České vysoké učení technické v Praze
Fakulta elektrotechnická

Diplomová práce

E-learningový systém pro podporu výuky algoritmů

Roman Hocke

Vedoucí práce: Mgr. Petr Matyáš

Studijní program: Elektrotechnika a informatika strukturovaný magisterský

Obor: Informatika a výpočetní technika

květen 2008

Poděkování

Děkuji svému vedoucímu Mgr. Petru Matyášovi především za obětavý přístup při konzultacích mojí práce.

Prohlášení

Prohlašuji, že jsem svou diplomovou práci vypracoval samostatně a použil jsem pouze podklady uvedené v přiloženém seznamu.
Nemám závažný důvod proti užití tohoto školního díla ve smyslu §60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne 27.4. 2008 .....

Abstract

The goal of the work was to implement e-learning solution for teaching specific courses of Department of Computer Scienece and Engineering at FEE CTU in Prague. I compared advantages and disadvantages of existing web-based e-learning systems and considered features needed for using such system at the Department and implementing these features in one of the open-source systems. Finally I tested the solution and filled it with sample data.

Abstrakt

Cílem práce byla implementace e-learningového řešení výuky vybraných předmětů na Katedře počítačů FEL ČVUT. Ve své práci jsem srovnal výhody a nevýhody již existujících webových e-learningových systémů a zamýšlel jsem se nad rysy, potřebnými pro nasazení takového systému na Katedře počítačů a jejich implementací pod některým z existujících open-source systémů. V rámci práce byl systém důkladně otestován a naplněn zkušebními daty.

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Cílem této práce je implementace e-learningového systému pro výuku předmětů zaměřených na teoretickou informatiku a podobné obory. Snaží se stanovit a shrnout požadavky na takový systém, poté zvolit vhodné existující open-source e-learningové řešení a v něm nové požadavky implementovat.
V posledním desetiletí jsme svědky rychlého a masivního rozšiřování informačních technologií, a to jak v oblastech odborných, tak i v běžném, každodenním životě. Informační technologie nám dnes pomáhají takřka ve všech vědních oborech a není tedy divu, že pronikají i do oblasti vzdělávání. Velké množství informací lze dnes nalézt v elektronické podobě (například na internetu) a jistě není náhoda, že toto platí obzvláště v oboru informatiky a programování (programátoři mají k elektronické podobě technických materiálů a dokumentací nejblíž, často ani jinou formu dokumentace neprodukují).
Nalézt odpovědi na otázky z technických oborů není dnes velký problém, často postačí použití kvalitního vyhledávače informací na internetu. Nalezené informace však často mají ryze technický, dokumentační charakter, který může zcela vyhovovat člověku, který si pouze doplňuje nebo rozšiřuje informace z oboru, ve kterém již má dostatečné znalosti na to, aby nové informace pochopil. Často jsou však tyto materiály méně schůdné pro lidi, kteří jsou v daném oboru nováčky a chtějí do něj teprve proniknout. I těmto lidem popisovaný druh informací může pomoci, jejich cesta k poznání však může být zdlouhavá a krkolomná.
Řešení tohoto problému nabízí e-learning, tedy elektronická forma učení, konkrétně jeho webová podoba. Slučuje v sobě výhodu elektronické podoby učiva a přednosti pedagogického přístupu k předávání informací (oproti „obyčejné“ technické dokumentaci). Předpokladem je samozřejmě kvalitní příprava výukového kurzu. Mezi výhody (webové formy) e-learningu patří:

- snadná dostupnost výukového kurzu studentům,
- pedagogický přístup – student je učivem metodicky provázen (oproti „obyčejné“ technické dokumentaci),
- snadná možnost aktualizace výukových materiálů (např. při zjištění chyby nebo při nutnosti rozšířit obsah),
- možnost ověření svých znalostí formou testů, kvízů a podobně,
- využití interaktivních a názorných prostředků (animací, interaktivních prostředků např. v podobě Java appletů, multimédií).

E-learning je dnes v ČR (a nejen zde) nasazován v různých oblastech. Od komerčních řešení nasazovaných v různých firmách na školení zaměstnanců (to zahrnuje mimo jiné školení řidičů i s testy na příslušné certifikáty, školení o bezpečnosti a ochraně zdraví při práci (BOZP) nebo např. školení o protipožární ochraně) přes odborné kurzy (např. kurzy AutoCADu) až po výukové kurzy na některých univerzitách. Několik příkladů za všechny:

- Ostravská univerzita (`http://moodle.osu.cz/`),

- Česko-moravská konfederace odborových svazů (`http://www.cmkos.cz/lidske-zdroje_nove/e-learning`),
- Filosofická fakulta Masarykovy univerzity (`http://www.phil.muni.cz/elf/`),
- Centrum pro otázky životního prostředí (`http://www.czp.cuni.cz/glob/`).

Cílem této práce je zvolit a zprovoznit e-learningový systém pro Katedru počítačů FEL ČVUT a přizpůsobit tento systém specifickým potřebám pro výuku teoretické informatiky, algoritmů a příbuzných odvětví počítačové vědy.
V práci se budu zabývat otázkou co vlastně e-learning je a v čem může spočívat jeho přínos pro Katedru počítačů. Nasazení e-learningu budu chápat především jako doplněk stávající výuky, nikoli její úplnou náhradu. Také budu klást důraz na fakt, že výuka teoretické informatiky a příbuzných témat, klade na e-learningové řešení některé specifické nároky, jako například:

- schopnost zapisovat a zobrazovat složitější matematické a maticové zápisy,
- možnost demonstrovat principy a algoritmy pomocí interaktivních prvků (animací, simulací).

1 Charakteristika e-learningu

1.1 Historie e-learningu

Jedna z prvních moderních metod výuky se objevila za druhé světové války. Tehdy byl jako výukový materiál použit film. Filmy sloužily k rychlé výuce základních vojenských dovedností, jakou byla například údržba zbraní, rozpoznávání letadel a podobně. Úspěch těchto filmů vedl ke spolupráci armády s univerzitami a k vedení výzkumu, který postupem času vedl k vývoji e-learningu.
Ve druhé polovině šedesátých let se začalo experimentovat se stroji na učení. Začalo se jim říkat vyučovací automaty. I u nás byl jeden vyvinut, jmenoval se Unitutor. Vykládaná látka byla v Unitutoru rozdělena na jednotlivé stránky, na konci stránky se nacházela kontrolní otázka s výběrem z několika možných odpovědí. Podle provedené volby bylo možné program dále větvit a pokračovat v libovolné další stránce. Informace o správném či chybném řešení představovala okamžitou zpětnou vazbu. Vyučovací automaty však byly příliš složité a ne moc účinné, proto se neujaly.
V druhé polovině osmdesátých let dvacátého století se objevují první šestnáctibitové počítače, trh ovládají osobní počítače (PC). Zároveň s tím můžeme sledovat obrovský rozmach kancelářských aplikací. Počítače se konečně začínají objevovat i v domácnostech. Ve školství dochází v souladu s celosvětovým vývojem kybernetiky a umělé inteligence k pokusu o zdokonalení vyučovacích automatů. Počítač se začíná používat jako učící a zkoušející stroj. Za pomoci počítače se začínají prověřovat teorie, které tvrdí, že by počítač měl částečně nahradit učitele.
Ve světě začalo několik (převážně univerzitních) vědeckých týmů vyvíjet inteligentní výukové systémy (Intelligent Tutoring Systems). Cílem těchto výukových systémů bylo vytvářet aplikace s dlouhodobou kontrolou nad výukovým procesem. Systémy v sobě vhodně spojovaly výklad učiva, procvičování probrané látky a testy. Dokázaly využívat grafiku, animaci, zvuk a byly schopny v sobě integrovat i zcela nezávislé programy. Tempo i obsah výuky byl individualizován. Dosažené výsledky studujícího se ukládaly a vyhodnocovaly. Tím se automaticky rozhodovalo o dalším postupu. Role učitele se omezila na kontrolu a obsluhu.
Postupem času se k testu se přidával výklad látky a procvičování. Z těchto prvků byly sestavovány jednotlivé lekce a z nich pak celé kurzy. Postup studentů byl individualizován a řídil se jejich výsledky. To ale znamenalo, že počítač musel předvídat všechny možné reakce studenta a situace, do kterých se mohl studující během práce dostat. Princip umělé inteligence u výukových programů spočívá ve vytvoření určitého modelu umělého studenta, na kterém je funkce programu založena.
Vývoj na univerzitách pokračoval rychle kupředu. Sylaby, knihovní zdroje, obsahy přednášek začaly být přemísťovány z klasických učeben na multimediální zdroje a na místní sítě. Soukromé společnosti začaly hledat možnosti potencionálního e-learningu. Na webu vznikly virtuální univerzity, které nabízely všechny své kurzy a získání certifikátů přes internet. Koncem devadesátých let již e-learningové nástroje umožňovaly zkoušení on line v reálném čase, hry v reálném čase, pomocí nástrojů bylo možné okamžitě určit slabosti a silné stránky jednotlivých studentů. Student tak mohl získat vysokoškolský titul, aniž by byl někdy fyzicky přítomen ve třídě. Plně zaměstnaní dospělí mohli studovat na vysoké škole svým vlastním tempem bez toho, aby museli řešit problémy spojené se svou fyzickou přítomností ve škole.

Pro další podrobné informace viz [7] nebo [9].

1.2 Používané pojmy

- *Tutor* je osoba, která studentovi zprostředkovává e-learningový kurz. Pomáhá mu vzdálenou formou při práci s výukovými materiály, hodnotí studentovu práci a motivuje ho k dalšímu studiu. Pomáhá při výběru kurzu, konzultuje jeho obsah a přijímá připomínky či náměty na obsah kurzu. Tutor nemusí být osoba, která vede prezenční studium.

- *Výukový kurz* je souhrn více výukových objektů (např. textů, animací, testů) na různá témata, která se všechna vztahují k probírané látce. Jeho posláním je zprostředkovat studentovi postupně a systematicky znalosti z dané problematiky a pomoci mu získat určité dovednosti.

- *CMS, Content Management System* neboli *systém pro správu obsahu* je software, který zajišťuje správu dokumentů (obvykle textů, obrázků, dalších materiálů). Zpravidla se jedná o webové nástroje, které slouží jako redakční či publikační systémy pro správu webového obsahu.

- *LMS, Learning Management System* je software, který v sobě zahrnuje funkcionalitu potřebnou pro elektronickou formu vzdělávání. Obsahuje jak nástroje pro studenta (prohlížení výukových materiálů, podstupování testů a kvízů, fórum, chat...), tak i nástroje zjednodušující samotnou tvorbu kurzů učitelem. Tyto systémy zpravidla fungují online, umožňují tedy kromě samotného učení z materiálů také komunikaci mezi studenty a tutory. Za běžné funkce LMS se považuje správa tříd a žáků, správa výukových kurzů a témat v rámci kurzů, testování a zkoušení studentů, nástroje pro komunikaci mezi žáky a studenty a další.

1.3 Klíčové vlastnosti

Podle [12] si studenti zapamatují jen na krátkou dobu informaci, kterou pouze slyší, uchovají si 40% informací, které vidí a slyší a uchovají si 75% informací, které vidí a slyší a také si ji interaktivně ověří (vyzkouší). Elektronická podoba výuky nabízí studentovi možnost si probíranou látku ihned vyzkoušet, ať už prostřednictvím interaktivních simulací či krátkého opakovacího testu, který studentovi okamžitě poskytne zpětnou vazbu, například s detailnějším vysvětlením nesprávně zodpovězené otázky.
Z toho mimo jiného také vyplývá, že e-learning by měl těžit ze své možnosti podávat látku zábavně a zajímavě, měl by se tedy vyhýbat dlouhým, monolitickým textům a měl by spíše využívat atraktivní a heslovitější formy (viz např. [1]).
Za důležitou také považuji studentovu možnost vyzkoušet si souhrnné znalosti z celého výukového kurzu. Kurz se skládá z jednotlivých témat a student si pomocí testu ověří, kterým tématům rozumí a kterým ne, zjistí tak, která témata by si měl zopakovat například před závěrečnou zkouškou.

1.4 Výhody a nevýhody zavedení e-learningu

Proč nasazovat e-learning na škole, kde výuka probíhá primárně formou přednášek a pravidelných cvičení? Důvodů může být několik.

- Možnost okamžitého vyzkoušení právě probírané látky může studentovi nabídnout právě e-learning s použitím různých virtuálních nástrojů, interaktivních simulací a podobně. Možnost vyzkoušet si fungování věcí v praxi samozřejmě nabízí i klasická školní cvičení, to však není možné vždy, například nemusí být pro danou demonstraci dostupné vybavení. Při klasickém cvičení je student také časově omezen.

- Elektronické řešení umožňuje využití hypertextu, tedy odkazů do jiných kapitol, případně odkazů na další (vnější) zdroje informací. Umožňuje také spravovat obsáhlý slovník často používaných pojmů a v učebním textu se mohou vyskytovat odkazy přímo na krátká vysvětlení těchto pojmů místo odkazů do jiných kapitol, kde jsou tyto pojmy vysvětlovány příliš detailně.

- Elektronická forma umožňuje atraktivnější prezentaci probírané látky, například používání barev, zvýrazňování klíčových slov, používání obrázků, animací, zvukových nahrávek a videozáznamů.

- E-learning je z principu prostředkem k distančnímu vzdělávání, může tedy najít své uplatnění u dálkové nebo kombinované formy studia na VŠ.

E-learning tedy má některé nesporné výhody. Jaké však mohou být jeho nevýhody? Jaká úskalí s sebou přináší?

- Vysoké nároky na kvalitní přípravu výukového kurzu. E-learningový výukový kurz by měl plně využívat možností elektronické formy. Měl by být poutavý a motivovat studenta – na začátku tématu studentovi oznámit, co se nového naučí, jaké vědomosti na konci tématu získá, jaké dovednosti si osvojí. Měl by se vyhýbat dlouhým, jednotvárným odstavcům textu. Tato práce si neklade za cíl přesně formulovat pravidla a zásady pro vytváření e-learningových kurzů, s touto problematikou se může čtenář blíže seznámit například ve videozáznamu [6] nebo v [5] či [2].

