov a taktiež správu uploadnutých súborov.

Administrátor...

Obsahuje súhrnne všetky vyššie popísané položky bloku Administratíva.

10.2.2 Tvorca kurzov

Používateľ s právomocami *tvorca kurzov* je vlastne administrátorom kurzov. Môže teda vykonávať takmer všetky činnosti popísané v časti *Administrátor*, t. j. vytvárať kurzy a taktiež zaraďovať kurzy do tématických skupín, ale nemôže mazať už existujúce kurzy. Vo vytvorenom kurze je tvorca tohto kurzu automaticky priradený ako učiteľ kurzu, t.j. má prístup k všetkým nastaveniam kurzu. Ostatné činnosti, ktoré sa týkajú učiteľa kurzu sú popísané v časti 10.2.3 Učiteľ.

10.2.2.1 Rozloženie prvkov na obrazovke

V **strednej časti** obrazovky sa vypisujú príspevky do diskusných fór, prípadne ak si tvorca kurzov prezerá stránky niektorého kurzu, obsah zvolených položiek tohto kurzu.

V **pravej časti** sú vypisované dôležité upozornenia ako oznamy o nových príspevkoch v diskusných fórach, do ktorých sa zapojil, zmeny od posledného prihlásenia atď.

V **ľavej časti** obrazovky nachádza niekoľko blokov, pomocou ktorých je možné prezeranie, prípadne administrácia častí systému Moodle, ku ktorým má tvorca kurzov prístup. Tieto bloky sú členené nasledovne:

Pohľady

Používateľ si muože zvoliť pohľad (náhľad) na bázu konceptov v predmete. Vždy je dostupný pohľad *Všetko*. Tvorca kurzov má povolené manažovanie (vytváranie, mazanie, ich typ a obsah) len v kurzoch, v ktorých je súčasne i učiteľom.

Miestne správy

Pomocou miestnych správ je možné zapojiť sa do prebiehajúcich diskusií.

Kurzy

Táto časť slúži na informovanie o práci prebiehajúcich kurzoch, prípadne na správu kurzov, v závislosti od toho či je tvorca kurzov súčasne i učiteľom daného kurzu.

Administratíva

Podobne ako v prípade modulu Kurzy je obsah tohto modulu závislý od toho, či je tvorca kurzov súčasne i učiteľom daného kurzu. Ak áno, platia pre neho tie isté pravidlá ako pre učiteľa.

10.2.3 Učiteľ

V ponímaní Dagwood je učiteľ administrátorom svojho kurzu (kurzov), t. j. môže meniť nastavenia kurzu, pridávať/rušiť učiteľov a študentov, zálohovať a obnovovať údaje kurzu. Takisto môže sledovať aktivity študentov (prihlásenia, s čím pracovali v rámci kurzu, kedy odovzdávali zadania a pod). Učiteľia majú v rámci kurzu k dispozícii vlastné učiteľské fórum, ktoré nie je prístupné študentom.

10.2.3.1 Rozloženie prvkov na obrazovke

V **strednej časti** obrazovky sa vypisujú príspevky do diskusných fór, prípadne ak si učiteľ prezerá stránky niektorého kurzu, obsah zvolených položiek tohto kurzu.

V **pravej časti** sú vypisované dôležité upozornenia ako oznamy o nových príspevkoch v diskusných fórach, do ktorých sa zapojil, zmeny od posledného prihlásenia atď.

V **ľavej časti** obrazovky nachádza niekoľko blokov, pomocou ktorých je možné prezeranie, prípadne administrácia častí systému Moodle, ku ktorým má učiteľ prístup. Tieto bloky sú členené nasledovne:

Pohľady

Používateľ si muože zvoliť pohľad (náhľad) na bázu konceptov v predmete. Vždy je dostupný pohľad *Všetko*. Manažovanie pohľadov (vytváranie, mazanie, ich typ a obsah) je povolené v plnom rozsahu pre kurzy, v ktorých je učiteľom.

Miestne správy

Pomocou miestnych správ je možné zapojiť sa do prebiehajúcich diskusií.

Kurzy

Táto časť slúži na informovanie o práci prebiehajúcich kurzoch a na správu svojich kurzov.

Administratíva

Položky administratívy učiteľa sú nasledovné:

- Zapnúť/vypnúť upravovanie - zobrazí/skryje nástroje na upravovanie priamo v jednotlivých položkách fóra (na editovanie niektorých vlastností kurzu ju nutné mať upravovanie zapnuté).
- Perspectives -slúží na manažovanie pohľadov pre daný predmet (vytváranie, mazanie, nastavovanie viditeľnosti).
- Nastavenia -obsahuje všeobecné nastavenia kurzu (ktoré sa definovali pri jeho vytváraní), t. j. prístup používateľov ku kurzu, informácie, ktoré sa zobrazia študentom atď.
- Učitelia -v tomto menu je možné pridávať/rušiť učiteľov daného kurzu.
- Študenti -toto menu slúži na manuálne prihlasovanie študentov do kurzu (je možné povoliť i prihlasovanie študentami).
- Zálohovanie -slúži na vytvorenie zálohy kurzu, pričom je možné presne špecifikovať, ktoré údaje a v akom rozsahu sa majú zálohovať.
- Obnoviť zo zálohy -pomocou tohto menu je možné obnoviť kurz zo zálohy.
- Stupnice -umožňuje učiteľovi definovať stupnicu pomocou, ktorej budú hodnotení študenti, súčasťou môže byť i slovný popis hodnotenia.
- Známky -v tomto menu je možné stiahnuť hodnotenia študentov v danom kurze vo formáte txt alebo MS Excel.
- Prihlásenia -tu si môže učiteľ prezrieť činnosti študentov za vybrané časové obdobie, prípadne vo vybranej časti kurzu.
- Súbory -umožňuje definovať adresárovú štuktúru, do ktorej sú ukladané nahrané a záložné súbory.
- Pomoc -stránka s vymenovaním a popisom možností práce učiteľa s Dagwood (odporúčame podrobne preštudovať).
- Učiteľské fórum -fórum slúžiace na komunikáciu medzi učiteľmi, študentom sa nezobrazuje.

Moodle poskytuje viacero preddefinovaných modulov:

- Anketa -zisťovanie informácií, ktoré neslužia na hodnotenie študentov, ale na zistenie názorov na danú problematiku ale aj na kvalitu predmetu.
- Chat -online komunikácia medzi práve prihlásenými učiteľmi/študentmi daného kurzu.
- Fórum -diskusné fórum, do ktorého môžu prispievať učitelia i študenti, zväčša vytvorené za cieľom diskusie o užšom probléme.
- Glossary -slovník glos na danú tému.
- Label -pridá na nástenku danej sekcie krátky oznam.
- Lesson -umožňuje vytvárať lekcie, na konci ktorých je možné preskúšanie formou testu.
- Písomná práca -dovolí učiteľovi zadať tému písomnej práce (článku).
- Test -vytvorenie testu, ktorý bude automaticky hodnotený podľa určených kritérií.
- Tvorivá dielňa -slúži na hodnotenie tvorivej činnosti študentov (bližšie ju špecifikuje učiteľ).
- Voľba -slúži na zadanie otázky pre študentov, kedy sa ako ich odpoveď očakáva výber jednej z možností.
- Zadanie -zadanie úlohy na vypracovanie pre študentov, pričom vypracovanie sa odovzdáva v stanovenej forme určeným spôsobom (napr. nahranie na server), pričom je možné špecifikovať, či študent môže odovzdať viac verzií zadaní a či po odovzdaní môže študent svoje zadanie zmazať (ak už nebolo ohodnotené učiteľom).
- Zdroj -vkladanie informácií z externého zdroja - odkaz na web-stránku, externý program atď.

Každý modul sa dá priradiť do pohľadu, t. j. v ktorých sa má zobrazovať. Blok *Aktivity*, ktorý sa nachádza v ľavej časti obrazovky obsahuje súhrn týchto modulov, ktoré sa v kurze používajú zoradené podľa druhu. Tento blok sa ale zobrazí len v prípade, že aspoň jeden z modulov už bol zadaný.

10.2.4 Študent

Z pohľadu študenta slúži Dagwood jednak na komunikáciu s učiteľom (napr. pomocou diskusného fóra - kam učiteľ zadáva oznamy a študenti môžu na ne i prípadne reagovať) a taktiež aj ako prostriedok pomocou ktorého je možné odovzdávať vypracované zadania a po ich ohodnotení učiteľom i prezrieť si svoje hodnotenie.

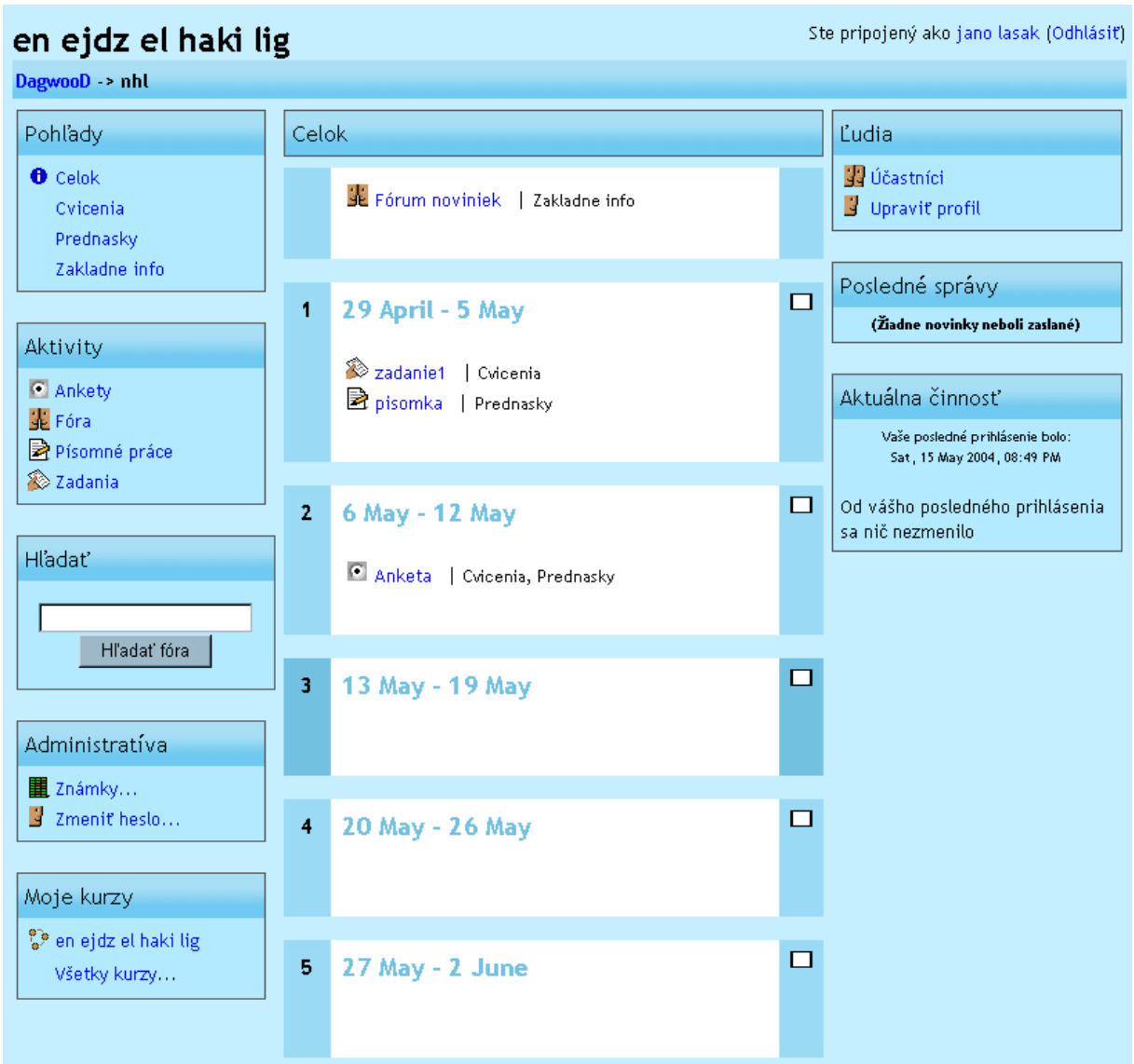
10.2.4.1 Rozloženie prvkov na obrazovke

V **strednej časti** obrazovky sa vypisujú príspevky do diskusných fór, prípadne ak si študent prezerá stránky niektorého kurzu (ktorého je študentom, alebo má kurz voľný prístup), obsah zvolených položiek tohto kurzu.

