VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

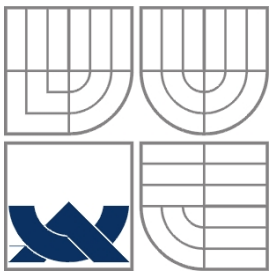
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MUTLIMEDIA

AUTOMATICKÉ VYHODNOCOVÁNÍ E-LEARNINGOVÝCH TESTŮ

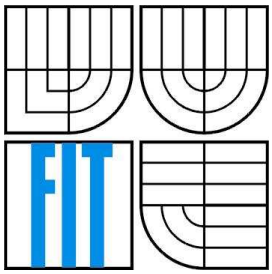
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE Bc. Jan Hort
AUTHOR

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

AUTOMATICKÉ VYHODNOCOVÁNÍ E-LEARNINGOVÝCH TESTŮ

AUTOMATIC EVALUATION OF E-LEARNING TESTS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. Jan Hort
AUTHOR

VEDOUCÍ PRÁCE — doc. RNDr. Pavel Smrž Ph.D.
SUPERVISOR

BRNO 2008

Abstrakt

Tato práce se zabývá výzkumem automatického vyhodnocování e-learningových testů. Tato technika je užitečná pro oblasti počítačem řízené vyučování, e-elearningu a inteligentních výukových systémů. Rozšiřuje možnosti testování znalostí studentů v online prostředí. Práce se také zabývá normami pro oblast vzdělávání a inteligentními vývojovými systémy. Také stručně seznamuje s některými projekty z oblasti výukových inteligentních systémů.

Klíčová slova

E-learning, inteligentní výukové systémy, doména (oblast) znalostí, studentský model, učitelský model, poznávací učení, přirozený jazyk, diagram případů užití, diagram tříd, sekvenční diagram, diagram balíčků, Java kontejner, servlet, jsp stránka, algoritmus, návrh systému, implementace.

Abstract

The thesis is dealing with research of automatic evaluation of e-learning tests. That technology is useful for domain of computer aided learning and e-learning and intelligent tutoring systems. This is extending possibility of online testing the student’s knowledge. This thesis is also dealing with norms for learning and intelligent tutoring systems. Briefly introduce some project of domain of intelligent tutoring systems.

Keywords

E-learning, intelligent tutoring systems, domain of knowledge, student model, teaching model, cognitive learning, natural language, use case diagram, class diagram, sequence diagram, package diagram, Java container, servlet, jsp page, algorithm, design of system, implementation.

Citace

Jan Hort: Automatické vyhodnocování e-learningových testů, diplomová práce, Brno, FIT VUT v Brně, 2008

Automatické vyhodnocování e-learningových testů

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením doc. RNDr. Pavla Smrže Ph.D.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Jméno Příjmení
Datum

Poděkování

Děkuji doc. RNDr. Pavlu Smržovi Ph.D. za trpělivé a důkladné vedení při řešení této práce.

Obsah

1 Úvod

Inteligentní výukové systémy jsou předmětem dlouholetého výzkumu poměrně velké komunity vědců. Hlavním rysem těchto výzkumů jsou modely a průzkumy efektivního učení. Dobrý výukový systém musí sledovat pokroky a citové pochody studenta a na základě toho adaptivně přizpůsobovat výuku. Některé systémy jsou vyvíjeny jako agentní a kladou důraz na co nevěrnější simulaci lidského učitele. Velmi důležitým aspektem je také testování studenty nabytých znalostí. Tímto aspektem se zabývá i projekt automatického vyhodnocování e-learningových testů.

Technika automatického vyhodnocování e-learningových testů je užitečná při zkoušení znalostí studentů. Jedná se o strojové vyhodnocování otázek s víceslovními odpověďmi, tedy o případy, kdy student odpovídá na otázku slovy či větami, stejně jako při písemné zkoušce vykonávané prostřednictvím papíru a tužky. Projekt je postaven jako webový systém s architekturou klient server.

K řešení projektu byla použita moderní metodika agilního programování, která je vhodná pro vývoj webových systémů a odpovídá modelu řešení semestrální práce pod dohledem a vedením vedoucího práce. K návrhu je použit modelovací jazyk UML. Implementace je provedena prostřednictvím moderních technologií, objektového programovacího jazyka Java SE, značkovacího jazyka XHTM a kaskádových stylů CSS. Projekt je multiplatformní a je určen pro běh na Java containeru TOMCAT a databázi MySQL.

Tato diplomová práce navazuje na předchozí semestrální projekt. Semestrální projekt byl zakončen prototypem webového systému, který uměl vyhodnotit zadané otázky s víceslovními odpověďmi. Cílem navazujícího diplomového projektu je optimalizace vyhodnocujícího algoritmu, rozšíření testování o nové možnosti a zavedení adaptivního chování systému při testování studenta. Prototyp systému vytvořeného v semestrálním projektu je v diplomové práci od základu přepsán a všechny testy byly vyhotoveny na novém systému. Diplomová práce neobsahuje žádné výsledky z předešlého semestrálního projektu.

2 Cíle práce

2.1 Hlavní cíle práce

Cílem diplomové práce je vyvinout webový systém schopný adaptivního testování studentů. Hlavním požadavkem je, aby zkoušený student odpovídal na otázky pomocí víceslovných odpovědí. Systém má simulovat zkoušení písemnou formou a přitom přizpůsobovat test podle odpovědí studenta. Druhotným cílem je zlepšení uživatelského prostředí. Podle výsledků prototypu vyhodnoceného v semestrálním projektu bylo slabé uživatelské prostředí největším problémem uživatelů. Systém má být vyvinut jako webová aplikace s architekturou klient - server. Klientem je běžný webový prohlížeč.