- Pokud student látku nepochopí, má při klasické formě výuky (přednáška, cvičení) možnost požádat vyučujícího, aby zkusil látku vyložit jinak nebo může vhodnými dotazy docílit pochopení látky. Při použití e-learningu se sice může student obrátit na tutora nebo na ostatní studenty (pomocí diskuzního fóra, chatu…), avšak odpovědi nebo vysvětlení se mu nemusí dostat hned, ale až například druhý den.

- Některé zdroje (např. [3]) se zabývají myšlenkou, že e-learning by mohl kromě některých svých úskalí představovat dokonce hrozbu:

  > Při nadměrném používání počítačů mizí běžný společenský život, který člověk nutně potřebuje. Snižuje se míra sociability v procesu poznávání a vzdělávání.

  To může jistě představovat závažný problém, nicméně cílem této práce není *nahradit* tradiční výuku na ČVUT e-learningem, ale *doplnit* ji. Běžný „akademický společenský život“ tedy nevymizí.

1.5 Standardy používané v e-learningu

Ve světě e-learningu bylo vytvořeno mnoho standardů, z nichž některé se ujaly a běžně se dnes používají, jiné zanikly nebo nejsou dostatečně rozšířené. V oblasti e-learningu byly vytvořeny především standardy popisující strukturu výukových materiálů, typy a složení testů a testových otázek, různé způsoby jejich bodování či hodnocení a podobně. Používání těchto standardů je výhodné kromě jiného proto, že umožňuje přenos dat (výukových materiálů, testů, výsledků testů...) mezi různými výukovými systémy (ať už webovými, desktopovými atd.). Následující dva standardy patří v e-learningu k nejpoužívanějším.

1.5.1 SCORM

Sharable Content Object Reference Model je referenční model pro e-learning. Je souborem specifikací a standardů, jejichž hlavním úkolem je, aby umožnily provozovat obsah vytvořený v souladu se SCORMem v libovolném LMS, který také musí pravidlům SCORM vyhovovat. Jak vyplývá z názvu, jde o model sdílitelných obsahových objektů (SCO), který umožňuje znovupoužití vzdělávacích materiálů na všech SCORMu přizpůsobených produktech a platformách. Pro popis výukových objektů SCORM používá manifest. Je to popisný soubor napsaný v jazyku XML. Aplikační profil metadat popisujících SCORM objekty má 64 prvků, ale jen malá část z nich je povinně vyžadována pro dosažení shody s referenčním modelem. Tento model je vytvářen americkou iniciativou ADL (Advanced Distributed Learning Initiative) a odvolává se na normy vytvořené konzorcii IEEE a IMS Learning Technology Standards. Viz `http://en.wikipedia.org/wiki/SCORM`.

1.5.2 AICC

Aviation Industry Computer-Based Training Committee. Jde o standard vytvořený v roce 1988 velkými výrobci letadel. Vznikl ze zájmu o využití multimédií ve výuce pilotů a leteckého personálu. Standard není založen na XML a údajně je mnohými považován za bezpečnější a spolehlivější než SCORM. Je využívám řadou e-learningových systémů a produktů (např. Lectora). Viz `http://en.wikipedia.org/wiki/AICC\_(CBT)`

2 Srovnání existujících řešení

2.1 Požadavky na e-learningové řešení

Při zkoumání vlastností jednotlivých LMS považuji za důležité především tyto vlastnosti:

- Systém umožňuje studentovi pomocí správně vytvořených testů ověřit si své znalosti.
- Tyto testy nemají za cíl studenta hodnotit, jejich účelem má být pouze zpětná vazba -– student zjistí, čemu nerozumí a test mu na základě nesprávných odpovědí doporučí kapitoly, které by si měl znovu projít, případně mu objasní důvod správných řešení a podobně.
- Možnost vytváření učební osnovy – seznamu témat jednotlivých týdnů s odkazy do příslušných výukových materiálů.
- Snadné přizpůsobení funkčnosti a vzhledu.
- Možnost rozšíření systému pomocí uživatelských pluginů, modulů apod.

Naopak věci, které u LMS pro použití na ČVUT považuji za poměrně zbytečné, jsou:

- Organizování studentů do virtuálních tříd (na ČVUT stejně probíhají reálná cvičení a LMS nemá za úkol je nahrazovat).
- Online hodnocené testy (v teple domova může student podvádět, může mít problém s internetovým připojením atd.).

2.2 E-learning na FEL ČVUT

V současné době je na ČVUT provozován e-learning v několika formách (shrnutí viz na `http://www.cvut.cz/informace-pro-studenty/cw`):

- *Courseware* umístěný na adrese `https://ocw.cvut.cz/ocw/` slouží především jako rozcestník, směřující na stávající studijní materiály (v případě Katedry počítačů jde obvykle o weby jednotlivých předmětů na serveru `http://service.felk.cvut.cz/courses`). Nejedná se o e-learning v pravém smyslu slova.
- *Learning Gateway* na adrese `https://www.lg.cvut.cz/` je rozšířená verze produktu *Microsoft Class Server 3.0 CZ*, který vyhrál výběrové řízení na systém pro podporu výuky na ČVUT v roce 2003 (viz [10]). V dalších letech probíhalo testování a shrnování požadavků na toto řešení, mezi hlavní požadavky patřilo např. propojení se stávajícími informačními systémy na ČVUT (agenda studentů apod.) nebo rozšíření systému tak, aby vyhovoval vzdělávacím potřebám vysoké školy (primárně je určen pro základní a střední školy). Poté byl systém nasazen do testovacího provozu na Katedře telekomunikací. V současné době se zdá, že činnost kolem Learning Gateway spíše stagnuje, soudě podle velmi nízké četnosti aktualit, podle výskytu „testovacího“ obsahu a také podle mé marné snahy zjistit další informace na e-mailové adrese správce serveru.

- *Moodle* – e-learningový systém na adrese `http://ocw.cvut.cz/moodle/` obsahuje několik výukových kurzů z oblasti architektury, elektrotechniky, programování a dalších. Některé z předmětů zde mají poměrně kvalitní kurzy, bohužel oblast programování a informatiky patří k těm nejméně zastoupeným.

2.3 Analýza dalších řešení

V dnešní době existuje nespočet webových open-source CMS a mezi nimi lze nalézt i systémy specializované na elektronickou podporu výuky (LMS). Vzhledem k tomu, že jsou tyto systémy volně k dispozici, rozhodl jsem se svou implementaci založit na některém z nich. V následující rešerši tedy zhodnotím kladné a záporné vlastnosti vybraných systémů a budu se zamýšlet nad tím, jak vhodné jsou pro implementaci mých požadavků.
Kandidáty na vhodné LMS jsem hledal několika způsoby. Jednak jsem procházel webové stránky různých škol a univerzit používajících e-learning a zkoumal, který systém používají. Dále jsem hledal weby zaměřené přímo na porovnávání LMS, kde jsem našel celé seznamy dostupných systémů. A nakonec jsem vyhledával a procházel různé internetové diskuze na téma e-learningu, ze kterých jsem si mohl o používaných systémech také utvořit představu. Z takto posbíraných kandidátů jsem vyřadil ty, které nebyly open-source, nebyly rozšiřitelné, nebyly zdarma nebo se nějakým jiným způsobem znatelně lišily od mnou vytyčených požadavků. Tím mi zbyl užší výběr kandidátů.
Mezi zdroje patřily například adresy:

- `http://www.unesco.org/cgi-bin/webworld/portal_freesoftware/cgi/page.cgi?d=1&g=Software/Courseware_Tools/index.shtml`
- `http://en.wikipedia.org/wiki/Learning_management_system`
- `http://www.lmstalk.com/`
- `http://google.com/?q=open%20source%20lms`

Takto vytříděné nejlepší systémy jsem postupně instaloval na lokální server a zevrubně testoval. Hodnotil jsem uživatelskou přívětivost, přehlednost, jednoduchost obsluhy jak z hlediska „učitele“, tak z hlediska „studenta“. Vytvořil jsem tedy učební osnovu, několik výukových materiálů („přednášek“) a testů a z pozice studenta jsem potom kurz navštěvoval, výukové materiály četl a testy absolvoval. Dále jsem hodnotil jak snadná je instalace a údržba celého systému a jak náročný problém představuje přizpůsobení vzhledu a funkčnosti mým požadavkům. Na základě zkušeností z krátkého testování jsem u každého systému z tohoto užšího výběru určil jeho klady a zápory a nastínil svůj celkový dojem z daného LMS.

2.3.1 A-Tutor

`http://www.atutor.ca/`

Klady:

- přehledná, jednoduchá instalace,
- diskuzní fóra, chat, FAQ,

- umožňuje vytvářet slovník pojmů pro daný kurz,
- řízení přístupu uživatelů ke kurzu (někdo smí výukový kurz používat, někdo ne, někdo jen bez testů atd.),
- vytváření seznamu doporučené literatury studentům,
- tvorba testů z náhodně vybíraných otázek,
- zpětná vazba u jednotlivých otázek v testu.

Zápory:

- nemá dosud český překlad (pracuje se na něm, ale pomalu),
- nepříliš uživatelsky přívětivé rozhraní pro učitele,
- nepodporuje SCORM,
- nepodporuje AICC.

Celkový dojem: Jedná se o standardní LMS. Z uživatelského hlediska má nepříliš přívětivé ovládání. Znatelným nedostatkem je absence českého překladu a též je jistou nevýhodou, že nepodporuje uznávané LMS standardy, ani jejich zjednodušené verze.

2.3.2 Claroline

`http://www.claroline.net/`

Klady:

- čeština (občas nepřesná, chybná, nedokončená),
- rozšiřitelnost pomocí extensions,
- kvalitní online dokumentace, tutoriály,
- vytváření výukových cest (kapitola následovaná cvičením atd.),
- diskuzní fóra, chat,
- import SCORM souborů,
- systém umožňuje v jedné otázce zobrazovat náhodnou podmnožinu odpovědí.

Zápory:

- nepracuje při zapnutém tzv. safe-módu na serveru (safe-mód je bezpečnostní mechanismus běhového prostředí PHP, který řeší některá rizika na serverech, které hostují více virtuálních serverů),
- málo druhů otázek ve cvičeních (neexistuje tu např. otázka typu přiřazení),
- místy relativně složitá administrace,
- nepodporuje AICC.

Celkový dojem: Na pohled jednoduchý, přesto poměrně silný LMS. Administrace je docela přehledná, ale ne moc intuitivní (člověk musí předem vědět, co dělá, například pro vložení nového testu do učebního plánu se musí nejdřív vrátit a vytvořit test, nemůže ho nechat tvořit rovnou do daného učebního plánu a podobně).

2.3.3 Dokeos

`http://www.dokeos.com/`
(vývojová větev Clarolinu)

Klady:

- spousta drobností nad Clarolinem (především co se týče správy tříd, agendy atd.),
- výuková cesta se může skládat z objektů libovolného typu (nejen z textů a testů jako v Claroline, ale též z prezentací, diskuzí atd.),
- videokonference,
- podpora SCORM,
- podpora AICC,
- konverze PowerPointových prezentací do výukových online materiálů.

Zápory:

- není český překlad,
- velmi špatná online dokumentace,
- špatná práce s kódováním češtiny (předvyplňuje do formulářových polí zápis entity místo samotného znaku apod.),
- bez podpory DHTML editoru prohlížečem má tvůrce kurzu hodně ztíženou práci.

Celkový dojem: Pěkný a přehledný LMS s užitečnými funkcemi (konverze PowerPointové prezentace do výukového materiálu), který bohužel trpí několika podstatnými nevýhodami – špatná práce s kódováním znaků se projeví nejen na samotném kurzu, ale i na tom, že výukové materiály nelze rozumně exportovat a podobně. Navíc PHP kód systému produkuje spoustů chybových hlášek typu *warning* a *notice*, což obecně svědčí o nepříliš kvalitním a bezpečném programování.

2.3.4 Ilias

`http://www.ilias.de/`

Klady:

- podporuje SCORM (pouze 1.2, což není nejnovější verze),
- podporuje AICC,
- umožňuje vytvářet pro celý kurz slovník pojmů,
- fóra, chat,
- český překlad.

Zápory:

- problematická instalace (vyžaduje instalaci různých potřebných balíčků na serveru, které na průměrném webhostingu zdaleka nemusí být k dispozici – např. ImageMagick, libxslt),
- mohutná instalace (přes 200MB),
- málo obsáhlá dokumentace a nápověda.

Celkový dojem: Standardní výukový systém s obvyklou sadou funkcí. Na kráse mu však ubírají přemrštěné nároky na software na serveru a mohutná instalace, u které je až s podivem, jak velký prostor zabírá v porovnání s ostatními LMS, které navíc tak náročné nároky na server nekladou.

2.3.5 Moodle

`http://moodle.org/`, `http://moodle.cz/`

Klady:

- kvalitní český překlad,
- kvalitní online dokumentace a podpora, bohatá nápověda,
- podpora SCORM,
- široká škála rozšíření a plug-inů, možnost rozšířit systém o vlastní,
- početná česká komunita uživatelů,
- diskuzní fóra, chat.

Zápory:

- nemusí pracovat správně při zapnutém safe-módu na serveru,
- neintuitivní a trochu matoucí ovládání (z pohledu tvůrce kurzu),
- nepodporuje AICC,
- mizí hlavní menu v administraci, administrátor se pak hůře orientuje.

Celkový dojem: Jednoduchý a přitom velmi silný a snadno rozšiřitelný LMS, o který se stará početná a aktivní komunita vývojářů a uživatelů. Mezi jeho největší přednosti patří jeho rozšiřitelnost a velmi dobrá a bohatá dokumentace a výstižná nápověda.

2.3.6 Olat

`http://www.olat.org/`

Systém	Použitelnost	Dokumentace	Standardy	Čeština
A-Tutor	nepřívětivé rozhraní pro učitele	ano, použitelná	-	zatím ne
Claroline	neintuitivní administrace	kvalitní, tutoriály	SCORM	ano
Dokeos	přehledné rozhraní, ale špatná práce s kódováním znaků	nepřehledná vývojářská dokumentace	SCORM, AICC	ne
Ilias	přehledné rozhraní	málo obsáhlá	(SCORM), AICC	neúplná
Moodle	místy neintuitivní rozhraní	kvalitní dokumentace, aktivní komunita	SCORM	kvalitní
Olat	nepřívětivé rozhraní	bohatá dokumentace	SCORM	ano

Tabulka 2.1: Srovnání open-source LMS

Klady:

- podpora SCORM,
- tvorba učebních materiálů na bázi wiki,
- diskuzní fóra,
- bohatá online dokumentace, slušná nápověda.