V **pravej časti** sú vypisované dôležité upozornenia ako oznamy o nových príspevkoch v diskusných fórach, do ktorých sa zapojil, zmeny od posledného prihlásenia atď.

V **ľavej časti** obrazovky nachádza niekoľko blokov, pomocou ktorých je možné prezeranie, prípadne administrácia svojich nastavení v kurzoch, ktorých je študentom. Tieto bloky sú členené nasledovne:

Zobrazenie kurzu študentovi je na obr. Obrázok 5.



Obrázok 5: Zobrazenie kurzu študentovi.

Pohľady

Používateľ si muože zvoliť pohľad (náhľad) na bázu konceptov v predmete z predefinovaných pohľadov. Vždy je dostupný pohľad *Celok*.

Miestne správy

Pomocou miestnych správ je možné zapojiť sa do prebiehajúcich diskusií.

Moje kurzy

Zoznam kurzov, do ktorých je študent prihlásený.

Administratíva

Blok administratíva umožnuje študentovi prezrieť si svoje hodnotenie za vypracované úlohy (položka *Známky*), prípadne upraviť svoj používateľský profil. Vypracovanie už odovzdaných úloh a ich hodnotenie zoradené podľa druhu úlohy si môže študent prezrieť v ľavej časti obrazovky v bloku *Aktivity*. Podobne ako u učiteľa sa toto blok zobrazí len v prípade, že už bola niektorá z úloh zadaná.

10.2.5 Hosť

Právomoci hosťa sú na úrovni prezerania si miestnych správ, informácií v kurzoch, v ktorých je prístup hosťom povolený. Zobrazené sú však len všeobecné informácie o kurze, k informáciam o používateľoch nemá hosť prístup.

10.3 Vybrané scenáre použitia

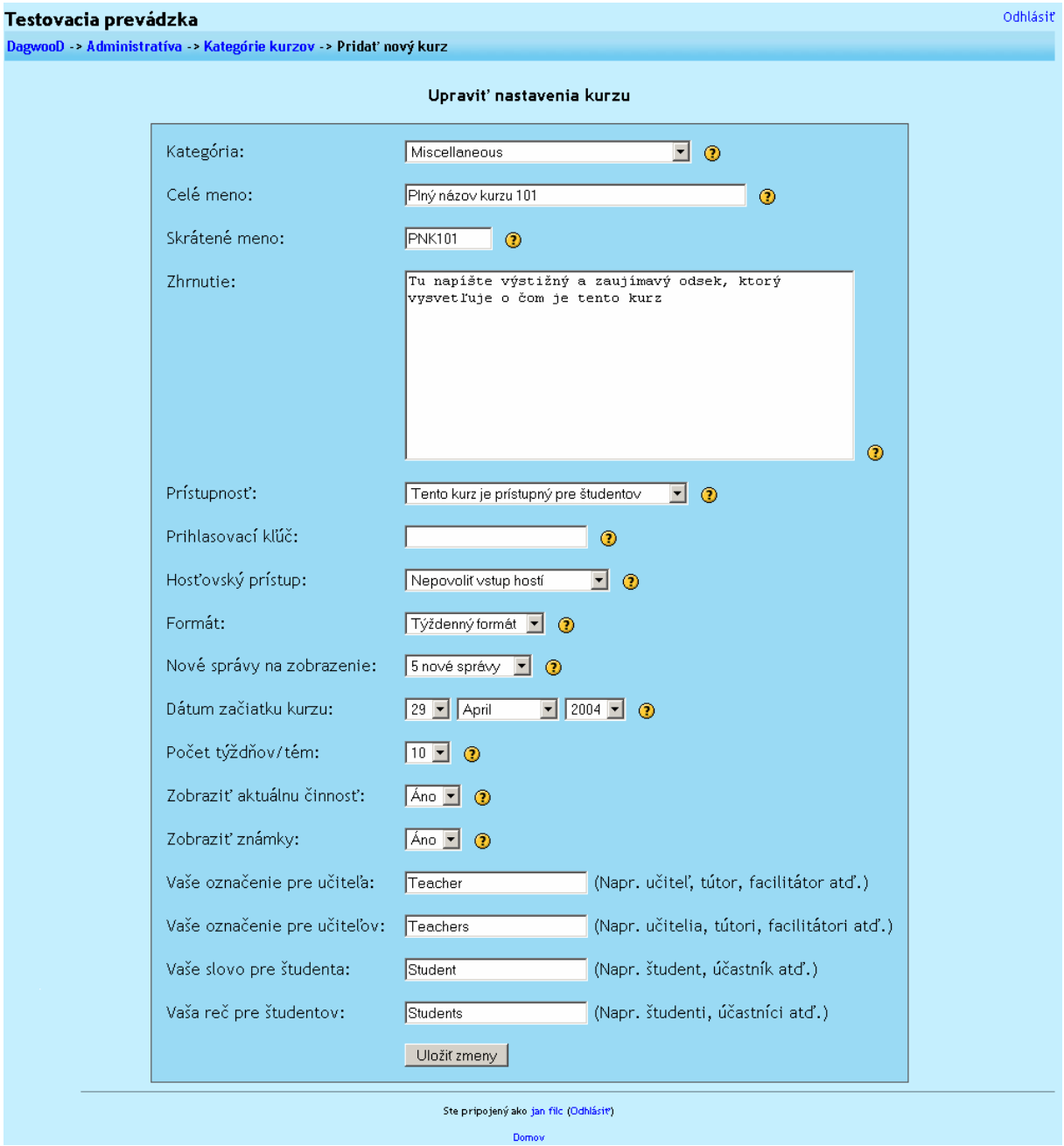
10.3.1 Vytvorenie nového kurzu

1. Po prihlásení kliknite na položku *Kurzy* v v bloku *Administratíva*. Zobrazí sa blok (Obrázok 6)

Obrázok 6.

umožňujúce pridanie nového kurzu, vyhľadanie už existujúceho kurzu a zobrazenie už existujúcich kurzov po kategóriach kde sú zaradené. Novú kategóriu kurzov alebo zmazať už existujúci kurz je oprávnený len administrátor. Pridať nový kurz je možné kliknutím na tlačítko *Pridať nový kurz*, alebo výberom kategórie kurzu a následným kliknutím na tlačítko *Pridať nový kurz*.

2. Po kliknutí sa zobrazí blok (obr.Obrázok 7). V posledných štyroch položkách je možné



Obrázok 7.

špecifikovať ako budú v kurze označovaní učiteľia a študenti. Pomocou jednotlivých položiek je možné nadefinovať vlastnosti nového kurzu:

a. kategória - slúži na výber kategórie pre kurz,

b. celé meno - plné meno pre kurz (pod akým sa zobrazí používateľom),

c. skrátené meno - skratka pre kurz,

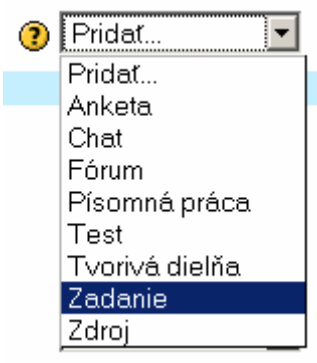
d. zhrnutie - výstižný a zaujímavý odsek, ktorý vysvetľuje o čom je tento kurz,

e. prístupnosť - či kurz je alebo nie je prístupný pre študentov,
f. prihlasovací kľúč - je možné špecifikovať heslo potrebné pre študenta na prihlásenie do kurzu,
g. hosťovský prístup - či je alebo nie je povolený vstup používateľov prihlásených ako hosť,
h. formát - v ako formáte zobrazenia sa bude zobrazovať kurz (týždenný formát, tématický a pod.),
i. nové správy na zobrazenie - či a koľko posledných správ sa má zobrazovať,
j. dátum začiatku kurzu - dátum od ktorého začne kurz,
k. počet týždňov/tém - dĺžka trvania kurzu (v závislosti od zvoleného formátu kurzu),
l. zobraziť aktuálnu činnosť - či sa majú akcie všetkých používateľov, ktorí sa prihlásili do kurzu zaznamenávať a sprístupniť pre učiteľa na prezeranie,
m. zobraziť známky - či sa hodnotenia úloh študentov v kurze majú zobrazovať všetkých alebo len danému študentovi.

3. Po vyplnení položiek kliknite na tlačítko *Uložiť zmeny* a systém Dagwood vytvorí podľa vašich požiadaviek nový kurz.
4. Úspešné vytvorenie kurzu potvrdťe kliknutím na nápis *Pokračovať* na ďalšej stránke, ktorá sa automaticky zobrazí.
5. Následne sa automaticky presuniete do novovytvoreného kurzu. Novovytvorený kurz má automaticky vytvorené fórum noviniek v hornej spoločnej sekcii.
6. Priradenie nových učiteľov do kurzu (tvorca kurzov je priradený automaticky) je možné klinkutím na položku *Teachers* v bloku Administratíva (vľavo dole). Pridanie učiteľa sa vykonáva kliknutím na nápis *Pridať učiteľa* v zozname používateľov v spodnej tabuľke - po pridaní sa tento používateľ pridá medzi učiteľov tohto kurzu, t.j. presunie sa do hornej tabuľky. Odstránenie učiteľa je možné podobným spôsob pomocou kliknutia na nápis *Odstrániť učiteľa* v hornej tabuľke. Vykonané zmeny potvrdťe kliknutím tlačítko *Uložiť zmeny*.