2.2 Návaznost na semestrální projekt

Ačkoli diplomová práce navazuje na semestrální projekt, celý systém byl vyvíjen zcela znovu. Důvodem je rozsáhlost změn oproti prototypu semestrálního projektu a zejména nutnost přechodu na technologii, která by umožnila kvalitnější zpracování uživatelského rozhraní. Aplikace se tak zcela změnila na prezentační i logické vrstvě. V datové vrstvě byly provedeny jen některé změny související se zavedením adaptivního procházení testem a rozšířením systému. Diplomová práce neobsahuje žádné výsledky z předchozího prototypu vytvořeného v kurzu semestrální projekt. Jediná podobnost se semestrálním projektem je v algoritmech, které vyhodnocují víceslovné odpovědi studentů (kap. 4.3.2.2). Tyto algoritmy byly pro nový systém znovuvyvinuty na principu algoritmů starého prototypu.

3 Inteligentní výukové systémy

3.1 Standardy pro vzdělávání a výuku

3.1.1 AICC

Standard „Aviation industry CBT Committee“ původně vznikl pro letecký průmysl. Později se rozšířil i do ostatních odvětví průmyslu a byl neoficiálně přijat jako standard pro online vzdělávání. Tento standard je dnes spíše historický a byl nahrazen standardem SCORM.

AICC dělí vzdělávací obsah na „assignable units“, což jsou přiřaditelné jednotky části tohoto obsahu. Tyto jednotky lze volně přiřazovat do kurzů, které lze navíc členit do bloků. Bloky lze

zanořovat do různých úrovní a mezi jednotkami i bloky kurzu lze definovat logické podmínky průchodu na základě dosažených výsledků studenta. Pro snadné vkládání kurzů by měl LMS[1] být schopen importovat strukturu kurzů (soubory s určitou příponou).

Standard může být podporován více nebo méně. Dobrý LMS by měl umět pracovat s informacemi o tom, jaká lekce je kým spuštěna, kdy byla tato lekce spuštěna, s dobou studia, dosaženými výsledky, s daty o interakci uživatele a využívat informace o uživatelských vstupech. Na LMS je možné získat certifikát potvrzující, že daný LMS podporuje standard AICC [1]. Vzhledem k zastaralosti standardu se této možnosti už příliš nevyužívá.

3.1.2 SCORM

Tento standard vytvořila organizace Advanced Distributed Learning. Shareable Content Object Reference Model je souborem specifikací a standardů, které umožňují provoz učebního obsahu vytvořeného různými nástroji podle standardu SCORM. Podmínkou je podpora tohoto standardu u daného LMS. Princip spočívá v použití objektů, ve kterých jsou uloženy části učebního obsahu, které lze vzájemně sdílet. Tímto principem je umožněno znovupoužití studijních materiálů v různých LMS. Výukové objekty jsou popsány v manifest souboru pomocí značkovací notace XML [2]. Kurz neobsahuje navigaci mezi učebními objekty, ze kterých je složen. O navigaci mezi objekty se stará samotný LMS, který musí obsahovat přehrávač SCORM kurzů SROTM RTE. Výhodou standardu oproti AICC jsou hlavně vyšší přizpůsobitelnost, prohledávatelnost, sdílení a znovupoužití obsahu kurzu [3]. Tyto výhody plynou z odlišného pohledu na strukturu kurzů. Podporu SCORM vytvořeného či zakoupeného produktu lze bezplatně ověřit zdarma pomocí aplikace Scorm Conformance Test Suit. Tu je možné zdarma stáhnout z internetu.

3.2 Inteligentní výukové systémy

3.2.1 Co je to inteligentní výukový systém

Pojem „Inteligentní výukový systém[2]“ má široký význam. Základní charakteristikou každého ITS je, že se skládá z nějakého počítačového programum jehož součástí je umělá inteligence a který je používán pro výuku. ITS se vyvinul z dřívějších systémů využívajících počítačem řízené instrukce nebo CAI[3] modelu.

Běžný ITS model je složen ze čtyř komponent. Tyto komponenty jsou: Doménový model, studentský model, učitelský model a učební prostředí či uživatelský interface. ITS se mohou velmi významně lišit v závislosti na relativní úrovní inteligence jednotlivých komponent [4]. Například ITS,

[1] LMS – Learning Management System, Systém řízení výuky.
[2] Dále jen ITS
[3] CAI – Computer Aided Instruction – Použití počítače pro výuku.

který je zaměřen na vyšší inteligenci v doménovém modelu, může generovat velké množství stále nových problémů, která mají obdobná řešení, takže student má možnosti neustálého procvičování bez toho, že by opakoval již procvičené příklady. Naopak může mít takový systém slabší možnosti v metodách samotné výuky při nižším stupni inteligence učitelského modelu. Jiný ITS může mít propracovaný učitelský model a nabízet mnoho způsobů vysvětlení probírané látky a mít slabší možnosti v jiné oblasti. Pokud více komponent systému obsahuje inteligenci, lze v něm použít homogenní nebo heterogenní reprezentaci.

Každý ITS lze klasifikovat na základě jeho základního algoritmu. Známou kategorií je tzv. model-tracing tutor, který sleduje postupy studentů při řešení a s určitou tolerancí je udržuje v mezích vhodné cesty řešení.

V oblasti výzkumu ITS je v současnosti největší výzvou doménová samostatnost. Tedy například míra znalostí, které jsou v učitelském modelu a mohou být znovupoužity v některé jiné doméně. Ačkoli z vnějšího pohledu vypadá doménová samostatnost jako základní charakteristika inteligence, mnoho expertů věří, že v každé doméně jsou některé základní pedagogické znalosti nezbytně doménově závislé. Například existují analogie ve vyučování fyziky a vyučování specifických témat ve fyzice, které nemají ekvivalenty v ostatních doménách.

Otázkou samostatnosti domény nebo míry, kolika znalostí v systému je možno využít pro podporu různorodosti otázek na straně studenta, se zatím většina systémů nezabývá.