Zápory:

- nevhodný způsob reprezentace výsledků testů (uživatel se dozví, kolik otázek zodpověděl špatně, ale nedozví se které a podobně, tedy v podstatě žádná zpětná vazba),
- chaotická práce s jazykovými verzemi systému (přepnutí do jiného jazyka se někdy projeví, někdy ne)
- celkově nepřehledné uživatelské rozhraní s nedostatečně popisnými chybovými hláškami a podobně

Celkový dojem: Systém Olat nepřináší oproti výše uvedeným žádná vylepšení. Práce s ním se mi nejevila moc příjemná, systém se nechoval uživatelsky přívětivě. Celý systém na mě zapůsobil jako jakási betaverze, ačkoli jí zřejmě není.

2.4 Srovnání uvedených systémů

Pro přehlednost si shrňme hlavní rysy, klady a zápory jednotlivých systémů ve společné tabulce (2.1). Vraťme se nyní k nejdůležitějším požadavkům, které jsme na hledaný LMS kladli v kapitole 2.1.

- *Snadné přizpůsobení funkčnosti a vzhledu.* Z tohoto hlediska mají velkou nevýhodu systémy s nedostatečnou vývojářskou dokumentací. Pokud neexistuje použitelná dokumentace popisující strukturu a význam souborů, aplikační rozhraní (API), strukturu volitelných vzhledů, základní kostru a principy uživatelských rozšíření,

je pro vývojáře velmi časově náročné (ne-li nemožné) vytvořit ve všech směrech správně fungující rozšíření systému.

Z tohoto hlediska si nejhůře stojí systémy Dokeos a Ilias, jejichž vývojářská dokumentace je chudá a málo přehledná.

- *Ověřování znalostí pomocí testů.* Toto umožňují v různé míře všechny zkoumané systémy. Ve všech systémech je možné vytvářet balíky testových otázek různých druhů („zaškrtávací", přiřazovací, seřazovací atd.) a sestavovat testy a ty potom předkládat k plnění studentům. Všechny tyto LMS také nabízejí možnost testy omezovat časově nebo počtem pokusů studenta, vést klasifikaci atd. Pro účely doplnění klasického studia na ČVUT se na testy dívám především jako na prostředek, jímž by si studenti měli ověřovat, oživovat a testovat své znalosti. Proto možnosti klasifikace nebo omezování počtu pokusů (či omezování časové) považuji pro naše účely za nedůležité. Mnohem důležitější je *odezva*, jíž se studentovi dostane na špatně zodpovězené otázky.

  V tomto směru nejvíc zaostává systém Olat, jehož možnost zpětné vazby vůči studentovi při testu je minimální (viz 2.3.6).

  Ostatní systémy umožňují přidat ke každé otázce vysvětlení správné odpovědi, toto vysvětlení student uvidí (při patřičném nastavení testu), pokud zodpoví na otázku špatně.

  Systémy Claroline, Dokeos a Moodle navíc umožňují také přidávat vysvětlující komentář ke každé z možných odpovědí, student se tedy při zaškrtnutí nesprávné odpovědi dozví, proč je ta konkrétní odpověď nesprávná.

- *Možnost vytvořit učební osnovu.* Tuto funkci nabízejí všechny jmenované systémy, patří to k jejich základním funkcím. LMS umožňují vytvářet buď *časovou* osnovu, odpovídající jednotlivým časovým úsekům (například týdnům) kurzu, nebo osnovu *tematickou*, kde nehraje roli čas, ale studentův vlastní pokrok ve studiu.

  Do studijní osnovy lze přidávat materiály různého druhu – od přednáškových textů a odborných článků (ne tolik vhodných pro e-learning – viz 1.3) přes interaktivní výukové materiály (animace, applety, zvukové nahrávky, simulace) až po již zmiňované testy. V tomto jsou si všechny zmiňované systémy v podstatě rovny, liší se jen například škálou toho, jak bohatě a komfortně lze editovat text, jaké druhy animací lze vkládat (Java, Flash, gif…) a podobně.

Na základě tohoto shrnutí vyberu v další kapitole nejvhodnějšího kandidáta na LMS k nasazení na Katedře počítačů, nainstaluji jej na školním serveru a přizpůsobím specifickým potřebám.

3 Výběr systému a jeho analýza

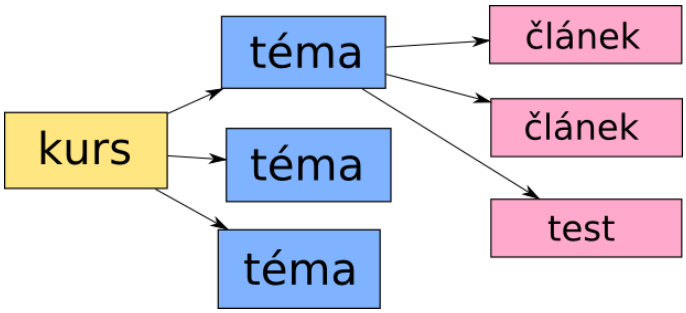
V této kapitole nejprve zvolím systém, který použiji a poté budu zkoumat, jak dobře tento systém vyhovuje požadavkům a co bude potřeba doprogramovat.

3.1 Volba systému

Při shrnutí poznatků z 2.4 vychází pouze několik málo možných kandidátů. Dokeos a Ilias mají příliš chudou vývojářskou dokumentaci, Dokeosu navíc chybí český překlad. Olat má jen velmi minimální možnost zpětné vazby pro studenta u testu a spolu s A-Tutorem trpí neintuitivním uživatelským rozhraním. Zbývá Claroline a Moodle. Vzhledem k tomu, že v ČR je Moodle rozšířený a často používaný a pohybuje se kolem něj široká (nejen česká) komunita vývojářů a uživatelů a i jeho dokumentace je o něco bohatší než u Claroline, zvolil jsem jako vítězného kandidáta právě Moodle.

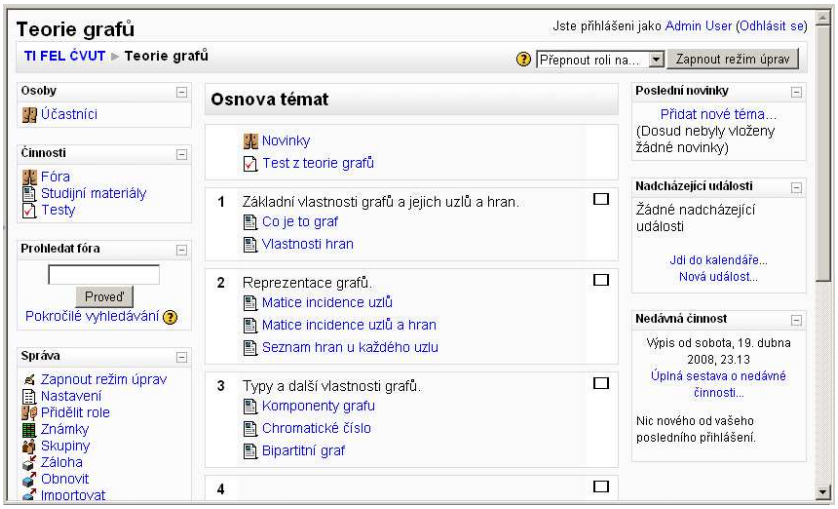
3.2 Hierarchie učebních textů

Jedním z požadavků na LMS byla možnost členit hierarchicky učební texty a materiály. Toto je v Moodle řešeno tak, jako ve většině jiných LMS – totiž pomocí *kurzů*, *témat* a jednotlivých *materiálů*.



Obrázek 3.1: Hierarchická struktura kurzu v Moodle

Na vrcholku hierarchie stojí kategorie kurzů. Každá kategorie obsahuje výukové kurzy. Ty svým záběrem odpovídají jednomu školnímu předmětu. Každý kurz se uvnitř rozděluje do témat (viz obr. 3.1). Ta mohou být členěna buď časově, kdy každé téma odpovídá například jednomu týdnu nebo čistě tematicky, nezávisle na čase. V principu se jedná o stejný způsob dělení, akorát v případě časového dělení je možné učební materiály zpřístupňovat až od určitého data.
A konečně, do každého tématu je možné umisťovat libovolné množství učebních materiálů, ať už se jedná o přednášky, články, soubory jakéhokoli druhu, testy, wiki stránky atd. Jak taková hierarchie vypadá v administraci Moodle lze vidět na obr. 3.2. Touto třívrstvou hierarchií je tedy realizováno hierarchické členění textů a výukových materiálů v Moodle.



Obrázek 3.2: Kurz, témata a výukové materiály v administraci Moodle

3.3 Slovník pojmů

Dalším požadavkem byla přítomnost přehledného slovníku pojmů. Má existovat možnost odkazovat se na definované pojmy přímo z učebních textů. Moodle disponuje modulem *glossary* (jeho český název je *Heslo ze slovníku*), který takový slovník realizuje. Slovník lze vytvořit na více úrovních hierarchie – lze ho vytvořit pouze pro dané téma nebo pro celý kurz, případně slovník použít jako globální pro celý web. Na vytváření slovníku se navíc mohou spolupodílet sami studenti.
Ke zprovoznění slovníku je potřeba aktivovat *činnost* glossary, to je ta část modulu, která umožňuje správci kursu vytvářet a spravovat slovníky, přiřazovat je do kategorií a podobně. Kromě toho je třeba aktivovat také *filtr* glossary, což je mechanismus, který zajistí, že výskyty hesel ze slovníku se při zobrazení učebních textů automaticky odkazují na ministránku s vysvětlením pojmu.

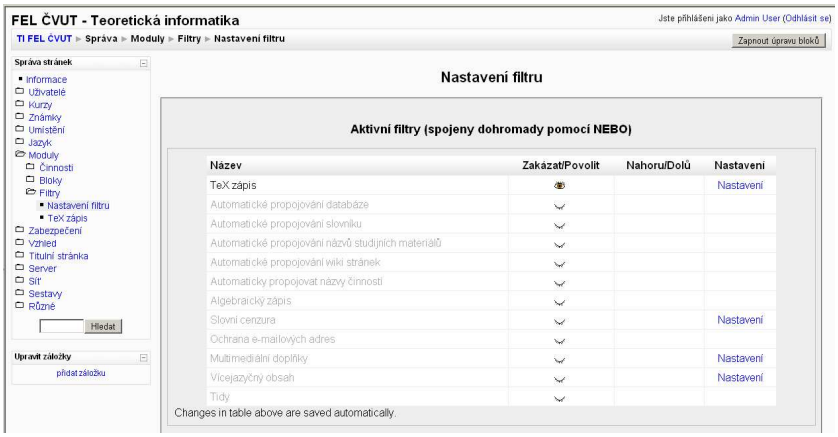
3.4 Zápis matematických výrazů

Při zpracovávání textů (učebních, zadání otázek, příspěvků ve fórech atd.) může Moodle používat *filtry*, což jsou zásuvné moduly, které umí text nějakým způsobem zpracovávat. Základní metodou, jak v Moodle zobrazovat matematické výrazy, je TEX filtr, který je v Moodle přítomen ihned po instalaci, před použitím je ale potřeba jej v administraci zapnout. To se provádí v administračním menu *Moduly* → *Filtry* → *Nastavení filtru* kliknutím na symbol zavřeného (spícího, neaktivního) oka, viz obr. 3.3. Poté lze v libovolných textech používat TEXový zápis matematických výrazů, tedy např.

```
$$ sqrt( x + y ) $$
```

Takovýto úsek textu se automaticky nahradí vygenerovaným obrázkem. Sestavení obrázku může fungovat dvěma způsoby:

- Je-li na serveru k dispozici některá implementace systému LATEX a grafická knihovna ImageMagick, stačí v nastavení filtru definovat cesty k jejich spustitelným souborům. Filtr potom matematický výraz převádí tak, že k němu přidá hlavičku, aby tak



Obrázek 3.3: Zapnutí TEX filtru pro zobrazování matematických výrazů

dočasně vznikl plnohodnotný TEXový dokument. Ten se programem `latex` nechá vykreslit do souboru v DVI formátu, který se utilitou `dvips` převede do PostScript podoby a posléze programem `convert` z knihovny ImageMagick do výsledného obrázku GIF.

V mém případě nešla bohužel tato varianta použít. Na serveru není nainstalována knihovna ImageMagick (o to bych však mohl požádat správce), ale hlavně je PHP nastaveno direktivou *open_basedir* tak, že nelze číst soubory, které nejsou v podadresářích právě běžícího PHP skriptu, což mi v důsledku znemožňuje programy `latex` a `dvips` používat. Po konzultaci se správcem serveru jsem se dověděl, že nemůže z bezpečnostních důvodů pro můj web udělat vyjímku.

- Druhou variantou je použití systému *MimeTEX*, který se jako alternativa dodává přímo s TEXfiltrem. Jedná se o systém vytvořený právě pro snadné zobrazení matematických výrazů na webu. Výraz v TEXu se předá jedinému programu, který rovnou vygeneruje obrázek ve formátu GIF.

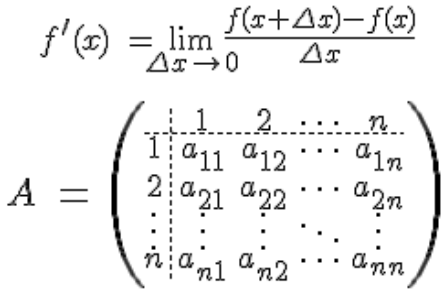
  S TEXfiltrem se dodává již zkompilovaná podoba MimeTEXu pro několik nejpoužívanějších operačních systémů (Windows, Linux, FreeBSD). Na serveru, kam jsem Moodle nainstaloval, však běží SunOS, bylo tedy potřeba stáhnout zdrojové kódy MimeTEXu z `http://www.forkosh.com/` a ty zkompilovat v adresáři `./filter/tex` příkazem

  ```
  gcc -DAA mimetex.c gifsave.c -lm -o mimetex.sunos
  ```

  a dále doplnit novou řádku do souboru `./filter/tex/lib.php`:

  ```
  switch (PHP_OS) {
    case "Linux":   return "$CFG->dirroot/filter/tex/mimetex.linux";
    case "Darwin":  return "$CFG->dirroot/filter/tex/mimetex.darwin";
    case "FreeBSD": return "$CFG->dirroot/filter/tex/mimetex.freebsd";
    case "SunOS":   return "$CFG->dirroot/filter/tex/mimetex.sunos";
  }
  ```

Obrázky produkované MimeTEXem nedosahují tak vysoké kvality jako při kombinaci LATEXu a knihovny ImageMagick, ale i přesto je výsledek uspokojivý (viz obr. 3.4).