10.3.2 Vytváranie nových zadaní

1. Kliknite na rolovacie menu *Pridať* v príslušnom týždni (téme) a vyberte položku *Zadanie* (obr. Obrázok 8).



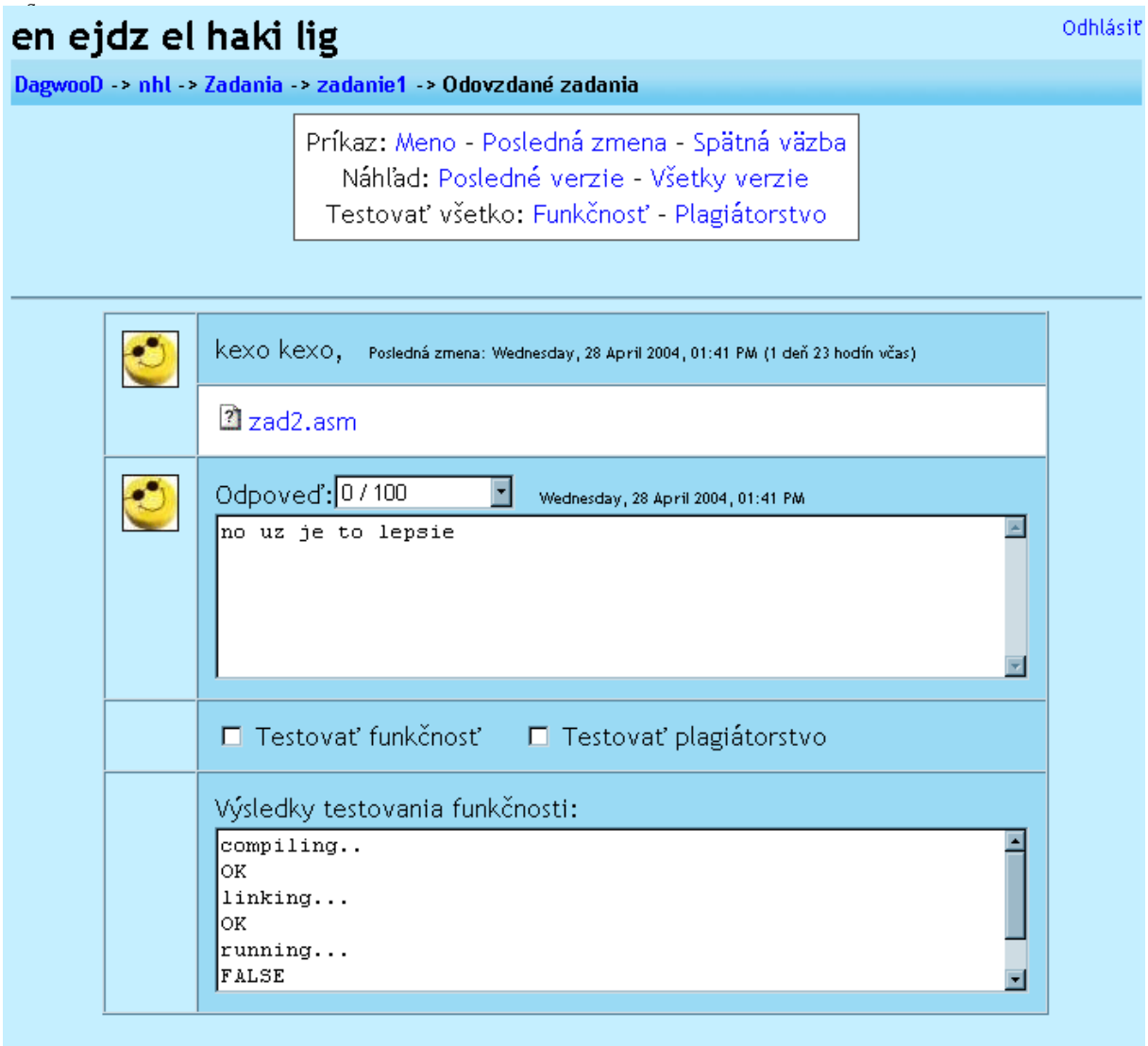
Obrázok 8.

2. Po výbere za zobrazí obrazovka pre pridanie nového zadania, ktorá obsahuje nasledujúce položky:
   a. názov zadania - kompletný názov pre zadanie.
   b. popis - vlastný text zadania.
   c. typ zadania - určuje či sa jedná o zadanie, ktorého riešenie študent zapíše ako text, alebo sa riešenie nahrá ako súbor.
   d. povoliť opätovné predloženie - či môže študent nahrať zadanie na server viackrát (toto má význam len pri zadaniach, ktorých sa riešenie nahráva).
   e. povoliť verzie - či sa pri viacnásobnom uploadovaní uchováva aj minulá verzia zadania, alebo sa uchováva len najnovšia (toto má význam len pri zadaniach, ktorých sa riešenie nahráva).
   f. povoliť mazanie zadaní - či môže študent zmazať svoje uz odovzdané zadanie (toto má význam len pri zadaniach, ktorých sa riešenie nahráva).
   g. známka - bodovanie zadania, t.j. či má byť zadanie bodované a ak áno, aký je max. počet bodov
   h. maximálna veľkosť - maximálna povolená veľkosť pre nahrávaný súbor.
   i. dátum vypracovania - dátum, do ktorého má byť zadanie odovzdané
3. Po vyplnení a nastavení položiek kliknite na tlačítko „Uložiť zmeny“. Týmto sa vytvorí zadanie a vloží do zvolenej sekcie (viď krok 1).

10.3.3 Testovanie zadaní na funkčnosť a plagiátorstvo

1. Kliknite na názov zadania v hlavnom okne kurzu.
2. Zobrazia sa vám informácie o zadaní (dátum vypracovania, hodnotenie, atď.). Tieto informácie je možné dodatočne aktualizovať kliknutím na tlačítko „Aktualizovať toto Zadanie“ v pravom hornom rohu.

3. Pod tlačítkom „Aktualizovať toto Zadanie“ sa nachádza nápis „Zobraziť predložené zadania“ - po kliknutí naň sa zobrazí obrazovka, ktorá umožňuje prezerať, hodnotiť a testovať už odovzdané zadania (obr. Obrázok 9). V hornej časti obrazovky sa nachádza textové menu



o

d

o

Obrázok 9.

vzdaných zadaní poďla zadaných požiadaviek, prípadne na spustenie testovania pre všetky zadania. Položky tohto menu majú nasledovný význam:

a. Príkaz:

   i. meno - zoradí zadania vzostupne podľa mena študenta

   ii. posledná zmena - zoradí zadania podľa dátumu poslednej zmeny zadania (najnovšie najskôr)

iii. spätná väzba - zoradí zadania podľa toho či už boli ohodnotené učiteľom (ohodnotené najskôr)

b. Náhľad:

i. Posledné verzie - zobrazí len posledné verzie odovzdaných zadaní

ii. Všetky verzie - zobrazí všetky verzie odovzdaných zadaní

c. Testovať všetko:

i. Funkčnosť - otestuje všetky zadania po funkčnej stránke

ii. Plagiátorstvo - otestuje všetky zadania na plagiátorstvo

Každé zadanie je možné ohodnotiť určitým počtom bodov a taktiež slovným komentárom. Po ohodnotení zadania už toto zadanie nemôže študent zmazať. Pri každom zadaní je možné určiť či sa má testovať z hľadiska funkčnosti a z hľadiska plagiátorstva, pričom test sa spustí po kliknutí na tlačítko „Uložiť odpovede a spustiť test“. Pre testovanie funkčnosti je potrebné zadať vstupný (t.j. súbor v ktorom je napísané čo je pre spustenie programu potrebné napísať na príkazový riadok) a výstupný subor (čo má testovaný program vypisovať na obrazovku). Výsledky testov funkčnosti sa zobrazia zvlášť pri každom testovanom zadaní (je nutné klinúť na „refresh“ tlačítko prehliadča). Pre zobrazenie výsledkov testov plagiátorstva je potrebné presunúť sa o jednu stránku naspäť (napr. kliknutím na názov zadania v ľavom hornom rohu obrazovky). Výsledy testov sú časovo zoradené v spodnej tabuľke. Po kliknutí na jeden z výsledkov testovania, zobrazi sa nová stránka s tabuľkou, kde sú zobrazené výsledky testovania. Tieto výsledky je možné po kliknutí na tlačítko XLS uložiť do súboru programu MS Excel a teda ho aj v tomto programe otvárať a pohodlne používať. V oboch prípadoch treba na vykonanie testov určitý čas (v závislosti od rozsahu testovaných zadaní), preto výsledky nemusia byť hneď k dispozícii. Ak určitý druh testu ešte prebieha, nie je možné spustiť tento test znova.

10.3.4 Manažovanie pohľadov

1. Najprv sa uistite, že máte zapnuté upravovanie (na tlačítku v pravom hornom rohu musí byť napísané "Vypnúť upravovanie"), v prípade že je upravovanie vypnuté, zapnete ho kliknutím na tlačítko "Zapnúť upravovanie" v pravom hornom rohu.
2. Kliknite na nápis „Pohľady“ v bloku Administratíva, zobrazí sa obrazovka v ktorej je možné p

e

r

Obrázok 10.

spektívy a mazať existujúce (obr. Obrázok 10).

3. Pre vytvorenie novej perspektívy napíšte do prázdneho textového poľa pod nápisom „Upraviť pohľady v kurze“ názov pre nový pohľad, vyberte z rolovacieho menu typ pohľadu a kliknite na tlačítko „Pridaj nový pohľad“. Nový pohľad sa pridá do zoznamu pohľadov (dolná tabuľka). Pre zmazanie existujúcej perspektívy kliknite na ikonu zmazania v riadku danej perspektívy.

11 Inštalačná príručka

11.1 Distribúcia

Dagwood sa distribuuje výlučne v zdrojovych kódoch. Systém by mal byť schopný prevádzky na ľubovoľnom „UNIX-like“ systéme. Primárnou platformou, na ktorej je aplikácia vyvíjaná, je OS GNU/Linux. Preto by balo byť na tomto systéme najmenej problémov.

Na inštaláciu je potrebné si zaobstarať balík obsahujúci modifikovanú verziu systému Moodle a balík obsahujúci rozširujúce moduly a démona na ich správu. Je ich možné nájsť na stránke projektu (http://www2.dcs.elf.stuba.sk/TeamProject/2003/team10/) alebo na priloženom CD. Aj keď systém veľmi rýchlo nadobúda požadované vlastnosti, je ešte stále v rannom štádiu vývoja. Vývojový tím preto uvíta všetky pripomienky a prípadné problémy s prevádzkou Dagwoodu.

11.2 Predpoklady inštalácie

Na úplnú inštaláciu systému je potrebné, aby systém obsahoval nasledovné programové vybavenie, podľa možnosti v čo najnovších verziách:

- Apache: Webový server Apache je potrebný na prevádzku používateľského rozhrania. Dagwood nemá žiadne špecifické požiadavky na verziu Apache.
- PHP: Webový server musí podporovať PHP skripty, pretože celá logika používateľského rozhrania je napísaná v tomto jazyku.
- GCC: Väčšina systému je napísaná v rôznych interpretovaných jazykoch, niektoré časti sú však napísané aj v jazyku C. Na ich kompiláciu by pravdepodobne stačil ľubovoľný ANSI C kompilátor, odporúčame však práve prekladač GCC.
- Bash: Niektoré časti modulu testovania funkčnosti sú vo forme shellových skriptov. Na ich funkcionalitu je potrebný interpret kompatibilný s Bourne shellom.
- Perl: Veľká časť systému, hlavne rozhrania SOAP, sú napísané v jazyku Perl. Opäť odporúčame použiť čo najnovšiu verziu.
- Perlové moduly: Dagwood bol napísaný za podpory niektorých Perlových modulov, ktoré sa štandardne nedodávajú. Ide o moduly SOAP::Lite a Net::Server.
- Dosemu: Na testovanie funkčnosti zadaní je využitý program *dosemu*. Je potrebné mať nainštalovanú verziu 1.2.0 alebo vyššiu.

- MySQL, PostgreSQL: Dagwood ukladá svoje údaje do SQL databázy. Dokáže spolupracovať s oboma uvedenými databázami.

11.3 Inštalácia

Inštalácia Dagwoodu prebieha podobne ako u väčšiny voľne šíriteľných programov na platforme Linux. Pre jednoduchú inštaláciu s nastavenými štandardnými hodnotami stačí najprv rozbaliť archív *dagwood.tar.gz*:

```
tar xzf dagwood.tar.gz
cd dagwood
```

Ďalej inštalácia pokračuje klasickým postupom:

```
./configure
make
make install
```

Celý „back-end“ systému by mal byť v tomto okamihu nainštalovaný v adresári */usr/local/dagwood*. Odporúčame ručne skontrolovať všetky nastavenia systému v adresári */usr/local/dagwood/etc*. Aby bolo možné používať moduly testovania funkčnosti a plagiátorstva, je ešte potrebné umiestniť CGI skirpty predstavujúce SOAP rozhranie modulov tak, aby boli dosiahnuteľné na URL zadanej v konfiguračnom súbore démona.