3.2.2 Výzkum efektivní výuky

Velký technologický pokrok a stále vyšší požadavky na člověka v moderním světě kladou velké nároky na všechny úrovně vzdělání napříč celou světovou populací. Systémy ITS reprezentují důležitou třídu vzdělávací technologie, která je určena k poskytnutí maximální pomoci při vzdělávaní. Proto probíhá intenzivní výzkum, který má zlepšit jejich vlastnosti a kvalitu.

Vědci, kteří zkoumají vědy zabývající se výukou a pedagogikou, hledají základní principy lidského učení. Ve zprávě organizace National Research Council, která se nazývá „How People Learn[1]“, je sumarizován asi 30letý rozsáhlý výzkum o lidském učení.

HPL klade důraz na to, aby si učící se člověk vytvořil hlubší porozumění konceptu ve chvíli, kdy se snaží naučit nový okruh vědomostí. To znamená jít až za fakta a procedury, které slouží k zapamatování použitelnosti podmínek a dynamické modifikaci stávajících znalostí a k jejich uplatnění v nových kontextech a situacích. Když studenti získají hlubší pochopení konceptu, učí se mnohem lépe i veškerá fakta a procedury, která souvisí s danou doménou znalostí. Nemají problém pochopit souvislosti a přeložit si neznámé problémy do jim srozumitelné běžné řeči. Základním cílem je umožnit studentům spolupodílet se na jejich učení. Tedy umožnit jim ovlivnit způsob, jakým se nové znalosti budou učit. Toto pravidlo je zmiňováno i v mnoha teoriích zabývajících se výukou.

[1] Dále jen HPL

Dalším důležitým bodem HPL je zdůraznění důležitosti přemýšlení a úvahy o nabytých znalostech a důležitost správného porozumění novým znalostem. Navíc při získávání nových znalostí (faktů a postupů) student zvyšuje svou schopnost učení jako takovou. Schopnosti, které souvisí s přemýšlením jako jsou plánování, pokládání otázek, vysvětlování nebo kritizování, jsou často vidět u expertů a téměř nikdy u nováčků oboru. Aby se co nejvíc urychlil přechod člověka od nováčka oboru k expertovi, je mu třeba vytvořit vhodné učební prostředí, které posílí a podpoří jeho vzdělávací aktivity. Prioritou vývoje moderních výukových počítačových prostředí by tedy mělo být zaměření se na aktivní vzdělávání a postupy, které nutí studenta přemýšlet [5].

3.2.3 Člověk a inteligentní vzdělávání

V oblasti výuky řízené počítačem zatím nejsou známé metody, které by byly stejně efektivní jako výuka vedená profesionálním školitelem [5]. Studenti, kteří jsou školení lidským expertem jeden na jednoho, v průměru dosahují známky 2.0 a mají tedy průměr přibližně o dva stupně lepší hodnocení než studenti, kteří byli školeni v klasické třídě. Při použití nejlepších inteligentních výukových systémů dosahují ve standardním srovnání známky okolo 1.0. Naproti tomu při použití nejlepších CAI systémů, které používají počítačem řízené tutoriály a nevyužívají prvky umělé inteligence, bylo při standardním srovnání dosaženo známky okolo 4.0.

3.2.4 Proč je tutoring efektivní?

Ačkoliv neexistuje zcela přesná odpověď na otázku, proč je tutoring efektivnější než jiné formy výuky, mnoho vysvětlení se opírá o fakt, že v dobře vybalancovaném tutoring systému se student aktivně podílí na řízení výuky. Tím je myšleno, že student pracuje tolik kolik zvládne, zatímco tutor[1] mu poskytuje dostatečnou odezvu k tomu, aby se minimalizovala frustrace a případné zmatení studenta [5]. Efektivní školení také méně vysvětluje probírané znalosti a spíše se soustředí na interakci mezi studentem a tutorem. Interakce se studentem má velmi podstatný vliv na kvalitu výuky. ITS by měl být prostředkem, který vyvolává u studenta konstruktivní odpovědi. Běžný postup při vývoji ITS je, že se prvně identifikují efektivní výukové události a vzory v lidském učení a následně se simulují v ITS.

3.2.5 Klasifikace technologií ITS

Jednou cestou, jak organizovat tutoring systémy, je organizace podle zamýšlené funkce systému. Některé systémy jsou zamýšleny jako replikace učebnice nebo emulace výkladu dodaného k dané probírané doméně. Další systémy jsou navrženy přímo pro podporu procvičování (někdy popisovány jako homework helpers) [5]. Ačkoliv to neplatí pro všechny systémy, výzkum ITS tendenčně směřuje

[1] Tutor – učitel, školitel, vedoucí výuky, nebo automatický systém který vede výuku.

k větší podpoře cvičení a procvičování. Samozřejmě cvičení je jen část výukového procesu a slouží především k odhalení mezer ve znalostech studenta a zautomatizování těchto znalostí. Moderní teorie, které se vyjadřují k učení jako takovému, zdůrazňují důležitost procvičování a správné odezvy ze strany tutora.

Školící systémy typicky podporují jeden nebo dva způsoby procvičování znalostí. Výsledkové tutory ohodnocují finální výsledek jako jsou eseje a podobně. Typicky tedy student pracuje na řešení nějakého projektu či problému, dokud si nemyslí, že je hotový, a pak předloží své závěry pro zhodnocení. Systém ITS pak výsledek analyzuje a snaží se nalézt chyby, nedostatky a jiné závady. Některé pokročilejší ITS jsou schopné použít k vyhodnocení testu reverzní inženýrské postupy, například může použít plán rozpoznávání. Základní slabostí výsledkových tutorů je, že student by mohl uváznout už před tím, než je schopen vytvořit řešení. Více interaktivní a pro-aktivní systémy, které poskytují podporu, zatím co student pracuje na řešení zadaného problému, jsou často popisovány jako procesní tutory. Je to důvěrně známá kategorie, protože i mnoho lidských učitelů uplatňuje při zkoušení studenta tento postup. Student je veden krok za krokem, tutor mu dává nápovědy a pokládá otázky, na něž student odpovídá.