Obrázek 3.4: Matematické zápisy vykreslené systémem MimeTEX

Filtr samozřejmě obrázky nenechává vykreslovat znovu při každém zobrazení, místo toho si je ukládá na disk pod názvem, který odpovídá otisku („hashi“) TeXového zápisu výrazu.

3.5 Testy a jejich zpětná vazba

Vzhledem k tomu, že testy znalostí chápu v našem řešení především jako prostředek, kterým si student ověřuje své znalosti, kladu velký důraz na to, aby nesprávné odpovědi v testu poskytly studentovi co nejkvalitnější zpětnou vazbu.
V systému Moodle k tomu slouží několik mechanismů:

- U každého testu lze definovat bodové ohodnocení jednotlivých otázek, resp. odpovědí. Celý test pak lze slovně ohodnotit podle počtu dosažených bodů. Vzhledem k tomu, že body může student ztratit na kterýchkoli otázkách, je jedinou zpětnou vazbou tohoto mechanismu informace o tom, jak dalece student dané téma ovladá, což pro něj jistě není dostačující informace. Mnohem užitečnější pro studenta je, když se dozví, které pojmy nebo dokonce která celá témata by si měl zopakovat a podobně.

- U každé otázky lze definovat *celkovou reakci* (*general feedback*), což je krátký text, který studentovi může celou otázku ozřejmit v případě nesprávné odpovědi. To je jistě užitečná vlastnost.

- Předchozí možnost dává studentovi dobrou zpětnou vazbu při odpovědích na konkrétní jednotlivé otázky. Co když chce ale student (například před závěrečným testem psaným na cvičení) otestovat své celkové znalosti a zjistit, ve kterých tématech má například celkové znalostní mezery? K tomu již nestačí zpětná vazba na jednu nebo dvě otázky např. z deseti. Moodle umí připravit test náhodným výběrem stanoveného počtu z celé banky otázek. Dejme tomu, že student odpoví nesprávně na dvě otázky z deseti a z *celkové reakce* se poučí, jaké jsou správné odpovědi. Co když ale tyto dvě otázky blízce souvisely a nesprávné odpovědi by

mohly naznačovat, že student nerozumí celému jednomu tématu? Touto otázkou se chci zabývat a jako její řešení implementovat rozšíření, které by mělo na základě studentových nesprávných odpovědí určit, která témata by neměl student podcenit a znovu si je měl projít.

Testové otázky se v Moodle třídí do kategorií, ty mohou být zavedeny u konkrétního testu, tématu, výukového kurzu, kategorie kurzu nebo u celého systému.
Testové otázky mohou být v Moodle různého typu, např.:

- vypočítávaná úloha,
- přiřazování,
- krátká tvořená odpověď,
- pravda/nepravda.

Více typů otázek a jejich podrobný popis lze nalézt v [4].

4 Realizace

4.1 Instalace a nastavení Moodlu

Pro instalaci a přizpůsobení systému Moodle jsem dostal k dispozici účet na webovém serveru Katedry počítačů FEL ČVUT – `http://webdev.felk.cvut.cz/~hocker1`. Základní parametry tohoto webserveru jsou následující (viz `http://webdev.felk.cvut.cz/~hocker1/phpinfo.php`):

- http server Apache 2.2.4,
- skriptovací jazyk PHP 5.2.4,
- databázový server MySQL 5.0.45,
- vypnutý safe-mód.

Z download sekce domovského webu systému Moodle (`http://www.moodle.org/`) jsem stáhl jeho nejnovější verzi 1.9. Archiv s celým systémem jsem dekomprimoval do kořenové složky svého účtu na webserveru. Poté jsem v administračním rozhraní serveru katedry (`https://webdev.felk.cvut.cz/db/`) vytvořil prázdnou databázi s kódováním znaků v utf-8. Pak už stačilo pouze spustit (přesně podle pokynů na webu Moodle) automatický instalátor zadáním adresy `http://webdev.felk.cvut.cz/~hocker1` do webového prohlížeče.

Obrázek 4.1: Moodle při instalaci zkontroluje nastavení PHP

Během instalace Moodle spolupracuje s uživatelem. Zjistí od něj, v jakém jazyce se má nainstalovat, jaká cesta vede k datovému adresáři (který může být pro větší bezpečnost naprosto oddělen od samotné instalace Moodle na webserveru), zkontroluje nastavení PHP (viz obr. 4.1) a zjistí od uživatele parametry použité databáze. Poté vytvoří na disku a v databázi všechny potřebné struktury a nakonec nechá uživatele vytvořit administrátorský účet. Poté je připraven k použití.

4.2 Oprava některých chyb v Moodle

Brzy po nainstalování systému na webserver Katedry počítačů jsem zjistil, že Moodle při některých akcích vypisuje chybovou hlášku typu *Strict Standards*, obvykle s dodatkem, že se Moodle pokouší vracet hodnotu z funkce odkazem nebo že se pokouší předefinovat již existující konstruktor nějaké třídy. Tyto chybové hlášky jsou vesměs upozorněním, že takto používané techniky jsou *zastaralé* a že by se neměly používat. Některé moduly systému Moodle jsou totiž psány pro PHP verze 4, zatímco na webserveru je nainstalováno PHP 5. V PHP manuálu se doslova píše:

> Run-time notices. Enable to have PHP suggest changes to your code which will ensure the best interoperability and forward compatibility of your code.

Na samotnou funkčnost systému Moodle však nemají vliv, jedná se jen o upozornění na používání zastaralých zápisů. Konkrétně se například při vytváření nového objektu dříve (PHP4) mohl právě vzniklý objekt předat proměnné referencí:

```
$object =& new Class();
```

V PHP5 se již všechny objekty vždy automaticky předávají referencí a tento zápis je proto zastaralý. K potlačení těchto chybových hlášek je třeba nastavit PHP direktivu *error_reporting* na hodnotu `E_ALL & ~E_STRICT`, což znamená, že se chyby typu *Strict Standards* nebudou zobrazovat, viz `http://cz.php.net/manual/en/function.error-reporting.php`. To lze provést více způsoby:

- Přímo v PHP skriptu volat funkci *error_reporting()* nebo *ini_set()* a s jejich pomocí hodnotu direktivy nastavit. Abych nemusel tuto funkci volat z každého skriptu, bylo by nejlepší zapsat ji do některého PHP souboru, který je využíván všemi ostatními skripty. Takovým je soubor `config.php`. Potřebný příkaz jsem do něj zapsal, kýžený efekt se ale bohužel nedostavil. PHP totiž nejdřív celý soubor prochází a překládá (resp. hledá definice funkcí, tříd a podobně) a teprve poté spustí, takže nejdřív během překladu narazí na zápis způsobující chybovou hlášku a teprve poté se skript spustí a proběhne (již pozdě) volání *error_reporting()*.

- Zapsat danou direktivu do konfiguračního souboru serveru (`httpd.conf`), tím by se direktiva nastavila ještě dřív, než by se PHP skript začal překládat a dalo by se tak vyhnout problému z předchozího bodu. K tomuto souboru však bohužel nemám na serveru z bezpečnostních důvodů přístup.

- Zapsat onu direktivu do konfiguračního souboru PHP (`php.ini`), k němu ovšem podobně jako v předchozím případě nemám přístup.

- Vytvořit v kořenovém adresáři Moodle soubor .htaccess a v něm hodnotu direktivy nastavit zápisem `php_value error_reporting 4095`. Školní webserver je bohužel nastaven tak, že v souborech .htaccess nelze používat direktivy typu `php_value`, po konzultaci se správcem serveru jsem se dozvěděl, že to není z bezpečnostních důvodů možné.

- Poslední možností je chyby ručně opravit a prozkoumat pečlivě kód, abych se ujistil, že se po opravě těchto chyb nezačne systém chovat jinak. Jak ale zjistit, které všechny skripty obsahují tyto chyby?

  Nejdříve jsem zapátral v dokumentaci k jednotlivým modulům, hledaje informaci o podporované verzi PHP. Tím bych vyloučil moduly psané přímo pro PHP5. V dokumentaci k modulům se však informace o verzi PHP nevyskytuje.

  Vyhledal jsem všechny soubory se zastaralým zápisem typu `$object =& new Class();`. Nalezl jsem jich více než 200 a většina z nich obsahovala chybu na více místech. Rozhodl jsem se tedy pro hromadné nahrazení výrazu `=& new` za `= new`, což pokrývá právě vytváření nových objektů. Náhradu jsem provedl v shellu v kořenovém adresáři webu příkazem:

  ```
  find . -name '*.php' | xargs perl -pi -e 's/=&\s*new +/= new /g'
  ```

  Tento příkaz najde všechny PHP soubory v adresářové struktuře Moodle a na každý z nich použije PERLovský regurální výraz pro náhradu výrazu „`=&\s*new`“ (čili znaky `=&`, poté libovolný počet prázdných znaků a operátor `new`) za „`= new`“.

4.3 Systém pro zpětnou vazbu testů

Jak již bylo řečeno dříve (viz 3.5), rozhodl jsem se Moodle rozšířit o speciální zpětnou vazbu u testů, tato zpětná vazba má sloužit k tomu, aby se student na základě svých odpovědí v testu dozvěděl, které tematické celky mu činí problémy a měl by si je tedy zopakovat. Tento problém budu řešit pro otázku typu *úloha s výběrem odpovědí* (*multichoice*), protože patří mezi často používaný typ otázky a také proto, že ke zjištění správnosti odpovědi není třeba posudek člověka (jako např. u *dlouhé tvořené odpovědi*). Zjištěné výsledky potom půjdou aplikovat i na některé další typy otázek.
Cílem mého snažení tedy bude:

- Propojit vhodným způsobem otázky s učebními tematickými celky, tedy umožnit autorovi otázky, aby definoval, se kterým tématem nebo tématy otázka, případně konkrétní možná odpověď, souvisí. Otázky v Moodle lze sice členit do kategorií k jednotlivým kurzům, to ale nestačí. Za prvé je možné, že otázka souvisí například se třemi různými učebními texty (bylo by tedy třeba pro každou kombinaci učebních textů vytvořit samostatnou kategorii a do ní otázku vložit a to je pracné a zdlouhavé). Za druhé je možné, že jedna z odpovědí souvisí s určitými tématy, jiná odpověď zase s jinými tématy. Například u otázky *„Vyberte pravdivá tvrzení o úplném grafu“* možná odpověď *„obsahuje cykly“* souvisí s tématy *Typy grafů* a *Posloupnosti hran a uzlů*, zatímco odpověď *„je reprezentován incidenční maticí s nulami na diagonále“* souvisí s tématy *Typy grafů* a *Reprezentace grafu*. Přitom nezáleží na tom, zda je odpověď pravdivá nebo ne. Tak či tak ji student může zaškrtnout nesprávně (nezaškrtne pravdivou nebo zaškrtne nepravdivou) a vzbudit tak podezření, že přiřazeným tématům nerozumí.

- Nalézt způsob, jak na základě studentových odpovědí a propojení otázek a témat určit, která témata doporučit k zopakování a s jakou váhou důležitosti. Mají se doporučit všechna témata, u nichž byly zjištěny nějaké neznalosti? Nebo se z nich mají vybrat pouze ta, u kterých student prokázal neznalost víckrát v různých otázkách?

4.3.1 Organizace otázek v Moodle

Systém Moodle umožňuje vývojářům vytvářet nejrůznější druhy zásuvných modulů (pluginů) a jedním z těchto druhů je také *typ otázky* (*question type*). Všechny typy otázek, které jsou v Moodle hned po instalaci přítomny, jsou řešeny jako zásuvné moduly. To dává vývojářům možnost snadno doplňovat další typy otázek.
Proto jsem se rozhodl vytvořit nový typ otázky, který bude vycházet z původního typu *multichoice* (*úloha s výběrem odpovědí*) a označil jsem jej *multichoicefdb – úloha s výběrem odpovědi a zpětnou vazbou.*
„Zaškrtávací“ otázky jsou navrženy tak, že může být jediná odpověď 100% správná nebo že správných může být více, přičemž některé mohou být „správnější“ a některé „méně správné“. Ve skutečnosti tedy mám k otázce m odpovědí a u každé procentuálně udávám, jak velkou část celkového počtu bodů za otázku tvoří tato odpověď. Špatné odpovědi mají 0 % nebo méně, správné mají více než 0 %. Má-li oodpověď 100 %, jedná se o jedinou správnou odpověď.

4.3.2 Funkce systému

Student se snaží co nejlépe odpovědět na otázku. Výstupem jeho „zaškrtávání“ odpovědí je seznam odpovědí, které zaškrtl a odpovědí, které nezaškrtl. A na základě toho, zda zaškrtal správné či ne, se bude odvíjet vstup do systému pro zpětnou vazbu.
Dejme tomu, že mám následující možné odpovědi a jejich procentuální ohodnocení správnosti, které určil tvůrce otázky:

```
A       B       C       D
-50%    -50%    50%     50%
```

Z toho plyne, že odpovědi C a D jsou správné a měly by v kompletně správné odpovědi být obě zaškrtnuty. Naopak A a B jsou nespravné a neměly by být zaškrtnuty. Student při testu odpoví následujícím způsobem:

```
A       B       C       D
ano     ne      ano     ne
```

V bodech A a D odpověděl nesprávně. A toto budou vstupní impulsy do systému, který by měl odhalit, ve které oblasti má student znalostní mezery. Spolu s odpovědí na tuto otázku také student podobně odpovídal dejme tomu na 9 dalších otázek v tomto testu. A na některé opět nesprávně. Náš systém má tedy z těchto 10 různých odpovědí co nejlépe vydedukovat, čemu student nerozumí. Výstupem by měla být nějaká informace typu:

> Pravděpodobně nerozumíš vůbec látce z kapitoly X, měl by sis ji prostudovat.
> Také ti dělá velké problémy téma Z.