Na presnejšie určenie parametrov inštalácie je potrebné zadať skriptu *configure* niektoré parametre, ktoré určujú požadované vlastnosti. Presný zoznam parametrov je možné dosiahnuť spustením:

```
./configure --help.
```

V nasledovnom príklade je ukážka inštalácie systému do adresára */opt/dagwood*. Nainštaluje sa tam iba hlavná časť systému a modul plagiátorstva. Zároveň sa nastavuje aj adresár obsahujúci súbory na činnosť modulov:

```
./configure --prefix=/opt/dagwood --spool=/var/spool/dagwood --
daemon-port=1234 \
          --enable-plagiat --plagiat-url=http://localhost/cgi-
bin/dagwood-plagiat.cgi \
          --disable-tf
make
make install
```

Po vykonaní týchto príkazov je opäť potrebné umiestniť CGI skripty tvoriace SOAP rozhranie modulov tak, aby boli dosiahnuteľné na URL zadanej pri inštalácii. Opäť, adresa týchto skriptov je napísaná v konfiguračnom súbore démona a je ju možné kedykoľvek zmeniť.

Vrstvu implementujúcu používateľské rozhranie sme zbalili do samostatného balíka *moodle.tar.gz*. Dôvodom je tá skutočnosť, že žiadnu časť systému Moodle nie je potrebné pred inštaláciou kompilovať. Predišli sme tak aj problémom s rôznym umiestnením adresárov prístupných cez HTTP server na rôznych systémoch. Jedinou podmienkou je to, že tento balík a démon zabezpečujúci spojenie s modulmi musia byť nainštalované na jednom stroji.

Inštalácia časti Moodle spočíva iba v rozbalení balíka v adresári prístupnom cez webový server Apache a nastavení prístupových parametrov k databáze:

```
cd /var/www/htdocs
tar xzf moodle.tar.gz
cd moodle
vi config.php
```

V súbore *config.php* je potrebné nastaviť parametre webového rozhrania systému. Ide hlavne o údaje potrebné na prístup k databáze. Ak bol pri inštalácii démona zmenený TCP port, na ktorom démon očakáva príkazy, je potrebné zmenený port upraviť aj v tomto súbore.

11.4 Spustenie

Na správne fungovanie je potrebné, aby bol najprv spustený démon. Spustiť ho je možné napriklad takto:

```
nohup /cielovy/adresar/bin/dagwood-daemon.pl &
```

Keďže moduly systému sú prístupné ako obyčajné CGI skripty, o ich spúšťanie sa stará server Apache na hostiteľskom počítači.

Po inštalácii systému Dagwood je potrebné, aby sa správca prvýkrát prihlásil do systému Moodle na adrese *http://server/moodle/* a nastavil tam používateľské kontá vyučujúcich. Tí sa potom môžu prihlásiť do systému a vytvárať tam obsah svojich predmetov. Presný popis konfigurácie systému a práce v ňom je popísaný v používateľskej príručke.

12 Zhodnotenie

Primárným cieľom projektu nebolo zhotoviť produkt, ale precvičiť si spoluprácu v tíme. Každý člen tímu už mal v minulosti možnosť sa stretnúť s prácou v tíme, napriek tomu predmet Tímový projekt bol prínosný pre každého z nás. Mohli sme si precvičiť časť znalostí z predmetu Princípy softvérového inžinierstva, ale naučili sme sa aj efektívnejšie organizovať si vlastnú prácu na projekte a aj samotný projekt.

Ďalším cieľom projektu bolo zhotoviť podľa zadania produkt vrátane jeho technickej dokumentácie. Od začiatku projektu sme stavali na progresívnosti nových technológií založených na jazyku XML ale aj na modularite systému. Podarilo sa nám vykonať pomerne hlbokú analýzu vzhľadom na nové technológie a nakoniec prísť s hrubým návrhom systému, ktorý by bol využiteľný nie len pri výučbe predmetu Systémové programovanie a asemblery, ale aj pri iných, ba dokonca by poskytoval dobrý základ pre informačný systém (možno nie len) fakulty. Vužitie jazyka XML, modularity systému, kde je prepojenie medzi nimi je realizované cez „web“ služby ponúka veľku flexibilitu. Takto možno vymeniť časť systému bez toho, aby sa museli vykonať zmeny aj v iných častiach. Ďalšou veľkou výhodou je aj možnosť rozložiť záťaž medzi niekoľko inštancií rovnakej služby (ak to jej charakter dovoľuje) v prípade návalu požiadaviek na systém a teda udržania reakcie systému na akceptovateľnej hladine, keďže predpokladá sa, že systém budú používať stovky používateľov.

Náš zámer bol veľmi odvážny, iniciovali sme aj verejnú diskusiu o problematike eVzdelávania na našej fakulte. Keďže ide o „mamuťí“ projekt, spravili sme v letnom semestri rozhodnutie, kde sme sa dohodli aj s pedagógom na novej špecifikácii tak, aby bola zrealizovateľná do konca semestra a aj následne použiteľná za účelom testovania myšlienky v praxi, ale aj ako dobrý základ pre ďalšie projekty.

Výsledkom nášho snaženia je hotový produkt s menom Dagwood. Vzhľadom na to, že aj táto upravená špecifikácia predstavovala veľké množstvo práce, vzhľadom na nedostatok času sme nezrealizoval úplne všetko ako sme chceli (viď. podkapitoly Ďalšie možnosti úprav v kapitolách 7.4 až 7.9).

Veríme, že produkt splnil svoj cieľ a očakávania a bude dobrým základom pre realizáciu myšlienky virtuálnej univerzity a aj fakultného informačného systému.

13 Zoznam použitých skratiek

Skratka	Význam
ADL	Advanced Distributed Learning Initiative
AICC	Aviation Industry CBT Commitee
BSD	Berkeley Software Distribution
CD-ROM	Compact Disc Read-Only Memory
CMS	Content Management System
CORBA	Common Object Request Broker Architecture
CSS	Cascade Style Sheet
CVS	Concurrent Version System
DCOM	Distributed Component Object Model
DTD	Data Type Definition
FAQ	Frequently Asked Question - často kladené otázky
FIIT	Fakulta informatiky a informačných technológií
GNU	GNU's Not Unix
GPL	General Public License
HTML	Hypertext Markup Language
ID	Identifikátor
IEEE	Institute of Electrical and Electronics Engineers
IIS	Internet Information Server
IMAP	Internet Message Access Protocol
IP	Internet Protocol
ISO	International Organization for Standardization
JMS	Java Messaging Service
JSI	Jazyk symbolických inštrukcií
LDAP	Lightweight Directory Access Protocol
LOM	Learning Object Metadata
MIME	Multipurpose Internet Mail Extensions
NNTP	Network News Transport Protocol
OASIS	Organisation for the Advanced of Structured Information Standards
PDF	Portable Document Format
PHP	Pretty Home Pages
POP3	Post Office Protocol, version 3
QML	QUIZIT Markup Language
RKS-GST	Runing-Karp-Rabin Greedy String Tiling

RPC	Remote Procedure Calling
SAML	Security Assertion Markup Language
SCORM	Sharable Content Object Reference Model
SGML	Standard Generalized Markup Language
SMTP	Simple Mail Transfer Protocol
SOAP	Simple Object Access Protocol
SPA	Systemove programovanie a asemblery
SQL	Structured Query Language
SSL	Secure Socket Layer
STU	Slovenská technická univerzita
SUNW	burzový kód firmy Sun Microsystems
TLS	Transport Layer Security
UCS	Universal Character Set
UDDI	Universal Description, Discovery and Integration Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTF	UCS Transformation Format
W3C	Worldwide Web Consortium
WBT	Web-Based Training
WSDL	Web Service Definition Language
WYSIWIG	What You See Is What You Get
XML	Extensible Markup Language
YAP	Yet Another Program Plagiarism Detector

14 Zoznam použitej literatúry

[1] Lonoco, L., Barnette, N. a Fox, E.: Online Evaluation in WWW-based courseware, SIGCSE '97, in SIGCSE Bulletin 29 (1), 1997.

[2] Tinoco, L., et al: QUIZIT : An Interactive Quiz System for WWW-based Instruction, 1996

[3] Mišovič, B.: Automatizované testovanie vedomostí, FEI STU, Bratislava, 2003

[4] PIKULA, M. ISZÁK, P. et al: Podpora cvičení v predmete SOJ, FEI STU, Bratislava, 2001

[5] VARGA,Z.: Automatické vyhodnocovanie programov v JSI., FEI STU, Bratislava , 2000

[6] Barankai J., et al: Počítačová podpora cvičení v predmete Strojovo orientované jazyky, FEI STU, Bratislava , 2000

[7] Barton J. J., Thatte S.,Nielsen H. F.: SOAP Messages with Attachments, W3C, 2000, http://www.w3.org/TR/SOAP-attachments

[8] Box D.,Ehnebuske B., et al: Simple Object Access Protocol (SOAP) 1.1, W3C, 2000, http://www.w3.org/TR/SOAP/

[9] Bray T., Paoli J., et al: Extensible Markup Language (XML) 1.0 (Second Edition), W3C, 1997, http://www.w3.org/TR/REC-xml

[10] Christensen E., Curbera F., et al: Web Services Description Language (WSDL) 1.1, W3C, 2001, http://www.w3.org/TR/wsdl.html

[11] Kohl, J., Neuman, C.,: The Kerberos Network Authentication Service (V5), The Internet Society, RFC 1510, 1993

[12] Fielding, R,et al.: Hypertext Transfer Protocol - HTTP/1., The Internet Society, RFC 2616, 1999

[13] Semancik, R.: Internet applications security, FEI STU, Bratislava, 2002

[14] Hallam-Baker, P., Maler, E.: Assertions and Protocol for the OASIS Security Assertion Markup Language, OASIS Standard, 2002

[15] Mishra, P., et al.: Bindings and Profiles for the OASIS Security Assertion Markup Language, OASIS Standard, 2002

[16] Bray, T., et al.: Extensible Markup Language (XML), W3C Recomendation, 2000

[17] Hodges, J.: Liberty Architecture Overview, Liberty Alliance Project Specification, 2002

[18] Rouault, J.: Liberty Bindings and Profiles Specification, Liberty Alliance Project Specification, 2002

[19] Gudgin, M, et al.: SOAP Version 1.2 Part1: Message Framework, W3C Working draft, 2002

Prílohy

Príloha A - WSDL opis jednoduchej SOAP služby

```
<?xml version="1.0" encoding="utf-8"?>
<definitions name="StockQuote"
targetNamespace="urn:x-example:services:StockQuote"
xmlns:tns="urn:x-example:services:StockQuote"
xmlns:xsd1="http://example.com/stockquote.xsd"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns="http://schemas.xmlsoap.org/wsdl/">

<types>
<schema targetNamespace="http://example.com/stockquote.xsd"
xmlns="http://www.w3.org/2000/10/XMLSchema">
<element name="GetLastTradePrice">
<complexType>
<all>
<element name="symbol" type="string"/>
</all>
</complexType>
</element>
<element name="GetLastTradePriceResponse">
<complexType>
<all>
<element name="Price" type="float"/>
<element name="Currency" type="string"/>
</all>
</complexType>
</element>
</schema>
</types>

<message name="GetLastTradePriceInput">
<part name="body" element="xsd1:GetLastTradePrice"/>
</message>
<message name="GetLastTradePriceOutput">
<part name="body" element="xsd1:GetLastTradePriceResponse"/>
</message>

<portType name="StockQuotePortType">
<operation name="GetLastTradePrice">
<input message="tns:GetLastTradePriceInput"/>
<output message="tns:GetLastTradePriceOutput"/>
</operation>
</portType>

<binding name="StockQuoteSoapBinding" type="tns:StockQuotePortType">
<soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
<operation name="GetLastTradePrice">
<soap:operation soapAction=""/>
<input><soap:body use="literal"/></input>
<output><soap:body use="literal"/></output>
</operation>
</binding>
```

```
<service name="StockQuoteService">
<documentation>My first Service</documentation>
<port name="StockQuotePort" binding="tns:StockQuoteBinding">
<soap:address location="http://example.com/StockQuote"/>
</port>
</service>

</definitions>
```