3.2.6 Oblasti výzkumu ITS

Oblasti výzkumu ITS zůstávají neměnné už několik posledních let. Výzkumy za posledních deset let, včetně těch poměrně nedávných, vyzdvihují několik témat, ke kterým se periodicky vrací. Jsou to především:

- Modelování studenta: Hlavní problémy v této oblasti zahrnují rozpoznávání mylných představ, sledování studenta po celou dobu učení, reprezentace chybného rozhodování, otevřené modelování učení, modelování citové a emoční stránky. Modely studentů jsou důležité pro jejich hodnocení. Také mohou být použity pro generování vhodných problémů využitelných při studiu. Jsou základem pro individuální adaptivní výuku.
- Dialog v přirozeném jazyce: Některým z prvním ITS se nedařilo použít přirozený jazyk pro simulaci komunikace „člověk – člověk“. Tento proud výzkumu je stále aktivní. V současnosti je pro produkci moderních výukových systémů s vylepšeným systémem dialogu využívána pomoc od ONR a NSF.
- Modelování rozpoznávání: Výzkum v této oblasti zahrnuje především vytváření možných symbolických reprezentací pravidel a také se zabývá potřebou strategického myšlení pro řešení problémů v dané doméně. Pro hodnocení akcí studenta, generování vhodné zpětné vazby a poskytnutí základu pro modelování studenta jsou používány vyhodnocovací modely. Vědci se snaží postavit rozpoznávací modely i pro expertní školení jako takové.
- Vyvíjení systémů a jejich hodnocení: Výzkum inteligentních výukových systémů se silně překrývá s výzkumy zabývajícími se učením. Kvůli tomuto výzkumu bylo postaveno tisíce

systémů a mnoho stovek z nich bylo hodnoceno. Tento výzkum by měl po určitém čase přispět k vývoji více efektivních školení, čímž přispívá i do výzkumů zabývajících se učením jako takovým.

- Autorské nástroje, sběr znalostí, vývoj nástrojů: Moderní doba klade stále vyšší nároky na rychlost vývoje ITS. Proto v této oblasti stále probíhá intenzivní výzkum a vývoj nových efektivnějších nástrojů pro rychlejší a pohodlnější vývoj ITS.

Výše zmíněný seznam zahrnuje mnoho podkategorií a zdaleka není kompletní. Mezi nejmenované oblasti výzkumu patří například strategie učení, systémového designu a spolupráce prostředí pro výuku. Tyto problémy jsou řešeny v pravidelných intervalech stále znovu, protože zatím neexistuje opravdu obecně použitelné dobré řešení. Nalezení ideálního obecného řešení komplikuje mnoho faktorů, mezi jinými je to široká oblast znalostí, ze kterých může být školení kompletováno, různé cíle školení a různý obsah školení a také rozdíly mezi školenými lidmi. Osvědčený přístup k tomuto problému spočívá ve stavbě specializované komponenty pro každý vzdělávací problém.

V prvních 20 až 30 letech výzkumu v oblasti ITS bylo vyprodukováno velké množství přístupů na principu umělé inteligence, určených pro výukový software [5]. Tyto školící systémy prokázaly, že jsou schopné zahrnout i přístupy k učení, které jiný vzdělávací software zahrnout nedokázal. Příkladem jsou detailní rozpoznávací modely, které dokáží pomoci studentům postupovat správným směrem při řešení problémů. To vede k tomu, že ITS se vyvíjejí se zvyšujícím se úsilím, a také k tomu, že rostou nároky na schopnosti vývojářů ITS. Také to motivuje k tomu, aby vývojáři věnovali větší pozornost autorským nástrojům pro ITS.

3.2.7 Současné trendy a vývoj v oblasti ITS

Široce používaným trendem je vytváření studentského modelu (learner modeling). Posledních šest let byla věnována velká pozornost citovým a emocionálním stavům studentů. Největší úsilí se věnuje problému motivace studentů. Bylo zjištěno, že lidský školící expert se snaží řídit motivaci a emoční stavy studentů a že motivace má zásadní vliv na efektivitu školení. Při používání počítačového prostředí lze odvozovat citové stavy studenta ze stavu výpočetního systému, například se může měřit čas mezi stisknutím kláves. Vývojáři již dokázali vyvinout algoritmus, který z podobných událostí na výpočetním systému určuje pravděpodobný stav studenta [5].

- *Otevřené modelování studenta (Open learner modeling).* Je to rozšíření tradičního modelování studenta, které dělá model viditelnou a interaktivní částí učebního prostředí. Jinými slovy zobrazuje reprezentaci toho, jak je na tom podle systému student se svými znalostmi. Pro běžnou vizualizaci této reprezentace se běžně používá například průběhový pruh (progress bar). Jak student řeší úkoly a problémy z dané oblasti, každá jeho akce je zaznamenána a pozitivně či negativně ovlivňuje povědomost systému o tom, jak dobře dané oblasti rozumí. Student může sledovat pohyb průběhového pruhu a tím mít přehled o tom, jak

systém hodnotí jeho znalosti. Tato skutečnost má motivační charakter, neboť student má přímou zpětnou vazbu na své akce a může se systémem v jistém smyslu soutěžit o lepší výsledek. Otevřený model studenta má podporovat přemýšlení a aktivní učení, protože studenti musí hodnotit to, jak porozuměli dané oblasti učiva. Postup výukovým systémem je pro ně v určitém smyslu výzvou.