4.3.3 Propojení otázek, odpovědí a témat

U každé odpovědi dané otázky bude učitelem nastaveno, které oblasti znalostí jsou pro danou odpověď třeba, nebo jinými slovy ve kterých oblastech má student patrně mezery, pokud tuto odpověď zaškrtnul nesprávně.
Každá odpověď bude obsahovat seznam dvojic:

- odkaz na znalostní okruh (téma)

- váhu daného okruhu u této odpovědi (když odpovím, že poloměr grafu je počet uzlů, pravděpodobně nerozumím pojmu *poloměr*, ale je zde také jistá šance, že mám problémy s pojmem *hrana* nebo dokonce *graf*)

Příklad otázky a jedné odpovědi se vstupy do systému pro zpětnou vazbu:

```
Co je to poloměr grafu?

A) Počet hran v grafu.          -> Poloměr grafu        80
                                -> Hrana grafu          15
                                -> Graf                  5
B) ...
C) ...
D) ...
```

Tedy pokud student zaškrtne (nesprávně) odpověď A, systém bude upozorněn, kterým tématům a s jakými váhami student nerozumí. Jak bude systém s touto informací nakládat a jak se z výsledků více otázek dobere k doporučení, která témata zopakovat, bude popsáno v 4.3.4.

4.3.4 Ohodnocení témat na základě odpovědí

Nyní máme tedy propojeny odpovědi na otázky s tématy a máme je ohodnoceny váhou. Nyní je třeba určit, jak se na základě těchto vazeb a studentových odpovědí určí, která témata doporučit studentovi k zopakování.
Ke každé otázce a každé odpovědi přísluší několik veličin, na kterých můžeme stavět:

- *Body* za správně zodpovězenou otázku. To je vlastnost, kterou má v Moodle každá otázka. Autor testu určí, kolik bodů student dostane za správně zodpovězenou otázku. Pokud má otázka více možných odpovědí a student zaškrtne například pouze jednu z nich, dostane z celkového počtu bodů pouze odpovídající část (na základě vlastnosti *fraction* – viz dále).

- *Známka* (v orig. „*fraction*“) je vlastnost odpovědi a autor otázky s její pomocí určuje, zda je odpověď správná. Pokud ano, má *známka* hodnotu větší než nula. Známka se udává v procentech a v případě, že je správná pouze jedna z možných odpovědí, měla by tato mít *známku* 100 %, ostatní potom méně než 0 %. Pokud je možných více odpovědí, měly by *známky* těchto odpovědí dávat v součtu 100 % a měly by vyjadřovat, jak důležitá která z odpovědí je (viz příklad v 4.3.2).

- *Váhy témat* u jednotlivých odpovědí, tak jak byly popsány výše. Zbývá vymyslet, v jakém rozsahu se budou tyto váhy pohybovat a zda má smysl například omezovat součet vah u jedné odpovědi a podobně.

Představme si například tuto otázku a možné odpovědi:

```
Chromatické číslo grafu je:
(2 body)

A) Minimální počet barev potřebný pro obarvení grafu.
   známka: 100%
   témata: Barvení grafu       -> 100

B) Počet rovnoběžných hran v grafu.
   známka: -50%
   témata: Barvení grafu       -> 100
           Vlastnosti hran     -> 30

C) Průměrný počet hran připadajících na jeden vrchol.
   známka: -50%
   témata: Barvení grafu       -> 100
           Vlastnosti hran     -> 30
```

Na této otázce ukážu, jak by mohl celý systém fungovat. Obodování otázky a oznámkování otázek je srozumitelné – otázka má jedinou správnou odpověď (A - známka 100 %) a za její správné zodpovězení získá student 2 body. Navíc ale při nesprávných odpovědích (např. zaškrtnutí odpovědi B místo A) získají příslušná témata body, které vyjadřují, že tématu student nerozumí. Např. v případě odpovědi B místo A dostane téma *Barvení grafu* 100 „trestných“ bodů a téma *Vlastnosti hran* 30 bodů, protože takto body určil autor testu.
Naskýtá se první otázka – měly by trestné body u každé odpovědi dávat určitý součet, například 100 (%)? Pokud by tomu tak bylo, může nastat například takováto situace:

```
A) Odpověď 1.
   témata: Téma 1         -> 50
           Téma 2         -> 50

B) Odpověď 2.
   témata: Téma 1         -> 25
           Téma 3         -> 25
           Téma 4         -> 25
           Téma 5         -> 25
```

Autor této otázky se snažil, aby v případě chybných odpovědí byly trestné body rovnoměrně rozděleny mezi určitá témata. Tím ale oslabil u odpovědi B počet trestných bodů pro téma 1, přestože se může jednat o stejně závažnou neznalost jako u odpovědi A. Proto nebudu omezovat součet trestných bodů. Místo toho je omezím na interval $< 1, 100 >$ pro každé téma a navrhnu přibližně stupnici, kterou by se měli autoři testů řídit (viz tabulku 4.1).
Další otázkou je, zda při sčítání trestných bodů brát v úvahu *známku* dané odpovědi a počet trestných bodů podle ní upravit. Známka určuje, jak velkou část celkového počtu bodů tvoří daná odpověď. Moodle umožňuje (v otázkách s více možnými odpověďmi) přiřadit i zápornou známku, která studentovi body odečítá (při nesprávném zaškrtnutí). Kladné známky musí v součtu dávat 100 %, u záporných známek toto omezení není.

Trestné body	**Závažnost tématu**
1-20	Okrajová znalost k otázce.
21-40	Znalost užitečná k pochopení otázky.
41-60	Středně důležitá znalost pro odpověď.
61-80	Důležitá znalost pro odpověď.
81-100	Stěžejní znalost pro zodpovězení otázky.

Tabulka 4.1: Stupnice pro přidělování trestných bodů tématům

Kladných i záporných známek by šlo využít k úpravě trestných bodů přičítaných k tématům. Čím větší kladnou známku má daná odpověď, tím je důležitější, tím větší váhu mají potřebné znalosti k zodpovězení otázky. Naopak čím nižší hodnotu má záporná známka, tím je její nesprávné zaškrtnutí závažnější a tím větší důraz by měl být kladen na zopakování příslušných témat. Jediným problémem je, že zatímco kladné známky jsou dohromady omezeny na 100 %, záporné známky toto omezení nemají a mohou tedy ve výsledku dávat mnohem vyšší trestné body. S tím souvisí také další otázka:
Mají se trestné body k tématům přičítat pouze za zaškrtnutí nesprávné odpovědi nebo i za nezaškrtnutí správné? V prvním případě by mohlo dojít k situaci, kdy student nezaškrtne žádnou odpověď, což je jistě nesprávné (alespoň jedna správná odpověď vždy existuje, jinak by kladné známky nedávaly součet 100 % a to Moodle nedovolí) a přesto se nepřičtou žádné trestné body k tématům, přestože by jistě měly. Proto zvolím druhou variantu – tedy brát v úvahu jak zaškrtnuté nesprávné odpovědi, tak nezaškrtnuté správné.
Tím se můžeme vrátit k otázce, zda započítávat *známku* odpovědi (kladnou či zápornou) do trestných bodů. Máme dvě možnosti:

- známku vůbec nezapočítávat,
- trestné body násobit příslušnou známkou (její absolutní hodnotou).

V prvním případě můžeme přijít o určitou informaci, ale je otázkou, jak moc je tato informace důležitá. Pokud téma dostane trestné body, nemusí už tolik záležet na tom, kolik jich bude. Ve druhém případě získáme přesnější informaci, která bude (při rozumně sestavené otázce) lépe odpovídat realitě. Musíme ovšem provést několik úprav:

- změnit u záporných známek znaménko,
- upravit všechny známky tak, aby jejich součet byl 100 %.

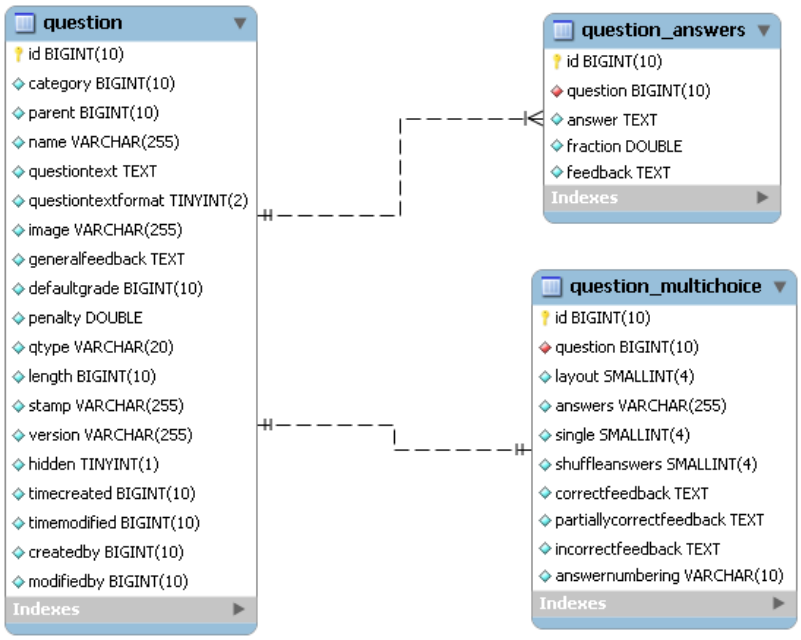
První úprava nám zajistí, že se trestné body budou k tématům vždy přičítat (ať už jsme nezaškrtli správnou nebo zaškrtli nesprávnou odpověď), druhá úprava nám zajistí, že záporné známky nebudou dohromady přesahovat 100 % a navíc že kladné i záporné známky zůstanou v takových vzájemných poměrech, jak je autor testu zadal.
Dále se nabízí otázka, zda trestné body podobně upravovat i v rámci celého testu (skládajícího se například z deseti otázek) podle bodového ohodnocení jednotlivých otázek. Vzhledem k tomu, že vyšší bodové ohodnocení dostávají otázky, které vyžadují více znalostí, měly by tomuto faktu odpovídat i trestné body.
Ve výsledku bude tedy postup pro přidělení trestných bodů tématům vypadat takto:

```
Krok 1 - příprava

A. inicializuj body_celkem = 0
B. pro každou otázku Q typu multichoicefdb z testu proveď:
   1. inicializuj Q.sz = 0 (součet známek)
   2. body_celkem += Q.body
   3. pro každou odpověď A z otázky Q proveď:
      I.  Q.sz += |A.znamka|
      II. pokud je odpověď správně zaškrtnuta, vyřaď ji
          z dalšího zpracování v Kroku 2
   4. pokud nebyla zaznamenána žádná nesprávná odpověď,
      vyřaď otázku Q z dalšího zpracování v Kroku 2
      (student zaškrtl všechny odpovědi správně)

Krok 2 - vyhodnocení

A. inicializuj trestné body všech témat T na T.tb = 0
B. pro každou nevyřazenou otázku Q typu multichoicefdb z testu proveď:
   1. pro každou nevyřazenou odpověď A z otázky Q proveď:
      I. pro každé téma S z odpovědi A proveď:
         a. T.tb += S.tb * (|A.znamka| / Q.sz) * (Q.body / body_celkem),
            kde T je téma odpovídající záznamu S, A.znamka je známka
            pro danou odpověď, Q.sz je součet spočítaný v kroku 1
            a Q.body je počet bodů za danou otázku
C. označ témata s nenulovým počtem trestných bodů (T.tb) jako
   doporučená ke zopakování v pořadí klesajících T.tb
```

Za povšimnutí stojí, že trestné body za nesprávnou odpověď se normalizují vzhledem k počtu všech možných odpovědí v otázce, ne jenom k těm nesprávně zodpovězeným (Krok 1 – A.3.I. – do součtu se započítávají i správně zodpovězené otázky). To v důsledku znamená, že pokud student zaškrtne špatně pouze jednu z odpovědí, nebude to mít tak velký dopad na trestné body, jako když zaškrtne nesprávně více odpovědí na onu otázku. Implementaci popsaného postupu popíšu v 4.3.7 poté, co v následujících kapitolách ukážu, jak se v systému Moodle implementuje nový typ otázky.