Príloha B - WSDL špecifikácia SAML rozhrania

```
<?xml version="1.0"?>
<definitions name="SAMLProtocol" targetNamespace="http://www.oasis-
open.org/committees/security/docs/WSDLdefinitions.wsdl"
xmlns:tns="http://www.oasis-
open.org/committees/security/docs/WSDLdefinitions.wsdl"
xmlns:samlp="http://www.oasis-
open.org/committees/security/docs/draft-sstc-schema-protocol-19.xsd"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns="http://schemas.xmlsoap.org/wsdl/">
<import namespace="http://www.oasis-
open.org/committees/security/docs/draft-sstc-schema-protocol-19.xsd"
location="http://www.oasis-open.org/committees/security/docs/draft-
sstc-schema-protocol-19.xsd" />

<!-- The body of the request is exactly a samlp:Request -->
<message name="SAMLRequestMessage">
<part name="body" element="samlp:Request"/>
</message>

<!-- The body of the corresponding response is exactly a
samlp:Response
-->
<message name="SAMLResponseMessage">
<part name="body" element="samlp:Response"/>
</message>

<!-- And the request-response protocol goes like this -->
<portType name="SAMLRequestPortType">
<operation name="SAMLRequest">
<input message="tns:SAMLRequestMessage"/>
<output message="tns:SAMLResponseMessage"/>
</operation>
</portType>

<!-- This describes how the request-response maps onto SOAP -->
<binding name="SAMLRequestSoapBinding"
type="tns:SAMLRequestPortType">
<soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
<operation name="SAMLRequest">
<soap:operation soapAction="http://www.oasis-
open.org/committees/security/SAMLRequest"/>
<input>
<soap:body use="literal" namespace="http://www.oasis-
open.org/committees/security/docs/draft-sstc-schema-protocol-19.xsd"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
</input>
<output>
<soap:body use="literal" namespace="http://www.oasis-
open.org/committees/security/docs/draft-sstc-schema-protocol-19.xsd"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
</output>
```

```
</operation>
</binding>

<!-- And this says that the SOAP service is available at a
particular URL -->
<service name="SAMLRequestService">
<documentation>This is the only per-installation
data</documentation>
<port name="SAMLRequestPort" binding="tns:SAMLRequestSoapBinding">
<soap:address
location="http://pizza.bgs.sk:58080/amserver/SAMLSOAPReceiver "/>
</port>
</service>
</definitions>
```

Príloha C - Používateľská príručka prototypu

Nasledovný prototyp produktu bol vytvorený tímom 10 Dagwood, v rámci tímového projektu. Prototyp je zameraný na technologickú stránku navrhovaného riešenia, a teda hlavný dôraz je kladený na použite navrhovaných moderným technológií, a prezentuje možnosť znovupoužiteľnosti už vytvoreného kódu [1].

Cieľom prototypovania boli základné funkcie pre prácu so zadaniami - zber zadaní, prezentácia zadaní a kontrola voči plagiátorstvu. Taktiež bola implementovaný aj jednoduchý spôsob pridávania užívateľov.

Keďže sa jedná o funkčný prototyp webovej aplikácie, nie je ho možné nainštalovať "offline", preto je prístupný na internetovej adrese http://labss2.dcs.elf.stuba.sk/TeamProject/2003/team10/cgi-bin/soap/prototype.cgi.

Pri práci s ním sa predpokladajú základné znalosti s prácou s vybraným grafickým operačným systémom (Windows) a s internetovým prehliadačom.

C.1 Požiadavky na hardvér a softvér

Jedinou požiadavkou pre prácu s prototypom je ľubovolný typ počítača, na ktorom je možné prevádzkovať grafický internetový prehliadač.

C.2 Popis funkcií prototypu

Po otvorení vyššie uvedenej internetovej adresy, sa zobrazí hlavná stránka prototypu. V ľavej časti stránky je zobrazené menu pre prístup k jednotlivým funkciám systému. V pravej časti je zobrazený zoznam študentov, a ovládací prvok pre pridávanie nových študentov. V strednej časti stránky sa zobrazuje samotný obsah, tj. ďalšie ovládacie prvky, alebo samotné údaje.

Pri funkciách "Odovzdaj zadanie" a "História zadaní" musí byť v zozname študentov vybraný jeden z používateľov.

C.1.1 Pridanie nového používateľa - študenta do systému

V pravej časti stránky zadajte do poľa "Nový používateľ" meno pridávaného študenta, a následne kliknite na "Zaregistruj". Nový používateľ bude pridaný do systému a zobrazený v zozname.

C.1.2 Odovzdanie zadania používateľom

V zozname študentov vyberte ľubovolného študenta, a v menu vyberte "Odovzdaj zadanie". Zobrazí sa formulár pre výber súboru a odovzdanie zadania.

Kliknite na "Procházet" (resp. "Browse", a pod. - závisí od jazykovej mutácie operačného systému), a vyberte súbor, ktorý chcete odovzdať. Po kliknutí na "Odovzdaj" je zadanie prenesené na server,a priradené vybranému študentovi.

C.1.3 Prezeranie zadaní odovzdaných študentom

Systém umožňuje archiváciu rôznych verzií zadaní odovzdaných študentom.

V zozname študentov vyberte ľubovolného študenta (napr. "u1"), a v menu vyberte "História zadaní". Následne je zobrazený zoznam verzií odovzdaných zadaní daného študenta, alebo informácia, že študent ešte žiadne zadanie neodovzdal.

V prvom prípade kliknite na ľubovolnú verziu zadania, a následne sa zobrazí samotné odovzdané zadanie.

C.1.4 Kontrola zadaní voči plagiátorstvu

V menu vyberte "Porovnanie zadaní". Zobrazí sa tabuľka so všetkými používateľmi, ktorí odovzdali zadanie, v ktorej je pre každú dvojicu uvedená percentuálna podobnosť ich zadaní.

C.3 Odkazy

[1] Dagwood - Tímový projekt 2003/2004, FIIT STU

Príloha D - Záznamy o vykonávaní zmien počas implementácie

Aby sme mohli jednoduchšie a čo najrýchlejšie zdokumentovať životný cyklus podsystému a celého systému sme vytvorili vlastné DTD podľa ktorého sme robili záznamy do changelog súborov. Na základe týchto súborov vieme nájsť rýclejšie prípadnú príčinu chyby.

D.1 Báza konceptov

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>
<!DOCTYPE file_changelog SYSTEM
"http://www2.dcs.elf.stuba.sk/TeamProject/2003/team10/DTD/file_chang
elog.dtd">

<file_changelog>

  <file filename="perspective.php" module="znalosti" language="php"
since="2004-03-21">
    <description>
      Cesta: moodle/course/perspective.php
      Skript zodpovedny za manazovanie pohladov (pridavanie,
nastavovanie viditelnosti a zrudenie pohladu).
    </description>

    <record who="dalxo" when="2004-03-21 18:00" curr_ver="0.1"
prev_ver="0.0">
      <change desc="vytvorenie suboru">Vytvorenie skriptu. Vie
pridavat, uberat a nastavovat viditelnost.</change>
    </record>
    <record who="dalxo" when="2004-04-03 18:00" curr_ver="0.2"
prev_ver="0.1">
      <change desc="drobna uprava">Malicko som upravil skript, aby
neponukol moznost pridavat pohlady, ked pouzivatel nie je v rezime
editovania.</change>
    </record>
    <record who="dalxo" when="2004-04-03 23:15" curr_ver="0.3"
prev_ver="0.2">
      <change desc="bugfix">Spravne vypisanie poctu modulov
priradenych danemu pohladu.</change>
    </record>
    <record who="dalxo" when="2004-05-02 22:30" curr_ver="0.4"
prev_ver="0.3">
      <change desc="bugfix">Ak sa zmaze typ tree, zabudli sa zmazat
zaznamy v tabulke tree_view.</change>
      <change desc="bugfix">Ak sa zmaze typ, vymaze ho aj z
course_modules v stlpci views</change>
    </record>

  </file>
```

```
  <file filename="view.php" module="znalosti" language="php"
since="2004-03-21">
    <description>
      Cesta: moodle/course/view.php
      Upraveny puovodny PHP skript v Moodle, ktory zodpovedal za
predprocesing udajov pri zobrazovani
      baz konceptov v Moodle.
    </description>

    <record who="dalxo" when="2004-03-21 18:00" curr_ver="0.1"
prev_ver="0.0">
      <change desc="uprava suboru">
        Upravil som puovdny subor, aby podla
        zvoleneho pohladu vedel zistit, ktory skript vyvovat a
vyvolal ho.
      </change>
    </record>
    <record who="dalxo" when="2004-03-28 21:00" curr_ver="0.2"
prev_ver="0.1">
      <change desc="pridanie spracovania argumentov pri moduloch">
          Pridal som spracovanie na pridavanie a uberanie modulov z
pohladov.
      </change>
    </record>
    <record who="dalxo" when="2004-04-02 18:15" curr_ver="0.3"
prev_ver="0.2">
      <change desc="bugfix pri mazani modulov z pohladov">
          Zle sa vykonavalo mazanie modulu z pohladu. Oprava zlej
podmienky a prace s premennymi.
      </change>
    </record>

    <record who="dalxo" when="2004-05-02 23:45" curr_ver="0.4"
prev_ver="0.3">
      <change desc="vylepsenie delper a addper">
          addper teraz vyvola funkciu z lib.php add_to_comma_list()
miesto vlastneho kodu.
          delper teraz vyvola funkciu z lib.php
remove_from_comma_list() miesto vlastneho kodu.
      </change>
    </record>

  </file>

  <file filename="weeks.php" module="znalosti" language="php"
since="2004-03-21">
    <description>
      Cesta: moodle/course/format/weeks.php
      Puovodny skript zobrazoval predmet vo "Weekly outline" forme.
Zodpoveda za zobrazovanie pohladov
      typu: Single, Multi a Whole.
    </description>

    <record who="dalxo" when="2004-03-21 18:00" curr_ver="0.1"
prev_ver="0.0">
      <change desc="pridanie side_block">
```

```
          Pridane zobrazenie side_block na vyber pohladov. Podla
zvoleneho pohladu zobrazi v hlavicke meno pohladu.
      </change>
    </record>
    <record who="dalxo" when="2004-03-28 21:00" curr_ver="0.2"
prev_ver="0.1">
      <change desc="filtrovanie sekcii podla typu pohladu">Podla
typu pohladu vie odfiltrovat sekcie.</change>
    </record>
    <record who="dalxo" when="2004-04-05 17:00" curr_ver="0.3"
prev_ver="0.2">
      <change desc="uprava filtrovania sekcii">
          Upravil som to, aby kontrolovalo podla mena sekcie a nie
ID kvuoli vyssec citatelnosti kodu.
      </change>
    </record>
    <record who="dalxo" when="2004-04-21 16:00" curr_ver="0.4"
prev_ver="0.3">
      <change desc="bugfix">
          Pri vypisovani single sa niekde startil uzatvaraci tag td.
      </change>
    </record>
    <record who="dalxo" when="2004-05-02 22:00" curr_ver="0.5"
prev_ver="0.4">
      <change desc="zrusenie niektorych side_block-ov">
          Zrusil som side_block-y F.A.Q., najblizsie udalosti a
kalendar.
      </change>
    </record>

  </file>

  <file filename="lib.php" module="znalosti" language="php"
since="2004-03-21">
    <description>
      Cesta: moodle/course/lib.php
      Kniznica s funkciami, ktore sa pouzivaju pri zobrazovani a
manazovani predmetu.
    </description>