- *Autorské systémy (Authoring systems).* Jak již bylo několikrát předesláno, pokud se mají inteligentní výukové systémy ve větším měřítku rozšířit mimo prostředí vývojových laboratoří, musejí být k dispozici dobré autorské nástroje pro snadný vývoj nových školících systémů, či vylepšit vlastnosti a chování těch stávajících. Existuje několik úspěšných systémů, které se soustřeďují na vytváření a modifikaci obsahu daného školení. V oblasti vytváření tutoriálů a expertních znalostí je situace zatím značně horší. Zvlášť nadějný přístup k problému autorství školení je založen na myšlence „*autorství pomocí příkladu*“. Hlavní idea spočívá vtom, že než tak zapracovávat do systému doménovou expertízu a veškeré znalosti, které se týkají školení v programovacím jazyce umělé inteligence, je lepší, když autor demonstruje ideální řešení. Pro vytvoření zpráv zpětné vazby pak vývojář pouze specifikuje, co by mohl systém vypisovat v různých okamžicích této demonstrace. Pro vypořádání se s omyly autor jednoduše označí části demonstrace jako chyby a poté pro ně vytvoří vhodné systémové zprávy. Tento přístup byl použit v projektu CTAT (Cognitive Tutoring Authoring Tools). Předběžné testování ukázalo zrychlení vývoje školení mezi 1.4x až 2x oproti vývoji s nástroji, které nedisponují schopností demonstrace. Protože demonstrační modely inklinují k tomu být příliš nepřizpůsobitelné, vědci zkoumají možnosti použití technik strojového učení k odvozování rozpoznávacích modelů ze série demonstrací.
- *Skupinové a online učení.* Rozvoj internetu a relativně jednoduchá komunikace mezi počítači prostřednictvím sítí s sebou přinesl nové možnosti v oblasti spolupráce při učení. Historicky byly práce na systémech podporujících skupinové školení limitovány, ale s úspěchem komunity CSCL[1] v posledních několika letech se možnosti výzkumu a práce v této oblasti značně zvýšily. První CSCL systémy využívaly sítí propojeného prostředí, které vykonávalo zrcadlení všech akcí v dané komunitě. Pokročilé CSCL systémy poskytovaly vyšší úroveň podpory než jen zrcadlení funkcí. Například poskytovaly informace o tom, jak rychle spolupracovníci řeší dané úkoly v rámci výuky. Také ukazovaly míru komunikace mezi spoluúčastníky a monitorovaly to, jak jednotliví členové týmu řeší dané úkoly, aby jim případně nabídli individuální pomoc. Protože robustní porozumění jazyku je zatím nevyřešený problém, rady dávající CSCL systémy inklinují k jiným technikám, kterými lze monitorovat komunikační aktivity. Jeden přístup předpokládá hledání začátků vět[2], které umožňují odvodit směr komunikace a sledovat vzorky spolupráce spoluúčastníků. Hodnocení

[1] Computer Supported Collaborative learning
[2] Například: I agree, but…

CSCL systémů často ukazuje, že nebylo realizováno mnoho očekávaných výhod těchto systémů, jako je například zvýšení motivace a spoluúčasti školených lidí. Různé opakovaně prováděné studie ukazují problémy, jakými jsou například nízká spoluúčast a problémové sledování komunikace, nízké uspokojení z učení a omezené možnosti výuky. Vývojáři ITS se snaží vypořádat s některými těmito problémy a zkoušejí různé přístupy, jak zlepšit spolupráci při týmovém školení. Mezi tyto přístupy patří například vyšší úroveň týmové vizualizace, podpora pro vzájemné učení a inteligentní přiřazování členů týmu ke společné výuce. Problémem je omezená možnost automatického posouzení výkonu jednotlivých členů týmu, protože vytvořit rozhodovací plán pro skupinu spolupracovníků je velmi složité. Je možné řešit tuto situaci školením jednotlivců stylem „jeden na jednoho" v rámci týmového prostředí.

- *Dialog v přirozeném jazyce.* Lidé využívají v komunikaci různé techniky, jako jsou řeč těla, gesta, intonaci a dialog. Protože se mnoho inteligentních výukových systémů vyhýbá použití přirozeného jazyka[1], někteří výzkumníci poukazují na fakt, že vhodné použití přirozeného jazyka navozuje větší porozumění mezi počítačovým školitelem a školeným člověkem. Je to také motivace pro výzkumníky pedagogických agentů, kteří využívají i jiné způsoby komunikace než dialog, jako jsou například výrazy obličeje, gestikulace, apod. Hlavní síla dialogu pro účel výuky spočívá ve větší možnosti interakce ve školících systémech. Pokročilé techniky porozumění přirozenému jazyku a autorské nástroje umožňují porozumět odezvě od studenta a udržovat dobrou produktivitu a vhodnost reakcí systému.
- *Přínos ITS pro vývoj vědy zabývající se výukou.* Často zmiňovanou výhodou automatizovaných školení je, že z jejich chování mohou být snadno zpracovány specifické hypotézy, které se zabývají efektivitou výuky. Při školení vedeného lidským školitelem je to problém. Dobře implementovaný ITS má obvykle velmi široké možnosti nastavení odezvy studentovi, ať už jde o četnost, formu nebo obsah zpráv a testů. Ve výzkumném centru zabývajícím se výukou PSLC[2] v Pittsburgu se zkoumá využití inteligentních výukových systémů, stejně jako klasických metod výuky, na širokém záběru učebních případů, a tak se tam vytváří silná teorie zabývající se učením jako takovým.

3.2.8 Další oblasti výzkumu ITS

Velká část výzkumu v oblasti inteligentních výukových nástrojů a umělé inteligence byla provedena na základě potřeb a požadavků americké armády [5]. Byly vyhotoveny rozsáhlé analýzy stávajících školení americké armády a konzultace s experty na ITS. Doporučení, která z těchto analýz a konzultací vznikla, mají široké uplatnění a obsahují podněty pro další výzkum. Mezi cíle výzkumu patří například vytvoření ontologie ITS k organizaci konceptu ITS a usnadnění jejich vývoje, dále

[1] Tyto systémy jsou často nazývány jako: Systémy druhé generace.
[2] Science of Learning Center.

mapování tříd architektury ITS a aplikačních domén, zvýšení výzkumu pedagogických agentů, virtuální emulace člověka, samotná podpora výzkumu a vývoje prototypů pro týmová školení a další vývoj a výzkum autorských nástrojů a samotných ITS systémů.