4.3.5 Databázový model

Základní tabulkou, která v systému Moodle reprezentuje otázku, je tabulka *question*, jejímž důležitým sloupcem je sloupec *qtype*. Ten obsahuje kód typu otázky. Na hodnotě v tomto sloupci tedy záleží, zda je otázka typu *přiřazování*, *numerická úloha* atd.
U otázek, k nimž patří nějaký počet možných odpovědí (např. typy *přiřazování*, *úloha s výběrem odpovědi*a i náš nový typ *multichoicefdb*) je další důležitou tabulkou *question_answers*. Každý záznam v této tabulce reprezentuje jednu možnou odpověď a obsahuje odkaz na otázku (z tabulky *question*), ke které patří.
K otázce typu *multichoice*, ze které vycházím, patří ještě tabulka *question_multichoice*, ve které jsou upřesněny některé údaje specifické pro tento typ otázky. Jedná se především o sloupec *answers*. Tento sloupec může obsahovat textový řetězec a zapisuje se do něj seznam identifikátorů (id) možných odpovědí (z tabulky *question_answers*) oddělených



Obrázek 4.2: E-R schéma databáze otázek

čárkou. Tato informace se zdá být redundantní vzhledem k tomu, že záznamy v tabulce *answers* se odkazují na otázku, ke které patří, nicméně systém Moodle používá obě informace, sloupec *answers* v tabulce *question_multichoice* například obsahuje i informaci o pořadí odpovědí. To vše je shrnuto v E-R schématu na obr. 4.2.
Budu vytvářet nový typ otázky jako plugin. Tento plugin se bude jmenovat *multichoicefdb* a stejně tak pojmenuji i novou tabulku v databázi, kterou budu potřebovat. Struktura tabulky bude stejná jako u *question_multichoice*s tím rozdílem, že bude obsahovat navíc sloupec *answerfdb*, ve kterém bude k dané otázce textově zaznamenáno, se kterými tématy souvisí které odpovědi.
Tento sloupec bude obsahovat čárkami oddělený seznam uspořádaných trojic, přičemž každá trojice bude mít tvar:

`odpověď:téma:ohodnocení`

Proč jsem zvolil právě takový formát zápisu? V celém systému Moodle je podobný zápis v databázi běžný, například seznam možných odpovědí k otázce je v tabulce uložen v podobě čárkami oddělených *id* odpovědí, rozhodl jsem se tedy zažité zvyklosti neměnit. Jednotlivé složky trojice mají tento význam:

odpověď je id odpovědi z tabulky *question_answers*

téma je id příslušného učebního tématu z tabulky *resource*

ohodnocení je váha vyjadřující, jak silně je daná odpověď spjatá s tématem

Tedy například záznam:

```
1:5:80,1:6:20,2:6:30,2:8:70
```

`db/install.xml`	obsahuje definici nových databázových tabulek
`db/mysql.php`	skript pro vytvoření nových tabulek (zastaralý, již se nepoužívá)
`db/postgres7.php`	viz db/mysql.php
`db/upgrade.php`	skript pro případnou aktualizaci databázových tabulek v případě přechodu na vyšší verzi Moodle
`display.html`	HTML šablona pro zobrazení otázky v rámci testu
`edit_multichoicefdb_form.php`	třída popisující formulář pro editaci otázky
`icon.gif`	ikonka typu otázky pro snadnější orientaci v administraci
`questiontype.php`	třída popisující samotný typ otázky, obsahuje mimo jiné metody pro načítání otázky z databáze do interních datových struktur
`version.php`	verze zásuvného modulu a informace o vyžadované verzi Moodle

Tabulka 4.2: Soubory pro plugin typu question type

říká, že odpověď s id 1 souvisí s tématy 5 a 6 s váhami 80 a 20 a odpověď s id 2 souvisí s tématy 6 a 8 s váhami 30 a 70:

```
odp. 1    ->    téma 5 s váhou 80
                téma 6 s váhou 20
odp. 2    ->    téma 6 s váhou 30
                téma 7 s váhou 70
```

4.3.6 Implementace pluginu s novým typem otázky

Jak již bylo řečeno, navrhl jsem nový typ otázky (*question type*) a ten se v Moodle realizuje jako zásuvný modul (plugin). Pluginy pro jednotlivé typy otázek se v Moodle nachází v adresáři

```
./question/type
```

ve kterém má každý typ otázky svůj adresář s několika soubory. V našem případě se jedná o soubory shrnuté v tabulce 4.2.
Nový zásuvný modul jsem vytvořil zkopírováním souborů modulu *multichoice*, tedy otázky typu *úloha s výběrem odpovědí*. Poté jsem provedl v souborech potřebné úpravy. V první fázi jsem přejmenoval soubory, třídy a některé jejich metody tak, aby místo původního *multichoice*obsahovaly potřebné *multichoicefdb*, tedy označení nového zásuvného modulu.
Poté jsem začal upravovat soubor `edit_multichoicefdb_form.php` a v něm třídu `question_edit_multichoicefdb_form`, která popisuje editační formulář otázky. Moodle pracuje s formuláři tak, že se nejdřív vytvoří instance třídy `moodleform`, ta se naplní datovou strukturou popisující jednotlivá pole formuláře a teprve na základě obsahu této instance se vygeneruje HTML kód formuláře. A právě takto funguje metoda

`definition_inner` třídy `question_edit_multichoicefdb_form`, tato metoda pouze naplňuje instanci editačního formuláře datovou strukturou. Bylo tedy potřeba doplnit editační políčka pro volbu příslušejících témat ke každé odpovědi:

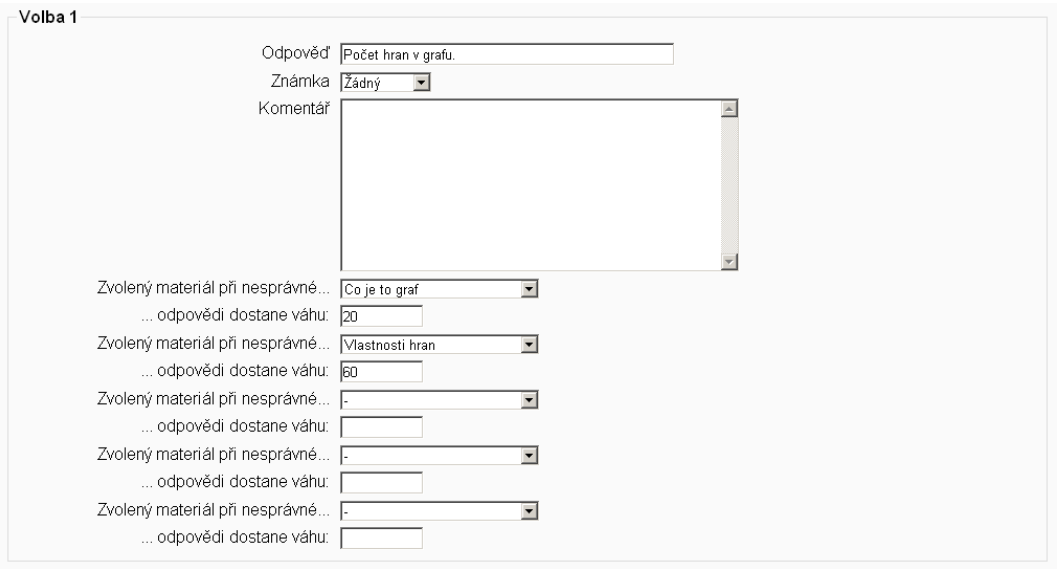
```
$resources = array(0 => '-') + get_records_menu('resource',
                                'course', $this->coursefilesid,
                                'name', 'id,name');
...
for ($i = 1; $i <= 5; $i++) {
    $repeated[] =& $mform->createElement('select',
                   'fdb['.$i.'][id_resource]',
                   get_string('feedbackres', 'qtype_multichoicefdb'),
                   $resources);
    $repeated[] =& $mform->createElement('text',
                   'fdb['.$i.'][weight]',
                   get_string('feedbackwgh', 'qtype_multichoicefdb'),
                   array('size' => 10));
}
...
$this->repeat_elements($repeated, $repeatsatstart, $repeatedoptions,
                       'noanswers', 'addanswers', QUESTION_NUMANS_ADD,
                       get_string('addmorechoiceblanks',
                       'qtype_multichoicefdb'));
```

První řádek načte z databáze seznam všech témat daného kurzu, aby bylo možné z témat volit pomocí HTML select-listu. Přidává také možnost nezvolit žádné téma (`array(0 => '-')`). Smyčka o několik řádků dále vytvoří ke každé odpovědi určitý počet (5) HTML select-listů pro volbu tématu a jeho váhy k dané odpovědi. Pole `$repeated` již obsahuje základní formulářové prvky pro editaci jedné odpovědi (text odpovědi atd.) a já k němu pouze přidávám několik dalších prvků. Výsledná podoba formuláře je vidět na obrázku 4.3.
V HTML kódu editačního formuláře se vytvořily dvojice select-list a textové políčko, které představují volbu tématu a jemu příslušející váhu:

```
<select name="fdb[3][id_resource][0]" ... >
<option value="0">-</option>
<option value="8">...</option>
...
</select>

<input size="10" name="fdb[3][weight][0]" type="text" ... />
```

V obou případech vidíme, že název formulářového prvku odpovídá třírozměrnému poli (a po odeslání se v PHP také jako třírozměrné pole vytvoří).
Číslo v první závorce říká, o kolikáté téma k dané odpovědi jde (číslováno od jedné). Chceme-li například k jedné odpovědi přiřadit tři témata a jejich váhy, navolíme je ve formulářových prvcích s různým číslem v první závorce.
Druhá závorka obsahuje textovou hodnotu a pouze nám říká, zda se jedná o prvek s volbou tématu nebo o políčko pro zadání váhy tématu.

Číslo ve třetí závorce říká, ke kolikáté odpovědi (číslováno od nuly) prvek patří. Tuto třetí závorku Moodle doplňuje automaticky u těch částí formuláře, které se opakují víckrát po sobě v jinak stejné podobě (viz řádek `$this->repeat_elements`).
Například výše uvedená část HTML kódu představuje třetí téma u první odpovědi.



Obrázek 4.3: Část formuláře pro editaci odpovědí

Tím máme tedy vytvořený formulář. Dále je potřeba doplnit kód v metodě `set_data`, která formulář naplňuje daty, pokud uživatel edituje otázku, která již existuje v databázi a tedy by měl být formulář předvyplněn. Tato data jsou načtena v objektu `$question`, který vstupuje do metody jako argument. Cílem celé metody je doplnit objekt `$question`, který již obsahuje základní hodnoty společné všem typům otázek (název a text otázky atd.) o informace specifické pro daný typ otázky (v našem případě budeme objekt doplňovat o informace o možných odpovědích a hlavně o informace o propojení s tématy).

```
$answers = $question->options->answers;
...
$key = 0;
foreach ($answers as $answer){
    $default_values['answer['.$key.']'] = $answer->answer;
    ...
    if (isset($answer->resourcefdb)) {
        $i = 1;
        foreach ($answer->resourcefdb as $id_resource => $weight) {
            $default_values['fdb['.$i.'][id_resource]['.$key.']']
             = $id_resource;
            $default_values['fdb['.$i.'][weight]['.$key.']']
             = $weight;
            $i++;
        }
    }
```

```
    $key++;
}
...
$question = (object)((array)$question + $default_values);
```

Za povšimnutí stojí řádky, v nichž se naplňuje pole `$default_values`, jehož klíčí jsou názvy proměnných odpovídajících názvům formulářových prvků v HTML. Díky tomu Moodle příště nezobrazí formulářová políčka prázdná, ale vyplněná hodnotami s odpovídajícím názvem.
Vzniká otázka, kde se naplnil objekt `$question`, než byl do metody `set_data` předán. Moodle ho opět předvyplnil základními údaji, které jsou všem otázkám společné (text otázky, bodové ohodnocení atd.) a poté ho předal metodě `get_question_options` třídy `question_multichoicefdb_qtype` v souboru `questiontype.php`. Zde je její důležitá část:

```
$question->options = get_record('question_multichoicefdb', 'question',
                                $question->id);
...
$triplets = explode(',', $question->options->answerfdb);
foreach ($triplets as $triplet) {
    list($id_answer, $id_resource, $weight) = explode(':', $triplet);
    if (isset($question->options->answers[$id_answer])) {
        $question->options->answers[$id_answer]->resourcefdb[$id_resource]
         = $weight;
    }
}
```

Nejdřív se z databáze načtou údaje specifické pro náš typ otázky včetně popisu propojení odpovědí a témat v podobě trojic popsaných v 4.3.5. Ty se postupně rozloží na jednotlivé údaje, které se vkládají do objektu `$question`.
A konečně přichází na řadu poslední potřebná část skriptu – kód, který uloží upravená data z formuláře zpět do databáze. K tomu slouží metoda `save_question_options` třídy `question_multichoicefdb_qtype`. Do ní opět vstupuje objekt `$question` s aktuálními daty z formuláře. Do metody jsem doplnil tyto řádky:

```
...
$answerfdb = array();
foreach ($question->fdb as $position) {
    foreach ($position['id_resource'] as $key => $id_resource) {
        if ($id_resource != 0 && isset($answers[$key])) {
            $id_answer = $answers[$key];
            $weight = @$position['weight'][$key];
            if ($weight > 0) {
                $answerfdb[] = $id_answer.':'.$id_resource.':'.$weight;
            }
        }
    }
}
```

```
...
$options->answerfdb = implode(",",$answerfdb);
...
```

V tomto kódu je postupně rozebíráno pole `$question->fdb`, což je ono třírozměrné pole zmíněné výše. Na nejvyšší úrovni se prochází podle pozice tématu u otázky, to znamená že nejdřív se projdou první pozice všech odpovědí, potom druhé atd. Toto pořadí nemá žádný zvláštní význam, pořadí tématu v odpovědi bylo zvoleno jako první rozměr pole proto, že třetí rozměr je doplněný samotným Moodlem a druhý rozměr má jen dvě možné hodnoty (rozlišuje mezi id tématu a váhou tématu) a je výhodnější mít ho na druhém místě, umožní to psát elegantnější kód.
Pro každou pozici tématu u odpovědi pracujeme jen s dvourozměrným polem `$position`, které obsahuje pole identifikátorů témat (`$position['id_resource']`) a pole příslušných vah (`$position['weight']`). Obě tato pole procházíme ve vnitřní smyčce, ve které už jen převedeme pořadí odpovědi v otázce (`$key`, číslo vzrůstající od jedničky po jedné) na databázové id odpovědi pomocí pole `$answers`, které bylo v metodě vytvořeno již dříve. Nyní, uvnitř smyčky, známe všechny potřebné informace, můžeme si vytvořit trojici a odložit ji do pole `$answerfdb`, které ve finále spojíme v řetězec a uložíme jej do databáze.