    <record who="dalxo" when="2004-03-21 18:00" curr_ver="0.1"
prev_ver="0.0">
      <change desc="pridanie funkcie">
        Pridana funkcia show_views_names($ids, $courseid).
Vykresluje lavy horny blok Perspectives, kde sa nachadza menu
        na vyber dostupnych pohladov pre daneho pouzivatela.
      </change>
    </record>
    <record who="dalxo" when="2004-03-28 21:00" curr_ver="0.2"
prev_ver="0.1">
      <change desc="uprava funkcie">
        Uprava funkcie print_section(), ktora vypisuje obsah sekcii
(vratane modulov).
        Vedla modulu sa zobrazi zoznam pohladov v ktorych je. Ak je
stranka v rezime editovania,
```

```
        zobrazi sa v pripade zakladneho pohladu Whole aj popup list
na pridavanie do zodpovedajucich pohladov.
        Ak nie sme vo Whole pohlade, zobrazi sa iba moznost vymazat
z aktualneho pohladu.
      </change>
    </record>
    <record who="dalxo" when="2004-04-03 21:15" curr_ver="0.3"
prev_ver="0.2">
      <change desc="pridanie funkcie">
          Pridal som funkciu print_modules_not_assoc_in_tree() ktora
vypise zoznam neasociovanych subor
          k uzlom strmu. Zatial ich vypise vsetky, ktore boli
priradene do konkretneho pohladu typu strom.
      </change>
      <change desc="pridanie funkcie">
          get_modules_in_per() -> hodi zoznam modulov, ktore su
priradene danemu pohladu. Uzitocne napr.
          pri zistovani poctu modulov v pohlade. Je vyuzivana aj
funkciou print_modules_not_assoc_in_tree().
      </change>
      <change desc="uprava funkcie">
          Jemne som modifikoval funkciu print_section() co sa tyka
vypisovania pohladov. Aby tam v pripade
          needitovania nebola zvisla ciara a za nou nic. Tak som to
osetril.
      </change>
    </record>

    <record who="dalxo" when="2004-04-05 17:00" curr_ver="0.4"
prev_ver="0.3">
      <change desc="uprava funkcie">
        Uprava funkcie print_section(), aby sa pohlad rozlisoval
podla name a nie type.
      </change>
      <change desc="bugfix">
        show_views_names() ak nenasla zaznam o pohladoch, tak dala
exit() miesto return.
        Ak bol vytvoreny novy predmet bez pohladov, potom rozbilo
celu stranku.
      </change>
      <change desc="bugfix">
        v print_section() som v kode, co generuje zoznam pre popup
na pridavanie pohladov, osetril
        som podmienku "if(isset($views) and $isediting)" na
"if(isset($views) and !empty($views) and $isediting)"
      </change>
    </record>

    <record who="dalxo" when="2004-04-21 16:00" curr_ver="0.5"
prev_ver="0.4">
      <change desc="zmena print_modules_not_assoc_in_tree()">
          Zmenilo sa GUI na pridavanie uzlov do stromu.
      </change>
    </record>
```

```
    <record who="dalxo" when="2004-05-02 22:00" curr_ver="0.6"
prev_ver="0.5">
      <change desc="funkcia print_modules_not_assoc_in_tree()
premiestnena do tree.php">
          Nema zmysel, aby bola v lib.php, kedze iba tree.php ju
pouziva.
      </change>
    </record>

    <record who="dalxo" when="2004-05-02 23:45" curr_ver="0.7"
prev_ver="0.6">
      <change desc="nove funkcie add_to_comma_list() a
remove_from_comma_list()">
          Comma list == 12,23,45
          add_to_comma_list(): prida polozku ak zatial v nej
neexistuje.
          remove_from_comma_list(): odstrani (ak existuje) polozku
zo zoznamu.
      </change>
    </record>

    <record who="dalxo" when="2004-05-03 15:00" curr_ver="0.8"
prev_ver="0.7">
      <change desc="premiestnenie niektorych funkcii z tree.php">
          tree_delete_node() a set_deleted_tree_nodes() sa
premiestnili z tree.php,
          pretoze su pouzivane del_module_from_tree().
      </change>
    </record>

  </file>

  <file filename="tree.php" module="znalosti" language="php"
since="2004-04-03">
    <description>
      Cesta: moodle/course/format.php
      Skript, ktory zobrazuje pohlady typu strom.
    </description>

    <record who="dalxo" when="2004-04-03 23:15" curr_ver="0.1"
prev_ver="0.0">
      <change desc="vytvorenie suboru">
          Zatial to dokaze len vypisat side_block-y a v strednom
stlpci vypisuje zatial len
          neasociovane moduly k uzlom stromu. V pripade editovanie,
da sa vymazat, hide, show,
          pridat do uzla (zatial nefunkcne) a vymazanie z pohladu.
      </change>
    </record>

    <record who="dalxo" when="2004-04-21 16:00" curr_ver="0.2"
prev_ver="0.1">
      <change desc="pridanie stromu">
          Pripojil sa k suboru JavaScript, ktory dokaze vypisat
strom. Vstupnym parametrom je
```

```
            nainicializovane pole. Pridanie funkcie print_tree(),
ktora necha vypisat strom vratane
            vyvolania vsetkych podpornych funkcii.
        </change>
        <change desc="pridanie array_PHP2JS()">
            Z PHP pola vygeneruje string, ktory inicilizuje v
JavaScripte pole s rovnakou strukturou.
        </change>
        <change desc="pridanie construct_tree()">
            K danemu uzlu vytvori strom ulozeny v PHP poli.
        </change>
        <change desc="set_deleted_tree_nodes()">
            Vyznaci v strome v DB, ktore uzly treba zmazat. Pouziva sa
preto, aby sa vyhlo problemom
            pri pristupe do DB a kvuoli elegantnejsiemu mazaniu, ked
sa potom nechaju zmazat vsetky
            uzly z pohladu s nastavenym priznakom.
        </change>
        <change desc="make_tree_editing_buttons()">
            Do stringu zapise HTML kod na vygenerovanie ovladacich
prvkov pre strom.
        </change>
    </record>

    <record who="dalxo" when="2004-05-02 22:00" curr_ver="0.3"
prev_ver="0.2">
        <change desc="modifikovanie construct_tree() a vytvorenie
find_sibling()">
            Teraz vie zobrazit uzly podla poradia.
        </change>

        <change desc="bugfix">
            Nedal sa pridat novy uzol (meno), ak boli vsetky moduly uz
priradene do stromu.
            Vlozila sa kontrola do JavaScriptu, ci objekt
reprezentujuci vyber je pole, alebo nie.
        </change>

        <change desc="zmena pouzivatelskeho rozrania">
            Spravila sa zmena ovladania stromu. Su tam dva select
skupiny.
            Jedna je na typ akcie a dalsia sluzi na spresnenie v
pripade pridavania
            uzla, ktora sa zobrazi iba pri vybere pridania uzla.
            Potom sa klikne na uzol na strome, nad ktorym sa ma
vykonat operacia.
            Vlozil som osetrovania hned do JavaScriptu, aby nedovolilo
            vymazat root, alebo editovat modul miesto editovania nazvu
bezmoduloveho uzla,
            posunutie uzla nahor, ktory je uz na vrchole a to iste pre
posunutie nadol.
            Musela sa zmenit aj obsluha delnode a addchild.
        </change>

        <change desc="pridanie JavaScript funkcii PerformAction,
ShowItems(), HideItems().">
```

```
          PerformAction sa vyvola vzdy pri kliknuti na uzol stromu.
Skontroluje sa, aka
          cinnost sa ma vykonat, ci je ju dovolene vykonat a ci ma
dodatocne parametre (pri
          pridavani dietata). Ak vsetko sedi, spravne vyplni
premenne, ktore su ako hidden a
          odosle ich ako POST.

          ShowItems sa vyvola pri kliknuti na Add child akciu.
Zvyditelni ponuku, co sa muoze
          este do stromu pridat.

          HideItems sa vyvolola pri vybere hocijakej inej akcie ako
je Add child a spuosobi
          zneviditelnenie ponuky na vyber elementov, ktore sa muozu
pridat do stromu.
      </change>

      <change desc="zmazanie JavaScript funkcii addnode() a
deletenode()">
          Kedze sa vytvorila spolocna funkcia PerformAction(), tieto
funkcie stratili zmysel.
      </change>

      <change desc="rozsirenie ovladania stromu o editovanie, posun
nahor a posun nadol">
          Pridala sa obsluha na editovanie nazvu bezmoduloveho uzla.
          Tak isto sa pridala podpora na posuvanie uzla nadol a
nahor v ramci urovne.
      </change>

      <change desc="pridanie funkcii is_last_node() a
is_first_node()">
          Uzitocne funkcie pri osetrovani submitnutych parametrov
pri presuvani uzla v ramci urovne.
          Tak isto su napomocne pri generovani zoznamu id uzlov,
ktore sa muozu posunut nahor a ktore
          nadol. Zoznam sluzi na strane pouzivatela, aby JavaScript
funkcia nedovolila pouzivatelovi
          zadat vykonanie operacii, ktore su bud nezmyselne, alebo
nedovolene.
      </change>

      <change desc="pridanie funkcie print_action_menu()">
          Vzhladom na zmenu pouzivatelskeho rozhrania sa pridala
funkcia, ktora vypise zoznam
          operacii, ktore sa daju vykonat nad uzlami stromu.
      </change>

      <change desc="premiestnena funkcia z lib.php :
print_modules_not_assoc_in_tree()">
          Tato funkcia je vyvolavana iba z tree.php a preto nema
zmysel ju udrziavat
          v lib.php.
      </change>
```

```
      <change desc="zrusenie niektorych side_block-ov">
          Zrusil som side_block-y F.A.Q., najblizsie udalosti a
kalendar.
      </change>

    </record>

    <record who="dalxo" when="2004-05-03 01:00" curr_ver="0.31"
prev_ver="0.3">
      <change desc="cistenie kodu">
          Pre vecsiu citatelnost som kod rozbil na tieto funkcie:
          tree_add_node(), tree_delete_node(), tree_moveup_node(),
tree_movedown_node().
      </change>
    </record>

    <record who="dalxo" when="2004-05-03 11:45" curr_ver="0.32"
prev_ver="0.31">
      <change desc="bugfix">
          V JavaScripte, ktory vytvaral pole sa argument vlozil do
uvodzoviek.
          Ak totiz pole ma iba jeden provok a bolo bez uvodzoviek,
tak
          argument sa bral ako velkost pola a nie ako prvy prvok
pola.
      </change>
    </record>

    <record who="dalxo" when="2004-05-03 15:00" curr_ver="0.4"
prev_ver="0.32">
      <change desc="premiestnenie niektorych funkcii do lib.php">
          tree_delete_node() a set_deleted_tree_nodes() sa
premiestnili do lib.php,
          pretoze su pouzivane del_module_from_tree().
      </change>
    </record>

    <record who="dalxo" when="2004-05-08 23:00" curr_ver="0.41"
prev_ver="0.4">
      <change desc="bugfix vo vytvarani pola v JavaScripte">
          Este raz to bolo treba opravit, uz je to urcite OK.
      </change>
    </record>

  </file>


  <file filename="moodle.php" module="znalosti" language="php"
since="2004-03-21">
    <description>
      Cesta: moodle/lang/en/moodle.php
      Anglicky slovnik na preklad sprav do anglictiny.
    </description>

    <record who="dalxo" when="2004-03-21 18:00" curr_ver="0.1"
prev_ver="0.0">
```

```
      <change desc="pridanie novych vyrazov"></change>
    </record>

    <record who="dalxo" when="2004-03-28 21:00" curr_ver="0.2"
prev_ver="0.1">
      <change desc="pridanie novych vyrazov"></change>
    </record>

    <record who="dalxo" when="2004-04-03 23:15" curr_ver="0.3"
prev_ver="0.2">
      <change desc="pridanie novych vyrazov">V suvislosti s
vytvorenim suboru tree.</change>
    </record>