Nejvýraznější oblasti výcviku americké armády, které jsou vhodné pro přechod na systémy ITS, jsou historie válečnictví a analýza bitev. Tyto domény mají velkou důležitost a obsahují dobře definované komponenty, čímž jsou vhodné k implementaci pomocí ITS. Úspěšná integrace těchto ITS by sloužila jako příklad a motivace pro širší zavedení těchto systémů i do dalších oblastí výcviku.

3.2.9 Školení a hodnocení ve špatně definovatelných doménách

V dobře definovatelných doménách vědy jako je algebra, fyzika, či počítačové programování, byl již ve výzkumu a vývoji ITS učiněn významný pokrok. V těchto oblastech lze jasně definovat rozdíl mezi špatnou a správnou odpovědí. Při školení a ověřování znalostí se snadno rozpoznají správné a špatné akce. Také generování vhodné zpětné vazby ve formě odpovědí systému je poměrně jednoduché. Velký počet oblastí, které jsou různě závažné pro výcvik americké armády, nelze zpracovat na základě takovýchto jednoduchých expertních modelů. Tyto oblasti jsou často popisovány jako špatně definovatelné domény. Těmto oblastem se obecně věnuje méně pozornosti než těm dobře definovatelným. Pokud se zpracovává doména, která je špatně definovatelná a její zvládnutí tedy zahrnuje případy, které nelze vyhodnotit jako dobré nebo špatné, může nastat jeden ze dvou případů. Buď daná doména jen vypadá jako špatně definovatelná a ve skutečnosti je dobře definovatelná, pak musíme danou doménu vhodně analyzovat, nebo je doména opravdu špatně definovatelná. Pak je třeba do zkoušení znalostí dané domény zahrnout názory expertů a element subjektivnosti.

I pro oblasti, které vypadají jako špatně definovatelné, byly úspěšně vyvinuty prototypy inteligentních školících systémů [5]. Jsou to například: správné uvažovaní, interpretace umění, kulturní povědomí, nebo design databází. Pro většinu špatně definovatelných domén však zatím žádné systémy vyvinuty nebyly. Existují technologie, jejichž hlavní síla se projevuje při zvládání nejasně definovaných domén, je to například „Bayesian modeling", což je přístup který je agnostický k tomu, co znamenají jednotlivé uzly modelu, a silně se zajímá o to, jak probíhají jeho aktualizace. U dalších přístupů to není tolik patrné, například „tracing algorithms" se často spoléhá na schopnost vyhodnotit akce jako správné a nesprávné. Obecně lze říct, že jak postupuje výzkum špatně definovatelných systémů, tak jsou unikátní požadavky na schopnosti takového systému stále jasnější.

Zvýšená pozornost se věnuje především dvěma základním oblastem výzkumu. První z nich je vypracování detailního popisu chování experta, který provádí analýzu špatně definovatelné domény a který řeší i další otázky, týkající se získávání a sběru znalostí dané domény. Druhá oblast se zabývá tím, jak lidští experti provádějí vyhodnocování ve špatně definovatelné oblasti. Je třeba detailní popis procesu, který experti používají pro rozhodování, klasifikaci a zpětnou vazbu, kterou dávají svým

studentům ve vysvětlování špatně definovatelné domény. Pro sběr vyhodnocovacích dat, na základě kterých se dělají rozhodnutí, jsou ideální hry.

3.2.10 Seriózní hry a vyprávěcí výukové prostředí

Experimentální výuka a výuka založená na vyprávění může mít zásadní úlohu pro vytváření zkušeností u studentů. Interaktivní prostředí, která jsou založena na vyprávění příběhu, mohou mít významnou úlohu v další generaci nástrojů určených k budování vůdcovských schopností. Pro vypravování a procvičování schopností je vhodné využít prostředí moderních her [5]. Poměrně nová oblast výzkumu se tedy zabývá vývojem tzv. „seriózních her“. Je zde snaha skloubit realistickou simulaci reálného světa s motivačními vlastnostmi hry.

Seriózní hry jsou od začátku vyvíjeny se vzdělávacím či školícím záměrem. Zatím nejsou pro učení pomocí seriózních her stanovena přesná hodnotící pravidla. V tomto ohledu je třeba provést další výzkum, protože neschopnost zhodnotit dosažené znalosti je velkou slabinou. Tato oblast výzkumu je zatím poměrně mladá. Prozatím existuje jen několik prototypů zajímavých školících her.

Poměrně málo pozornosti od komunity vývojářů ITS je věnováno inteligentnímu ovlivňování sebe sama za účelem dosažení nějakých pedagogických cílů. Například změna obtížnosti v komerčních počítačových hrách vede k posílení zábavného prožitku ze hry. Zajímavý je výzkum, který je prezentován prozkoumáváním prostoru obtížnosti a řízení chování hry. Přitom se sleduje, jak tyto aspekty mohou motivovat k učení.

3.2.11 Poznávací učení

Doménoví experti inklinují k vytvoření rozpoznávacích schopností, které se tvoří a akumulují v průběhu času z jejich zkušeností. Mezi tyto schopnosti patří plánování, přemýšlení a uvažování o hypotetických situacích. Ačkoli určitý počet současných školících systémů je zaměřen na rozšíření rozpoznávacích schopností, ve špatně definovatelných doménách je třeba udělat ještě velké množství práce. Tyto schopnosti jsou zásadní pro proces rozhodování a kritického myšlení [5]. Také další oblasti výzkumu umělé inteligence jsou pro výzkum inteligentních výukových systémů důležité. Jsou to především výzkumy přirozeného jazyka, dialogové systémy a výzkum rozhodování na základě běžných smyslů. Výzkum v oblasti špatně definovatelných domén zahrnuje především zjišťování, jak v dané doméně pracuje lidský expert. To je asi nejdůležitější pro pochopení procesu myšlení.