4.3.7 Implementace vyhodnocení testu

V kapitole 4.3.4 bylo nastíněno, jak se bude postupovat při určování, která témata se mají doporučit studentovi ke zopakování a s jakou závažností. Tento výsledek by se měl zobrazit studentovi po dokončení testu při zobrazení výsledků a správných odpovědí. O to se stará skript `./mod/quiz/review.php`. Bohužel tuto funkcionalitu nelze dopsat jako samostatný plugin, je třeba upravit přímo zmíněný soubor.
Nejdříve je třeba získat celkové informace o *všech* otázkách v testu. Skript `review.php` v normálním případě používá stránkování (např. po třech otázkách na stránku) a načítá informace pouze o těch otázkách, které bude zobrazovat. My však potřebujeme k výpočtu znát všechny otázky. K zjištění potřebných informací slouží první část našeho skriptu:

```
if ($showall) {
  // všechny otázky jsou již načteny, nemusíme je načítat sami
  $questions_all =& $questions;
  $states_all    =& $states;
} else {
  // zjistíme id všech otázek v testu a provedeme dotaz do databáze
  $pagelist_all = quiz_questions_in_quiz($attempt->layout);
  $sql = "SELECT q.*, i.grade AS maxgrade, i.id AS instance".
         "  FROM {$CFG->prefix}question q,".
         "       {$CFG->prefix}quiz_question_instances i".
         " WHERE i.quiz = '$quiz->id' AND q.id = i.question".
         "   AND q.id IN ($pagelist_all)";
  // přečteme doplňující informace a okamžitý stav otázek
  $questions_all = get_records_sql($sql);
  get_question_options($questions_all);
  $states_all = get_question_states($questions_all, $quiz, $attempt);
}
```

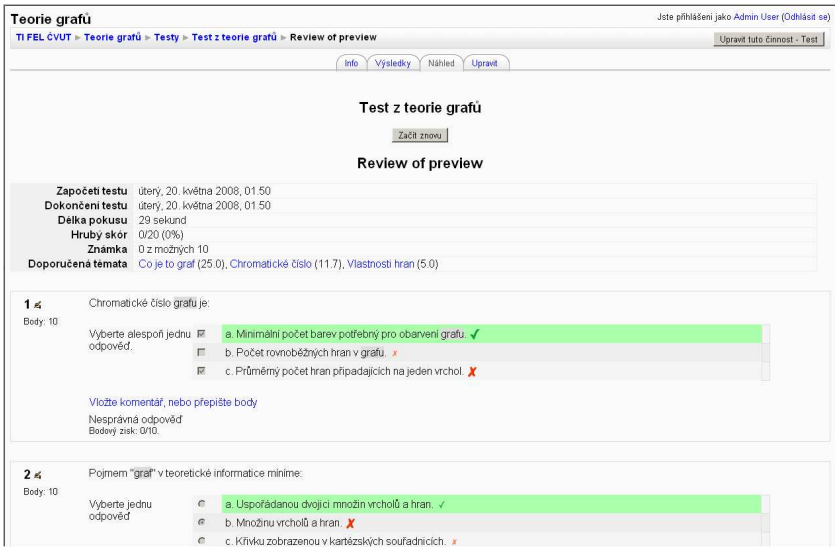
Je-li vypnuté stránkování, můžeme rovnou použít již načtené informace o otázkách. V opačném případě si otázky sami načteme pomocí SQL dotazu a doplníme potřebné údaje. Informace se skládá ze dvou složek – pole `$questions_all` obsahuje obecné informace o otázkách v testu (typ otázky, bodové ohodnocení, možné odpovědi), zatímco pole `$states_all` obsahuje údaje o studentových odpovědích při tomto konkrétním pokusu o zdolání testu.
Následuje část kódu, která odpovídá kroku 1 z kapitoly 4.3.4:

```
$q = array();
$body_celkem = 0;
// cyklus přes všechny otázky
foreach ($questions_all as $id_question => $question) {
  // zajímáme se pouze o otázky typu multichoicefdb
  if ($question->qtype == 'multichoicefdb') {
    $q[$id_question]['sz'] = 0;
    $q[$id_question]['body'] = $question->maxgrade;
    $body_celkem += $question->maxgrade;
    $answers = $question->options->answers;
    $responses = $states_all[$id_question]->responses;
    // pro každou možnou odpověď na otázku
    foreach ($answers as $key => $answer) {
      $q[$id_question]['sz'] += abs($answer->fraction);
      // pokud je otázka zodpovězena nesprávně...
      if (in_array($answer->id, $responses) xor ($answer->fraction > 0)) {
        // ... a je pro ni definována množina témat...
        if (isset($answer->resourcefdb)) {
          // přidáme odpověď do struktury otázky
          $q[$id_question]['answer'][$answer->id] = array(
            'resourcefdb' => $answer->resourcefdb,
            'fraction'    => $answer->fraction
          );
        }
      }
    }
    // pokud byla otázka správně zodpovězena, není třeba ji přidávat do $q
    if (empty($q[$id_question]['answer'])) {
      unset($q[$id_question]);
    }
  }
}
```

Tento úsek kódu vytvoří zcela nové pole `$q`, do kterého ukládá pouze ty nejdůležitější informace (potřebné v kroku 2 – viz dále), a to pouze o otázkách typu `multichoicefdb` a pouze tehdy, pokud na otázku student odpověděl nesprávně.
Další část kódu (krok 2 v kapitole 4.3.4) vypočítá trestné body jednotlivých témat:

```
// inicializace pole s trestnými body (id => body)
$t = array();
```

```
// pro každou zbylou otázku a její zbylé odpovědi (z kroku 1)
foreach ($q as $id_question => $question) {
  foreach ($question['answer'] as $id_answer => $answer) {
    // pro každé téma přiřazené k dané (nesprávně zodpovězené) odpovědi
    foreach ($answer['resourcefdb'] as $id_theme => $tb) {
      // zvyš trestné body tématu podle dříve odvozeného vztahu
      if (!isset($t[$id_theme])) {
        $t[$id_theme] = 0;
      }
      $t[$id_theme] += $tb * abs($answer['fraction']) * $question['body']
                           / ($question['sz'] * $body_celkem);
    }
  }
}
```



Obrázek 4.4: Zpětná vazba po absolvování testu

Nakonec zbývá už jen témata v přehledné formě vypsat:

```
// seřadíme témata podle trestných bodů sestupně
arsort($t);
// načteme z databáze informace o výukových textech
$themes = get_records('resource', '', '', 'name', 'id,name');
$results = array();
// sestavíme pro každé téma odkaz a výpis zaokrouhleného počtu bodů
foreach ($t as $id_theme => $tb) {
  $results[] = '<a href="'.$CFG->wwwroot.'/mod/resource/view.php?r='.$id_theme.'">'.
               $themes[$id_theme]->name.
               '</a> ('.number_format($tb, 1).')';
}
// sestavíme výsledný text připravený k vypsání
$result = implode(', ', $results);
```

Nyní si může student kdykoli po absolvování testu na stránce s vyhodnocením přečíst, která témata se mu doporučují k zopakování a s jakou mírou závažnosti, viz obr. 4.4.

4.4 Podpora grafových algoritmů

Implementace frameworku pro demonstraci a simulaci grafových algoritmů je diplomovou prací jiného řešitele. Výsledkem jeho práce je Java applet, který studentovi zprostředkuje a názorně předvede principy a průběh jednotlivých algoritmů.
Hlavními vlastnostmi celého systému jsou:

- Jedná se o Java applet, který může běžet jak v rámci webu, tak jako samostatná Java aplikace v desktopovém prostředí.
- Aplikace umožňuje uživateli navrhnout si libovolný graf s libovolně propojenými uzly (ať už ručním nakreslením nebo předáním v textové podobě) a poté nad tímto grafem spustit jeden z implementovaných algoritmů. Průběh algoritmu může libovolně zastavovat a krokovat a sledovat aktuální pozici v zobrazeném pseudokódu.
- V grafech jsou podporovány orientované i neorientované hrany (v jednom grafu pouze jeden druh), rovnoběžné hrany a smyčky (v orientovaném grafu se navíc rozlišují rovnoběžné orientované i neorientované hrany).
- Uzly i hrany jsou popsány několika základními vlastnostmi (barva, označení, unikátní název), programátor algoritmu může doplňovat další potřebné vlastnosti (typicky např. délku hrany). Navíc může určit, které vlastnosti smí měnit uživatel a které jen samotná aplikace. Navíc lze uživateli zakázat některé zásahy do grafu, např. změnu struktury v průběhu demonstrace algoritmu.
- Celý systém se nachází v jediném souboru (`jar` archivu), navíc však ještě potřebuje adresář s několika obrázky používanými jako ikony na tlačítkách. Jednotlivé algoritmy jsou řešeny jako samostatné `jar` archivy, které základní framework používají. Pro přidání nového algoritmu tedy stačí nahrát archiv s příslušnými třídami a spustit jej, program už si sám najde framework a použije jej.
- Systém je připraven pro podporu vícejazyčnosti, ta však zatím není plně implementovaná a počítá se s ní do budoucna.

Jak tedy použít tuto aplikaci v Moodle? Jednou možností by bylo vkládat Java applet přímo do webové stránky s popisem daného algoritmu, kde by se zobrazil mezi textem. Druhou možností je vkládat do textu pouze odkaz, který by otvíral nové okno s appletem. Vzhledem k tomu, že aplikace potřebuje pro svůj běh poměrně velkou plochu obrazovky, rozhodl jsem se pro druhou variantu. V samostatném okně nebude applet „překážet“ v textu a na větší ploše bude práce s ním pro uživatele pohodlnější.
Jako příklad popíšu postup zprovoznění celého frameworku s algoritmem *prohledávání grafu do šířky* (BFS – *Breadth-first search*). Implementace tohoto algoritmu se nachází v souboru `bfs.jar`. HTML kód pro vložení appletu bude vypadat následovně:

```
<applet code="ti/bfs/Bfs.class" archive="bfs.jar"
        codebase="." width="950" height="550">
  <param name="lang" value="czech" />
</applet>
```

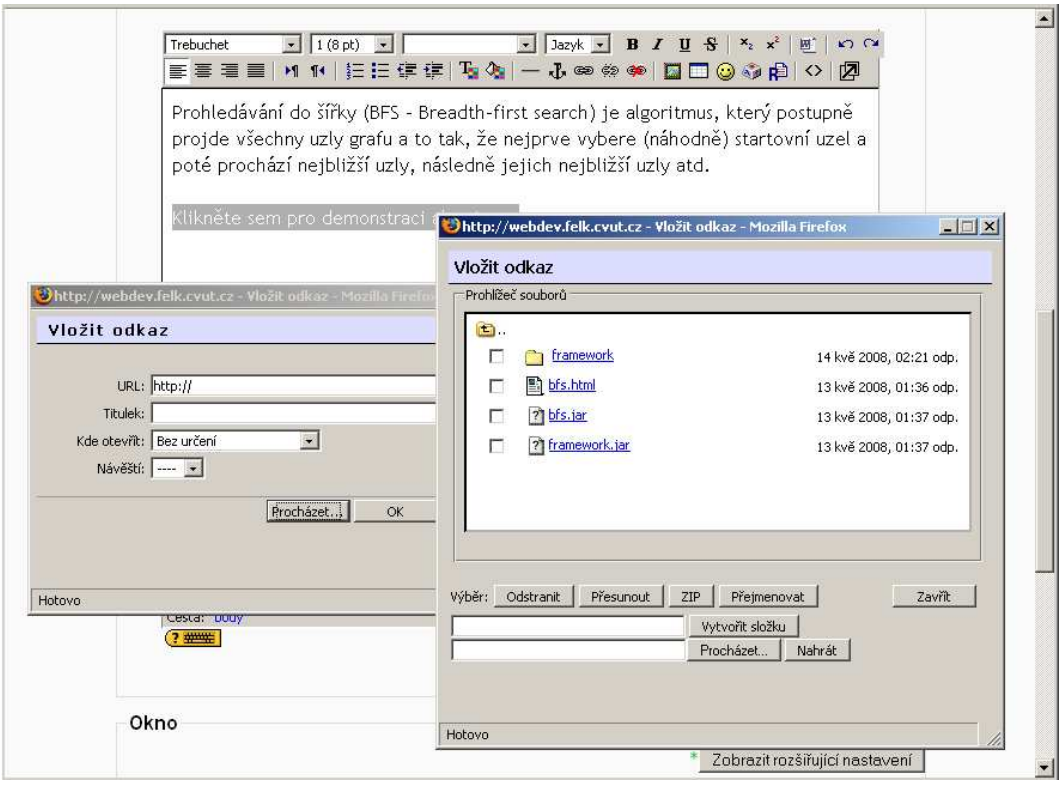
Parametr `code` je odkazem na hlavní třídu v archivu `bfs.jar`, šířka a výška jsou navrženy autorem appletu a zajišťují optimální zobrazení ve webovém prohlížeči na (dnes již obvykle minimálním – viz např. statistiku `http://www.w3schools.com/browsers/browsers_display.asp`) rozlišení monitoru 1024×768 obrazových bodů.
Pro jeden algoritmus budeme tedy potřebovat příslušný `jar` archiv a HTML soubor, který applet zobrazí. Tyto soubory, včetně archivu s frameworkem a adresářem s obrázky na tlačítka potřebujeme nahrát na server do systému Moodle. Moodle disponuje velmi podařeným správcem souborů, který umožňuje vytvářet pro daný kurz libovolnou adresářovou strukturu, nahrávat, mazat a přejmenovávat soubory jakéhokoli typu a komprimovat a dekomprimovat soubory. Některé druhy souborů lze navíc přímo editovat (*.html, *.txt a další). Nevýhodou je, že každý kurz má svou vlastní strukturu souborů, takže nelze např. applet nahraný v kurzu A používat v kurzu B, musí se nahrát zvlášť. Máme však možnost se v B odkazovat do výukového materiálu kurzu A, v němž se odkaz na applet vyskytuje.

Obrázek 4.5: Archiv s appletem po nahrání na server

Obrázek 4.6: Struktura souborů appletu po rozbalení na serveru

V Moodle jsem vstoupil jako administrátor do příslušného kurzu a v levém menu zvolil položku *soubory*, tím jsem se dostal do správce souborů. Vytvořil jsem složku `applet`, do které budu nahrávat celou aplikaci. Poté jsem si (pro co nejjednodušší upload) na disku zkomprimoval všechny potřebné soubory (framework, adresář s jeho obrázky, archiv s BFS algoritmem a příslušnou HTML stránku) do `zip` archivu a ten jsem nahrál na server (přes formulář ve správci souborů) do složky `applet`, viz obr. 4.5. Poté jsem ho na serveru dekomprimoval kliknutím na *rozbalit*. Výsledná struktura je vidět na obr. 4.6. Applet je tedy nahrán, nyní je potřeba do učebních textů vložit odkaz na něj. Pro editaci textů používá Moodle vlastní *WYSIWIG* editor, jehož funkce pro vkládání obrázků nebo

odkazů jsou úzce propojeny se správcem souborů. V editoru jsem tedy označil text, který se měl stát odkazem a kliknutím na tlačítko pro vložení odkazu jsem vyvolal dialog (viz obr. 4.7). V tomto dialogu zvolím otvírání do nového okna a kliknutím na *procházet* zobrazím zjednodušeného správce souborů, v němž vyberu dříve nahraný soubor `bfs.html`. Tím je odkaz v textu vytvořen a applet lze při prohlížení výukového materiálu spustit (viz 4.8).