    <record who="dalxo" when="2004-05-03 01:00" curr_ver="0.4"
prev_ver="0.3">
      <change desc="pridanie novych vyrazov">V suvislosti s
dokoncenim stromu.</change>
    </record>

  </file>

</file_changelog>
```

D.2 Odovzdávanie zadaní

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>
<!DOCTYPE file_changelog SYSTEM
"http://www2.dcs.elf.stuba.sk/TeamProject/2003/team10/DTD/file_chang
elog.dtd">

<file_changelog>

  <file filename="lib.php" module="assignment" language="php"
since="2004-03-16">
    <description>
      Kniznica funkcii modulu odovzdavania zadani.
    </description>

    <record who="kexo" when="2004-03-16 23:30" curr_ver="0.1"
prev_ver="0.0">
      <change desc="pridanie funkcie pre vytvorenie nazvu adresara
pre dalsiu verziu zadania">Nova funkcia
assignment_file_area_name_timestamp($assignment, $user,
$timestamp)</change>
      <change desc="pridanie funkcie pre vytvorenie adresara pre
dalsiu verziu zadania">Nova funkcia
assignment_file_area_timestamp($assignment, $user,
$timestamp)</change>
      <change desc="pridanie funkcie pre ziskanie vsetkych verzii
zadania studenta">Nova funkcia
assignment_get_submissions($assignment, $user)</change>
```

```
      <change desc="pridanie funkcie pre ziskanie cesty k suboru s
poslednou verziou zadania studenta">Nova funkcia
assignment_get_submission_file_last($assignment, $user)</change>
    </record>

    <record who="kexo" when="2004-03-20 21:00" curr_ver="0.2"
prev_ver="0.1">
      <change desc="modifikacia vypisu zadania - vypis konkretneho
zadania">Funkcia assignment_print_submission</change>
      <change desc="pridanie funkcie pre vypis suborov verzie
zadania">Nova funkcia
assignment_print_user_files_submission($assignment, $user,
$submission)</change>
    </record>

    <record who="kexo" when="2004-03-28 18:30" curr_ver="0.3"
prev_ver="0.2">
      <change desc="doplnenie zistenia a vypisu vysledku testovania
funkcnosti">Funkcia assignment_print_submission</change>
      <change desc="pridanie funkcie pre spustenie testu funkcnosti
a vratenie vysledkov">Nova funkcia
assignment_functionality_test($assignment, $user,
$submission)</change>
      <change desc="pridanie funkcie pre ziskanie cesty k suboru
lubovolneho zadania studenta">Nova funkcia function
assignment_get_submission_file($assignment, $user,
$submission)</change>
    </record>

     <record who="kexo" when="2004-03-29 03:00" curr_ver="0.4"
prev_ver="0.3">
       <change desc="pridanie funkcie pre zaradenie zadania do testu
plagiatorstva">Nova funkcia
assignment_plagiarism_test_file($assignment, $user,
$submission)</change>
       <change desc="pridanie funkcie pre spustenie testu
plagiatorstva">Nova funkcia
assignment_plagiarism_test_start($assignment)</change>
       <change desc="pridanie funkcie pre ziskanie vysledku testu
plagiatorstva">Nova funkcia
assignment_plagiarism_test_result($assignment)</change>
     </record>

     <record who="kexo" when="2004-03-29 03:45" curr_ver="0.5"
prev_ver="0.4">
       <change desc="zmena parametra zasielanemu demona z casu
odovzdania na id verzie zadania">zmena parametra v rozhrani s
demonom</change>
       <change desc="pridanie funkcie pre poskytnutie globalneho
nazvu adresara zadania predmetu">Nova funkcia function
assignment_dir_area($assignment)</change>
       <change desc="pridanie funkcie pre poskytnutie plnej cesty k
suboru vybraneho zadania">Nova funkcie
assignment_get_submission_file_fullpath($assignment, $user,
$submission)</change>
     </record>
```

```
      <record who="kexo" when="2004-03-30 02:00" curr_ver="0.6"
prev_ver="0.5">
        <change desc="uprava funkcie pre ziskanie objektu zadania -
doplnenie parametra ID zadania, pri jeho nezadani sa vrati posledne
zadanie">Funkcia assignment_get_submission($assignment,
$user)</change>
        <change desc="pridanie funkcie pre poskytnutie relativnej
cesty ku globalnemu adresaru zadania">Nova funkcia function
assignment_dir_area_name($assignment)</change>
      </record>

  </file>

  <file filename="view.php" module="assignment" language="php"
since="2004-03-16">
    <description>
      Zobrazovanie jednotlivych obrazoviek suvisiacich s modulom
zadani.
    </description>

    <record who="kexo" when="2004-03-16 23:30" curr_ver="0.1"
prev_ver="0.0">
      <change desc="doplnenie vypisu vsetkych verzii zadani pre
studenta i ucitela"/>
    </record>

    <record who="kexo" when="2004-03-27 20:30" curr_ver="0.2"
prev_ver="0.1">
      <change desc="bugfix">Oprava cislovania poradia
zadani</change>
      <change desc="bugfix">Oprava chybovej hlasky ak este student
neodovzdal zadanie</change>
    </record>

      <record who="kexo" when="2004-03-29 04:50" curr_ver="0.3"
prev_ver="0.2">
        <change desc="pridanie zobrazovania zoznamu testov
plagiatorstva pre ucitela"/>
      </record>

    <record who="kexo" when="2004-03-30 01:40" curr_ver="0.4"
prev_ver="0.3">
      <change desc="pridanie moznosti mazat neohodnotene a
neokomentovane zadanie"/>
      <change desc="doplnena moznost zobrazovania vysledkov testu
plagiatorstva pre ucitela"/>
    </record>

      <record who="kexo" when="2004-03-31 01:25" curr_ver="0.41"
prev_ver="0.4">
        <change desc="bugfix - pri navrate zo zoznamu odovzdanych
zadani na stranku zadania boli chybne generovane linky na vysledky
testov plagiatorstva">bugfix pri navigacii medzi strankami</change>
      </record>
```

```
      <record who="kexo" when="2004-04-16 14:50" curr_ver="0.42"
prev_ver="0.41">
        <change desc="bugfix - po spusteni testu plgiatorstva sa v
zozname testov zobrazoval vysledok testu aj informacia, ze prave
prebieha">bugfix pri vypise zoznamu testov plagiatorstva</change>
      </record>
  </file>

  <file filename="upload.php" module="assignment" language="php"
since="2004-03-16">
    <description>
      Skript pre upload zadania na server.
    </description>

    <record who="kexo" when="2004-03-16 23:30" curr_ver="0.1"
prev_ver="0.0">
      <change desc="uprava ukladania zadani - podpora verzii
prostrednictvom casovych peciatok"/>
    </record>

  </file>

  <file filename="mod.html" module="assignment" language="php"
since="2004-03-16">
    <description>
      Skript pre upload zadania na server.
    </description>

    <record who="kexo" when="2004-03-16 23:30" curr_ver="0.1"
prev_ver="0.0">
      <change desc="doplnenie parametra pre nastavenie moznosti
viacerych verzii zadania"/>
    </record>

      <record who="kexo" when="2004-03-30 01:10" curr_ver="0.2"
prev_ver="0.1">
        <change desc="doplnenenie parametra pre nastavenie moznosti
mazania zadania"/>
      </record>

  </file>

  <file filename="submissions.php" module="assignment"
language="php" since="2004-03-16">
    <description>
      Skript pre zobrazenie zoznamu zadani pre ucitela.
    </description>

    <record who="kexo" when="2004-03-20 21:00" curr_ver="0.1"
prev_ver="0.0">
      <change desc="prednastavene zoradenie zadani podla mena
studentov"/>
      <change desc="doplnenie zobrazenia vsetkych verzii zadania pre
kazdeho studenta"/>
    </record>
```

```
    <record who="kexo" when="2004-03-27 19:40" curr_ver="0.2"
prev_ver="0.1">
      <change desc="prednastavene zobrazenie len poslednych verzii
zadani"/>
      <change desc="doplnenie moznosti zobrazit vsetky verzie zadani
studentov"/>
    </record>

     <record who="kexo" when="2004-03-28 21:00" curr_ver="0.3"
prev_ver="0.2">
       <change desc="zmena tlacitka 'save all my feedback' na 'save
feedback and run test'"/>
       <change desc="doplnenie moznosti hromadneho
zaradenia/vyradenia zadani do testov"/>
     </record>

     <record who="kexo" when="2004-03-29 03:15" curr_ver="0.4"
prev_ver="0.3">
       <change desc="doplnenie spustania testov funkcnosti a
plagiatorstva"/>
       <change desc="doplnenie logovania spustenia testov"/>
     </record>

     <record who="kexo" when="2004-04-16 14:55" curr_ver="0.5"
prev_ver="0.4">
       <change desc="doplnenie formulara pre zadavanie vstupnej a
vystupnej vzroky pre test funkcnosti"/>
     </record>

  </file>

  <file filename="assignment.php" module="lang" language="php"
since="2004-03-16">
    <description>
      Jazykove modifikacie textov moodle. Subory sa nachadzaju v
adresaroch moodle/lang/[jazyk]/.
      Modifikovane boli nasledovne verzie: anglicka (en), slovenska
(sk), ceska (cz).
    </description>

    <record who="kexo" when="2004-03-16 23:30" curr_ver="0.1"
prev_ver="0.0">
      <change desc="doplnenie textov pre vypisy suvisiace s podporou
viacerych verzii zadania"/>
    </record>

    <record who="kexo" when="2004-03-27 19:40" curr_ver="0.2"
prev_ver="0.1">
      <change desc="doplnenie textov pre typ zobrazenia zoznamu
zadani"/>
    </record>

    <record who="kexo" when="2004-03-28 18:10" curr_ver="0.3"
prev_ver="0.2">
      <change desc="doplnenie textov pre testy funcknosti"/>
      <change desc="doplnenie textov pre test plagiatorstva"/>
```

```
    </record>

    <record who="kexo" when="2004-03-30 01:15" curr_ver="0.4"
prev_ver="0.3">
      <change desc="doplnenie textov pre nastavenie moznosti mazat
zadanie"/>
    </record>

    <record who="kexo" when="2004-04-16 14:55" curr_ver="0.5"
prev_ver="0.4">
      <change desc="doplnenie textov pre testovanie funkcnosti"/>
    </record>
  </file>

  <file filename="mysql.*" module="assignment" language="php"
since="2004-03-16">
    <description>
     mySQL skript pre update tabuliek modulu assignment podla danej
verzie
    </description>

    <record who="kexo" when="2004-03-16 23:30" curr_ver="0.1"
prev_ver="0.0">
      <change desc="doplnenie priznaku pre nastavenie moznosti
viacerych verzii"/>
    </record>

    <record who="kexo" when="2004-03-28 17:30" curr_ver="0.2"
prev_ver="0.1">
      <change desc="doplnenie priznaku prebiehajuce testovania
funkcnosti zadania">Tabulka assignment_submissions</change>
      <change desc="doplnenie stlpca pre uchovavanie vysledky
testovania funkcnosti zadania">Tabulka
assignment_submissions</change>
      <change desc="doplnenie priznaku prebiehajucej kontroly
plagiatorstva">Tabulka assignment</change>
    </record>

     <record who="kexo" when="2004-03-29 23:50" curr_ver="0.3"
prev_ver="0.1">
       <change desc="doplnenie priznaku pre nastavenie moznosti
mazania zadani">Tabulka assignment</change>
       <change desc="doplnenie textovych poli pre archivaciu
vstupnych a vystupnych vzoriek">Tabulka
assignment_submissions</change>
     </record>

  </file>

  <file filename="postgres7.*" module="assignment" language="php"
since="2004-03-28">
    <description>
     Postgres SQL skript pre vytvorenie tabuliek modulu assignment
    </description>
```

```
    <record who="kexo" when="2004-03-28 16:50" curr_ver="0.1"
prev_ver="0.0">
      <change desc="doplnenie stlpca pre nastavenie moznosti
viacerych verzii"/>
    </record>

    <record who="kexo" when="2004-03-28 17:30" curr_ver="0.2"
prev_ver="0.1">
      <change desc="doplnenie priznaku prebiehajuce testovania
funkcnosti zadania"/>
      <change desc="doplnenie stlpca pre uchovavanie vysledky
testovania funkcnosti zadania"/>
      <change desc="doplnenie priznaku prebiehajucej kontroly
plagiatorstva"/>
    </record>

     <record who="kexo" when="2004-03-29 23:50" curr_ver="0.3"
prev_ver="0.1">
       <change desc="doplnenie priznaku pre nastavenie moznosti
mazania zadani">Tabulka assignment</change>
       <change desc="doplnenie textovych poli pre archivaciu
vstupnych a vystupnych vzoriek">Tabulka
assignment_submissions</change>
     </record>

  </file>

  <file filename="config.php" module="system" language="php"
since="2004-03-28">
     <description>
          Konfiguracny subor moodle
     </description>

     <record who="kexo" when="2004-03-28 18:50" curr_ver="0.1"
prev_ver="0.0">
          <change desc="doplnenie nastaveni daemona">pridana sekcia
7</change>
     </record>

  </file>

  <file filename="moodlelib.php" module="system" language="php"
since="2004-03-29">
     <description>
          Kniznica systemovych funkcii moodle.
     </description>

     <record who="kexo" when="2004-03-29 05:20" curr_ver="0.1"
prev_ver="0.0">
          <change desc="pridana funkcia pre poskytnutie listingu
len suborov adresara">Nova funkcia get_directory_filelist($rootdir,
$excludefile="", $descend=false)</change>
     </record>

  </file>
```

```
  <file filename="viewplagtest.php" module="assignment"
language="php" since="2004-03-30">
      <description>
            Novy skript pre nacitanie a zobrazenie vysledkov testu
plagiatorstva.
      </description>

      <record who="kexo" when="2004-03-30 03:00" curr_ver="0.0"
prev_ver="">
            <change desc="vytvoreny novy subor a implementovana
zakladna funkcnost - nacitanie suboru s vysledkami a jeho
zobrazenie"/>
      </record>

      <record who="kexo" when="2004-03-30 11:30" curr_ver="0.1"
prev_ver="0.0">
            <change desc="implementovane zobrazenie vysledkov plag
testu v tabulke"/>
      </record>

      <record who="kexo" when="2004-03-31 01:20" curr_ver="0.2"
prev_ver="0.1">
            <change desc="doplnena moznost zobrazenia vysledkov v XLS
formate (MS Excel)"/>
      </record>
  </file>

  <file filename="index.php" module="assignment" language="php"
since="2004-04-19">
      <description>
            Zobrazenie zoznamu zadaní predmetu.
      </description>

      <record who="kexo" when="2004-04-19 18:00" curr_ver="0.1"
prev_ver="0.0">
            <change desc="doplnenie zobrazovania statusu demona"/>
      </record>

      <record who="kexo" when="2004-03-30 11:30" curr_ver="0.2"
prev_ver="0.1">
            <change desc="bugfix">Zakázanie zobrazovania statusu
demona pre studenta</change>
      </record>
  </file>

</file_changelog>
```