3.2.12 Automatická detekce neproduktivního chování

Jelikož studenti často nemají dostatečně rozvinuté rozpoznávací schopnosti a potřebné základní znalosti, je dobré jim názorně ukázat, jaké chování je neproduktivní. Určitý čas je třeba studenty vést, aby si rozvinuli smysl pro produktivní učení. Vzhledem k tomu, že každé školení je třeba provést v co nejkratším čase, je vysoce důležité co nejvíce omezit neproduktivně strávený čas. Neproduktivní

chování lze rozdělit do dvou základních kategorií. První kategorií je tzv. „hráčské chování“. Je to situace, kdy student zneužívá výukové prostředí k tomu, aby rychle dosáhl daných cílů. Tento případ nastává, když student nadmíru využívá nápovědu k řešenému problému. Druhá kategorie zahrnuje situace, kdy student při učení uvázne a neví jak dál. Takový student často začne dělat různé pokusy, aby zjistil, zda se nestane něco, co by mu pomohlo. To obvykle zahrnuje rozbalování menu, klikání na tlačítka a jiné podobné aktivity. Při úspěchu student splní požadované podmínky systému bez toho, že by skutečně nabyl potřebných znalostí.

3.2.13 Závěrečný pohled na inteligentní systémy

Výsledkem výzkumu systémů ITS jsou složité algoritmy a techniky, které jsou aplikovány na problémy spojené se vzděláním. Specifická výhoda vzdělávacího softwaru založeného na umělé inteligenci spočívá především ve schopnosti reprezentovat danou doménu znalostí a možnosti aplikace takových způsobů učení, které v jiném prostředí nejsou možné. Různé studie ukazují vysokou důležitost správného vedení při výuce, přičemž přítomnost lidského školitele není vždy možná (například doma). ITS systémy se snaží tyto mezery zaplnit. Pro určité oblasti lidského vzdělávání jsou ITS velmi vhodné, stejně jako simulace a inteligentní seriózní hry.

Inteligentní výukové systémy se v posledních letech vyvíjejí velmi vysokým tempem. Především se to týká dobře definovatelných domén. Naproti tomu automatizované vzdělávání u špatně definovatelných domén stále naráží na celou řadu problémů. Problémovými doménami jsou například schopnosti vůdcovství či mezilidské vztahy. Oblast špatně definovatelných domén lze obecně brát jako velkou výzvu pro vědce zabývající se reprezentací znalostí, modelováním studenta, pořizováním expertíz a vytvářením autorských nástrojů. Mnoho vědců věří, že velkou budoucnost pro vzdělávání mají seriózní hry, které motivují k učení a dělají ho zábavným. Otázka skutečného využití seriózních her však zatím nebyla zcela prozkoumána. Také stále pokračující výzkum dialogových systémů má pro systémy ITS potenciál.

3.3 Stávající projekty ITS

3.3.1 SYSPROS a POINTRA

Inteligentní výukový systém SYSPROS se zabývá oblastí synchronizace paralelních procesů pomocí semaforů. Zkratka SYSPROS znamená „**SY**nchronization of parallel **PRO**cesses with **S**emaphores“. Systém se snaží co nejlépe emulovat schopnosti lidského učitele.

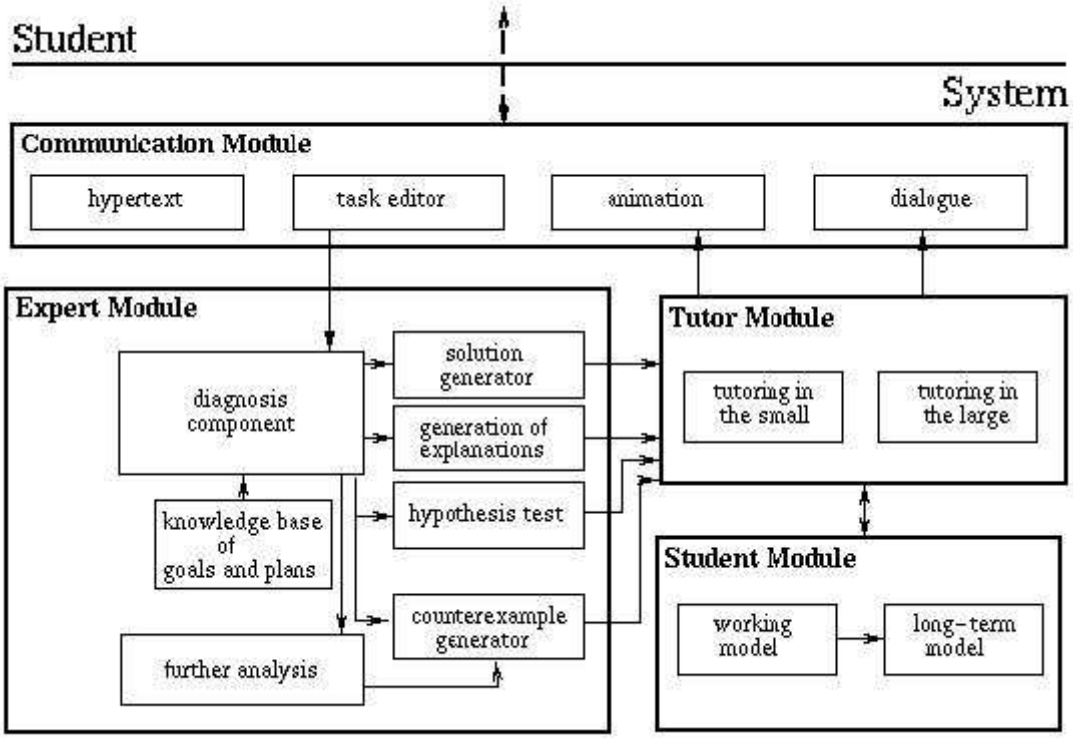
Stejně jako u jiných systémů se architektura systému SYSPROS skládá ze čtyř hlavních komponent (viz Obr. 1). Expertní modul projektu obsahuje bázi znalostí příslušné domény a snaží se generovat stromy řešení daných procesů, které reprezentují jednak správná řešení a jednak běžné

omyly, které při řešení paralelních procesů vznikají. Studentovo řešení daného problému může být pomocí těchto stromů diagnostikováno a systém na jejich základě poskytuje vhodnou odezvu, reprezentovanou ukázkou správného řešení, vysvětlením problému, vysvětlením chyb, apod. Systém disponuje různými možnosti testování a generování příkladů, na kterých je dobře ukázána chybnost daného řešení.