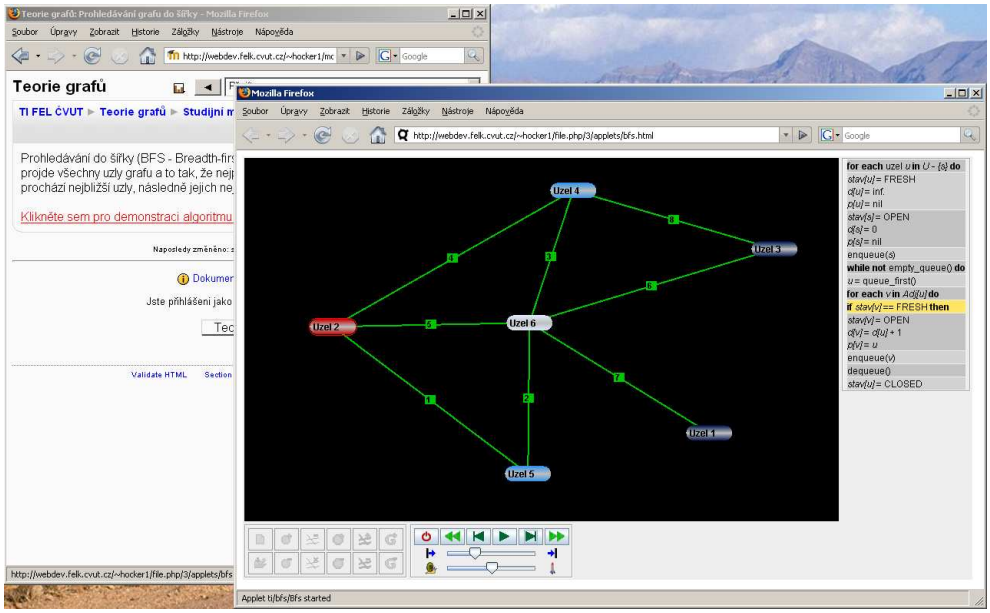
Obrázek 4.7: Dialog pro vložení odkazu na soubor s appletem

Může nastat případ, že tvůrce kurzu nemá k dispozici WYSIWYG editor, například pokud používá prohlížeč, který není editorem podporován a proto se v něm nespustí (bezpečně se spustí pouze v prohlížečích Internet Explorer od verze 5.5 nebo na prohlížečích stojících na jádru *Gecko*, např. FireFox) nebo proto, že má z jakýchokli důvodů vypnutý javascript (editor je aplikace napsaná v javascriptu). V takovém případě se mu místo editoru zobrazí „obyčejné“ formulářové okno typu *textarea*, v němž tvůrce upravuje přímo zdrojový HTML kód stránky. Do něj pak jako odkaz na applet stačí vložit kód:

```
<a href="/~hocker1/file.php/3/applets/bfs.html"
   title="Kliknutím zobrazíte demonstrační applet v novém okně">
Text odkazu
</a>
```

Popišme si z čeho přesně se skládá URL odkazu:

- `/~hocker1/file.php` je cesta ke skriptu, který předává prohlížeči soubory z adresářové struktury daného kurzu. „/“ na začátku říká, že cestu zadáváme od kořene



Obrázek 4.8: Ukázka běžícího appletu

serveru (`http://webdev.felk.cvut.cz/`), místo toho bychom klidně mohli použít absolutní cestu (`http://webdev.felk.cvut.cz/~hocker1`).

- `/3` obsahuje id výukového kurzu v databázi. Toto id lze zjistit z URL například po vstupu na úvodní stránku kurzu (`http://webdev.felk.cvut.cz/~hocker1/course/view.php?id=3`).

- `/applets/bfs.html` je cesta v adresářové struktuře daného kurzu.

Do systému Moodle tedy lze nahrát applety i další pomocné soubory a odkazovat se na ně ať už pomocí nástrojů k tomu určených ve WYSIWYG editoru, tak i v méně pohodlném případě, bez editoru.

5 Testování

5.1 Zobrazování matematických výrazů

Jednou ze složek systému, které je třeba otestovat, je rozšíření pro zobrazování matematických zápisů. Dokumentaci možností systému MimeTEX lze nalézt na `http://www.forkosh.com/mimetex.html`, odkud lze také brát zkušební výrazy. Samotný MimeTEX není třeba testovat, je dobře vyzkoušen svými vývojáři. Otestovat je však třeba filtr, který v textech hledá a nahrazuje TEXové výrazy. Ty se ohraničují znaky `$$`. Je tedy třeba otestovat, zda nějaká kombinace těchto znaků neuvede filtr v omyl. Po vyzkoušení mnoha různých variant jsem zjistil následující:

- Pokud je výraz započat, ale není ukončen, pak se vůbec nenahrazuje obrázkem, to je v pořádku, nejedná se o korektní výraz.
- Pro zapsání znaku `$` je třeba použít zápis `\$`.
- Sekvenci `\$\$` filtr správně rozpozná jako dva znaky, které *neohraničují* TEX zápis.
- Sekvence `$$$` (mimo TEXový zápis) se převede na `$$`, což je nesprávné.
- Při zápisu tří a více znaků `$` za sebou uvnitř výrazu se tyto znaky převedou na HTML entity `&x0024` a ty se vzápětí vykreslí v této podobě. Filtr totiž na začátku filtrování převádí sekvence tří a více znaků `$` na entity proto, aby první (nebo poslední) dva z nich nepovažoval za hranici výrazu, nepočítá však s tím, že tato sekvence může být *součástí* výrazu.
- Při zápisu např. `$$ \$$$` filtr považuje za matematický výraz pouze znak `\`, zápis je totiž nejednoznačný. Ohraničující znaky `$$` by měly být na obou stranách od samotného výrazu oddělené mezerou.

TEX filtr lze prohlásit za dostatečně funkční. Je třeba mít na paměti, že ohraničující značky se od výrazu musí oddělovat alespoň jednou mezerou. Problémy se sekvencí tří a více znaků `$` nejsou pro nás důležité, takovéto sekvence se obvykle v matematických výrazech nevyskytují.

5.2 Systém zpětné vazby testů

Je třeba vyzkoušet, zda se systém chová tak, jak jsem jej popsal v 4.3.4, to znamená zda jsou splněny tyto vlastnosti:

- Pokud jsou nesprávné odpovědi u otázek provázaných s tématem `T`, nezáleží trestné body tohoto tématu na správnosti odpovědí na otázky provázané s tématem `V`.
- Při správném zodpovězení všech otázek se nepřičítají žádnému tématu trestné body.
- Při jedné nesprávné odpovědi na otázku `Q` provázanou s tématem `T` dostane toto téma menší počet trestných bodů, než při nesprávných odpovědích na více otázek (včetně `Q`) provázaných s `T`.

Otázka	**Odp. A**	**Odp. B**	**Odp. C**	**Odp. D**
1	T1(50), T2(50) *	T1(50), T3(50) *	T1(50), T4(50)	T1(50), T5(50)
2	T5(10), T6(70)	T5(10), T6(70) *	T5(20), T6(80) *	T5(30), T6(10)
3	T1(50), T2(10) *	T1(50), T2(50)	T1(60), T2(50)	T1(50), T2(50)
4	T3(30), T4(50)	T4(30), T5(50) *	T3(50), T6(50) *	T5(50), T6(50) *
5	T8(80), T7(20)	T6(50), T5(80)	T4(50), T3(40) *	T2(10), T1(60) *
6	T1(10), T8(50)	T2(10), T7(50) *	T3(20), T6(50)	T4(50), T5(50)

Tabulka 5.1: Test pro ověření vlastností zpětné vazby

- Otázky s větším bodovým ohodnocením produkují při nesprávné odpovědi vyšší počet trestných bodů, podobně také odpovědi s vyšší absolutní hodnotou známky.

Za účelem testování těchto vlastností jsem vytvořil cvičný kurz, v němž se nachází osm různých témat a test se šesti otázkami. Každá otázka obsahuje čtyři odpovědi a každá odpověď je provázána se dvěma tématy. Podobu testu zachycuje tabulka 5.1. V prvním sloupci najdeme číslo otázky a v dalších sloupcích jsou popsány odpovědi – výraz `Tx(b)` určuje, se kterými tématy je odpověď provázána a kolika trestnými body. Odpovědi označené `*` jsou správné, ostatní nesprávné. Body za všechny otázky jsou 1 a známky u odpovědí jsou vyváženy tak, aby správné odpovědi dávaly v součtu 100 % a nesprávné -100 %. Cvičný test jsem pro ověření jednotlivých bodů zkoušel vyplňovat vždy tak, abych si danou vlastnost ověřil i abych ji zkusil vyvrátit.

- První vlastnost jsem testoval tak, že jsem v testu nesprávně odpověděl na `1.A`, zatímco na ostatní otázky správně. Dostal jsem bodové ohodnocení témat `T1(2.1), T2(2.1)`. Poté jsem vyplnil test znovu, navíc jsem však nesprávně odpověděl na `2.B`, resp. další odpovědi, které nejsou provázány s `T1, T2`. Ohodnocení `T1, T2` se nezměnilo, vlastnost je tedy zachována. Pokus jsem opakoval s dalšími obdobnými variantami.

- Otestovat druhou vlastnost je triviální. Zodpověděl jsem správně na všechny otázky a zkoušel jsem některé otázky z testu vyřazovat. Výsledek byl zcela správný a očekávaný – systém nerozdal žádné trestné body a nedoporučil tedy k zopakování žádná témata.

- Třetí vlastnost jsem tesoval tak, že jsem nejdřív odpověděl nesprávně na `1.A`, dostal jsem bodové ohodnocení témat `T1(2.1), T2(2.1)`. Poté jsem navíc přidal nesprávnou odpověď `3.A` a dostal ohodnocení `T1(6.3), T2(2.9)` nebo nesprávnou odpověď `1.B` a dostal `T1(4.2), T3(2.1), T2(2.1)`. Z toho je vidět, že i třetí vlastnost funguje správně, trestné body u témat s více nesprávnými odpověďmi se kumulují.

- Poslední vlastnost jsem ověřil úpravou obodování otázky `1` (resp. ohodnocením odpovědi `1.A`). Nejdřívv jsem bodování otázky zvýšil z 1 na 5 a nesprávnou odpovědí `1.A` získal `T1(6.3), T2(6.3)`. Ohodnocení je oproti původnímu stavu trojnásobné. To není náhoda. Dříve měla otázka 1 bod a tedy váhu v rámci celého testu 1/6. Při ohodnocení 5 bodů měla otázka váhu $5/10 = 1/2$, tedy trojnásobek. Obdobně jsem otestoval i změnu při zvýšení známky otázky `1.A` z 50 % na 80 %. Dospěl jsem

k trestným bodům `T1(3.3), T2(3.3)`. Pokus jsem opakoval s různými dalšími variantami, bodovými ohodnoceními a nesprávnými odpověďmi. Poslední vlastnost jsem tedy také ověřil.

Systém se tedy chová tak, jak bylo předpokládáno.

Závěr

Cílem práce bylo nasadit, otestovat a naplnit zkušebními daty e-learningový systém pro výuku Teoretické informatiky a podobných předmětů. Po důkladném zkoumání open-source řešení jsem zvolil systém Moodle, nainstaloval jej na školním serveru pro testování webových projektů a upravil tak, aby splňoval specifické požadavky.
V rámci rešerší jsem zjistil, že ČVUT už systém Moodle začíná využívat na adrese `http://ocw.cvut.cz/moodle/`. I když předměty Katedry počítačů nejsou zatím příliš hojně zastoupeny, dá se předpokládat, že pokud začne Katedra e-learning více využívat, využije právě služeb tohoto serveru. Až se tak stane, bude možné snadno využít poznatků z mojí práce k tomu, aby systém co nejlépe vyhovoval výuce daných předmětů. Jedná se především o zápis matematických výrazů a nasazení appletu pro demonstraci grafových algoritmů.
Ve své práci jsem také navrhl rozšířený systém pro zpětnou vazbu testů vůči studentům. Ověřil jsem, že tento systém funguje správně tak, jak jsem jej navrhnul a může být nasazen na `http://ocw.cvut.cz/moodle/`, kde by mohl být otestován v reálném provozu.
Práce splnila své cíle, nicméně dálo by se na ni navázat dalšími úkoly:

- zvážit a případně implementovat systém zpětné vazby na další typy otázek, ne jenom na *výběr z více odpovědí*,
- prozkoumat detailně normu AICC nebo SCORM a umožnit export a import otázek i s informacemi o zpětné vazbě.

Literatura

[1] Michal Novák. *E-learning - nástroje pro tvorbu a řízení výuky.* Praha, 2007. Vedoucí bakalářské práce Ing. Jiří Vaněk, Ph.D.

[2] Mark Nichols. *A theory for eLearning.* Palmerston North, New Zealand, 2003. Přednáška.

[3] Doc. Ing. Čeněk Celer, CSc. *Může nahradit e-learning klasickou výuku?* FSE UJEP Ústí nad Labem. Zpráva.

[4] *Using Moodle Chapter 5: Quizzes* [online]. Dostupný z WWW: <`http://docs.moodle.org/en/Using_Moodle_book`>.

[5] F. E. Terry Anderson, editor. *Theory and Practice of Online Learning.* Athabasca University, 2004.

[6] J. Vejvodová. *Tvorba a tutorování on-line kurzu.* FI MUNI. Záznam přednášky dostupný z WWW: <`http://video.fi.muni.cz/`>.

[7] Hana Stříteská. *Historie e-learningu v České republice.* Článek dostupný z WWW: <`http://www.fi.muni.cz/usr/jkucera/pv109/2003p/xstrites.htm`>.

[8] Matthew Fox. *50 Ideas for Free E-learning.* Kineo, 2005.

[9] Bořivoj Brdička. *Jaké mělo být vzdělávání v roce 1975?* Učitelský pomocník [online]. 2008. Dostupný z WWW: <`http://www.spomocnik.cz/index.php?id_document=2213`>.

[10] Doc. Ing. Boris Šimák, CSc. *Rozvojový program MŠMT pro rok 2005.* FEL ČVUT, 2005.

[11] Phil Bourne *E-learning: Theory to practice.* Greensward College.

[12] P. Zídek. *Mixování tradičního přístupu s novými technikami pro zvýšení efektivity e-learningu.*