D.3 DTD pre tvorbu záznamov zmien v rámci modulu, resp. subsystému.

```
<!--

  version 3

-->
```

```
  <!ELEMENT file_changelog (file*)>
  <!ELEMENT file (description, record*)>
  <!ELEMENT description (#PCDATA)>
  <!ELEMENT record (change*)>
  <!ELEMENT change (#PCDATA)>
  <!--
       filename     : nazov suboru, ktoreho sa tyka tento changelog
       module       : ktoreho, prip. ktorych modulov sa to tyka
       language     : programovaci jazyk
       since        : kedy sme ho vytvorili, teda zacali prace na
implementacii
       finished     : kedy sa definitivne ukoncili prace na nom -
t.j. zme to zastavili, pripadne sme ho zrusili
  -->
  <!ATTLIST file
            filename      CDATA   #REQUIRED
            module        CDATA   #REQUIRED
            language      CDATA   #REQUIRED
            since         CDATA   #REQUIRED
            finished      CDATA   #IMPLIED
  >
  <!--
       when     : kedy sa vytvoril tento zaznam.
       curr_ver : nova (sucasna erzia)
       prev_ver : z ktorej ver. vznikla sucasna verzia
       who      : kto tento zaznam vytvoril(iba 1 z 5 moznosti ,
t.j. je zodpovedny sa spravne udaje v nom
  -->
  <!ATTLIST record
            when      CDATA   #REQUIRED
            curr_ver  CDATA   #REQUIRED
            prev_ver  CDATA   #REQUIRED
            who (dalxo | ico | chrco | hollub | kexo) #REQUIRED
  >
  <!--
       desc: kratky popis zmeny. Ak nestaci, muoze sa viac popisat v
tele elementu (vid priklad).
          Kratky popis je naschval povinny, pretoze musi byt jasne a
strucne povedane, co sa zmenilo (pri rychlom vyhladavani).
          Ostatne ako napr. implementacne deatily a vykecavacky
puojdu do toho tela elementu feature (ak treba).
  -->
  <!ATTLIST change
            desc  CDATA   #REQUIRED
  >
```

D.4 DTD pre tvorby záznamov zmien celého systému

```
<!--

   version 3
```

```
-->
  <!ELEMENT module_changelog (module*)>
  <!ELEMENT module (description, record*)>
  <!ELEMENT description (#PCDATA | file)*>
  <!ELEMENT record (feature*, not_affected_file*)>
  <!ELEMENT feature (#PCDATA | affected_file)*>
  <!ELEMENT affected_file (#PCDATA)*>
  <!ELEMENT not_affected_file (#PCDATA)*>
  <!ELEMENT file (#PCDATA)*>


  <!--
       module_name  : nazov modulu
       since        : kedy sme ho vytvorili, teda zacali prace na
implementacii
       finished     : kedy sa definitivne ukoncili prace na nom -
t.j. zme to zastavili, pripadne zrusili
  -->
  <!ATTLIST module
            name          CDATA   #REQUIRED
            since         CDATA   #REQUIRED
            finished      CDATA   #IMPLIED
  >

  <!--
       when       : kedy pribudol tento zaznam
       curr_ver   : nova verzia modulu
       prev_ver   : z ktorej verzie vznikla
       who        : kto vvytvoril tento zaznam (iba 1 z 5 moznosti ,
t.j. je zodpovedny sa spravne udaje v nom
  -->
  <!ATTLIST record
            when        CDATA   #REQUIRED
            curr_ver    CDATA   #REQUIRED
            prev_ver    CDATA   #REQUIRED
            who         (dalxo | ico | chrco | hollub | kexo)
#REQUIRED
  >

  <!--
       short_desc : kratky popis zmeny. Ak nestaci, muoze sa viac
popisat v tele elementu (vid priklad).
                    Kratky popis je naschval povinny, pretoze musi
byt jasne a strucne povedane, co sa zmenilo (pri rychlom
vyhladavani).
                    Ostatne ako napr. implementacne deatily a
vykecavacky puojdu do toho tela elementu feature (ak treba).

  -->
  <!ATTLIST feature
            desc  CDATA   #REQUIRED
  >


  <!--
```

```
       id           : referencia na subor (vid nizsie element file a
priklad)
       current_ver  : sucasna verzia suboru
       prev_ver     : predchadzajuca verzia (mozno nebude treba a je
zbytocny)
                      Tieto informacie o verzii sa tu uvadzaju preto
(aj ked nejake info o verzii suborov su v metainfo), lebo kazdy
subor
                      ma vlastny proces vyvoja a zmena v jeho vnutri
nemusi ovplynit cely modul. Napr. ak sa prerobi nejaka
                      funkcia, aby fungovala rychlejsie, alebo sa
niektore funkcie v subore viac modularizuju, teda vznikne viacej
mensich funkcii.

        Takto budeme vediet s akou verziou suboru sa pracuje (ak sa
zmeni).
        Budeme mat aj jasno, v zodpovednosti jednotlivych suborov na
novej featuriji.
  -->
  <!ATTLIST affected_file
            id            IDREF   #REQUIRED
            current_ver   CDATA   #REQUIRED
            prev_ver      CDATA   #REQUIRED
  >

  <!--
        Tento element je dobry na to, aby sme mali prehlad aj o tych
suborov, ktore sa nemodifikovali (teda neboli vo feature)
        ale chceme vediet verzie suborov tvoria tuto verziu. Ako som
spominal, subory sa muozu aj vnutorne menit bez toho, aby
        to malo za nasledok zmenu verzie modulu. Toto je dobre na
to, aby ked sa spravi vnutorna zmena subora a urobila sa v tom
        chyba a teraz zistime, ze nefunguje ani modul, aby sme hned
vedeli, ake verzie suborov sme pouzili.

        Samozrejme, ak sme zmenili vsetky subory co su ako file
element potom not_affected_file sa neuvadza (vid priklad nizsie).

       id           : referencia na subor (vid nizsie element file a
priklad)
       current_ver  : pouzita verzia suboru
       prev_ver     : predchadzajuca verzia (mozno nebude treba a je
zbytocny)

  -->
  <!ATTLIST not_affected_file
            id            IDREF   #REQUIRED
            current_ver   CDATA   #REQUIRED
            prev_ver      CDATA   #REQUIRED
  >

  <!--

      id        : identifikator, v ramci XML musi mat jedinecny
nazov (v ramcy vsetkych atributov typu ID). XML nevie semanticky
```

```
                  rozlisit napr. ci vo vasom atribute IDREF alebo
IDREFS sa miesto id elementu auto odkazujete na od elementu person.
                  XML vielen zabezpecit, ze ID bude jedinecne v
celom XML subore.

      name      : cesta a nazov suboru
      since     : odkedy je pridany do modulu
      since_ver : od ktorej verzie bol pridany do modulu
      short     : strucny, alebo kratky popis suboru (nepovinne).
                  Ak chcete popisat k suboru viac, takto muozete v
ramci ementu file -vid. example.

      Poviete si, ze naco je atribut id a aj name, ked sa rovnaju.
To nemusi byt zase pravda uplne. Preco?
      Lebo:
        Chcem, aby bolo jasne, dole v tych zaznamoch pri zmene
verzie, ktore subory to mali na svedomi.
        XML-ko podporuje identifikatory a referencie na ne, lenze
tie identifikatory a referencie (teda ID a odkaz na nu IDREF)
        musia byt "XML name".
        "XML name" znamena, ze lubovolne pismeno, cislica a len
tieto znaky: -, _ a . su povolene.
        Ako vidite, tak tam nie je \ a ani /. Preto v atribute name
bude cesta a meno suboru, teda nazov file a v id atribute bude
        "prijatelna" forma mena, aby sa dalo v ramci XML
referencovat.
  -->
  <!ATTLIST file
            id          ID      #REQUIRED
            name        CDATA   #REQUIRED
            since       CDATA   #REQUIRED
            since_ver   CDATA   #REQUIRED
            short       CDATA   #IMPLIED
  >
```

Príloha E - Obsah elektronického media

`prototype/`
: Prototyp systému.

`zapisnice/`
: Zápisnice s regulárnych stretnutí.

`docs/`
: Všetky dokumenty vygenerované naším tímov vrátane technickej dok. a dok. k riadeniu.

`foto/`
: Rôzne fotky z rôznych udalostí na rôzne témy.

`img/`
: Obrázky potrebné pre vygenerovanie web stránky.

`dtd/`
: DTD pre changelogy subsystémov a celeho systému.

`product/`
: Súbory produktu

`changelogs/`
: Changelogy subsystémov a celeho systému.