Didaktické schopnosti systému jsou obsaženy v modulu *tutor*. Tento modul obsahuje základní strategie pro probírání obsahu jednotlivých kapitol školení. Přístup ke studentovi, tedy především rozvržení času a míra interakce, je řízen na základě znalostí studenta a jeho motivace.

Modul *student* projektu uchovává informace o znalostech studenta a jeho typických metodách práce. Tyto informace jsou základem pro rozhodování o průběhu výuky. Metody studenta systém sleduje pomocí pracovních diagramů. Sleduje jak dlouhodobý vývoj, tak aktuálně řešený problém.

Poslední částí systému je komunikační modul. Ten srozumitelnou cestou reprezentuje znalosti domény v systému. Systém využívá grafické reprezentace, animací a ilustrací. Je možné editovat stávající grafy procesů a provádět simulace běhu těchto posloupností procesů. U textů je využíván hypertext.



Obr.1. Schéma systému SYSPROS,
Staženo z: http://wwwbruegge.in.tum.de/projects/intusys/intusys_eng.html

Souběžně se systémem SYSPROS se vyvíjí i systém POINTRA, který je vyvíjen na stejné bázi a liší se oblastí domény. ITS POINTRA se zabývá výukovou programování v programovacím jazyce Pascal s ukazateli [6].

3.3.2 CIRCSIM-Tutor Project

Tento inteligentní výukový systém je zaměřen na oblast vyučování některých znalostí z oblastí medicíny. Jedná se o principy kontroly reflexů a krevního tlaku a je určen studentům prvního ročníku medicíny. Studenti řeší drobné problémy a vedou dialog v duchu Sokrata s počítačem. Výzkum se zaměřuje především na dva aspekty. Prvním aspektem je charakteristika jazyka a chování při lidském vzdělávání a druhým je snaha o co nejúplnější adaptaci této charakteristiky do prostředí ITS.

Aktuálně se pracuje už na třetí verzi programu. První verze systému byla odvozena od expertního systému MACMAN, který je postaven na matematickém modelu reflexního baroreceptoru. Ten byl napsán ve FORTRANU a běží na mainframe serveru univerzity McMaster, kde byl vyvinut. V podstatě se jednalo o přepsání kódu MACMANu do jazyku TUTUO pro platformu PLATO. Pro vzdělávání studentů však systém neměl vhodné vlastnosti.

Na matematickém modelu MACMANu se vyvinul HEARTSIM. Ten byl napsán přímo pro platformu PLATO a skládal se ze dvou hlavních komponent, z nichž jedním byl matematický model MACMANu a druhým učitelský modul. Tato didaktická komponenta je založena na principu sady experimentů, které studenti vykonávají. Protokol má 6 kroků. Prvním je výběr experimentu (procedury), druhým je vytvoření předpovědí pro daný experiment, třetím je oprava logických chyb, čtvrtým oprava chyb ve vztazích a pátým výstup výsledků ve formě tabulky. Šestý krok je porovnání předpovědí s aktuálními výsledky a poskytnutí odpovídající zpětné vazby studentovi.

Ze systému HEARTSIM se později konverzí z platformy PLATO do platformy DOS vyvinul projekt CIRCSIM [7]. Hlavním účelem této konverze bylo rozšíření programu více studentům. CIRCISM se také lišil tím, že hlavní komponentou již nebyl matematický model, ale didaktické vlastnosti systému. Projekt tedy neobsahuje samotný matematický model, ale jen data nutná k zobrazování a porovnávání výsledků. Tato data pochází ze systému HEARTSIM. Velmi byly rozšířeny didaktické vlastnosti, především možností generování zpětné vazby studentovi.

3.3.3 ANDES

Projekt ANDES se zabývá oblastí výuky fyziky. Tento systém není typickým ITS [8], jedná se spíše o inteligentního pomocníka s domácími úkoly z fyziky. Důraz je kladen na uživatelské rozhraní, kde by měl mít student stejnou volnost projevu jako při použití papíru a tužky. Systém poskytuje kontrolu správnosti vstupů a poskytuje nápovědy. Podle průzkumů se studenti, kteří používali pro úkoly s fyziky ANDES, naučili mnohem více než studenti, kteří tvořili domácí úkoly na papíře. Systém se snaží vést studenta při řešení úkolu správným směrem, a to tak, aby se co nejvíce snížila frustrace studenta a časové nároky na řešení úkolu. Systém byl vyvinut Univerzitou v Pittsburgu a Akademií Amerického námořnictva. Je volně dostupný a existuje i online verze.

3.3.4 ITAAM

Zkratka ITAAM znamená „Intelligent Tutoring Agents in the Area of Medicine“. Tento systém je vyvíjen centrem počítačových vět a technologie[1] na Univerzitě v Minho. Koncepcí systému je vytvořit multi-agentní výukový systém, ve kterém bude výukový agent dohlížet na studium studenta a poskytovat mu pomoc. Doménou znalostí je medicína [9]. Agent se snaží porozumět studentům a jejich znalostem. Na základě porozumění studentům uplatňuje vhodnou školící strategii a spolupracuje se studenty na dosažení stanovených cílů. Při implementaci systému se objevilo mnoho problémů, z nichž některé nebyly úplně dořešeny. Souvisely především se sbíráním znalostí o studentovi. Protože nebylo možné systém implementovat pomocí monolitické architektury, byl implementován jako multi systém s komunitou adaptivních agentů, kteří spolu vzájemně komunikují a spolupracují na daných cílech.

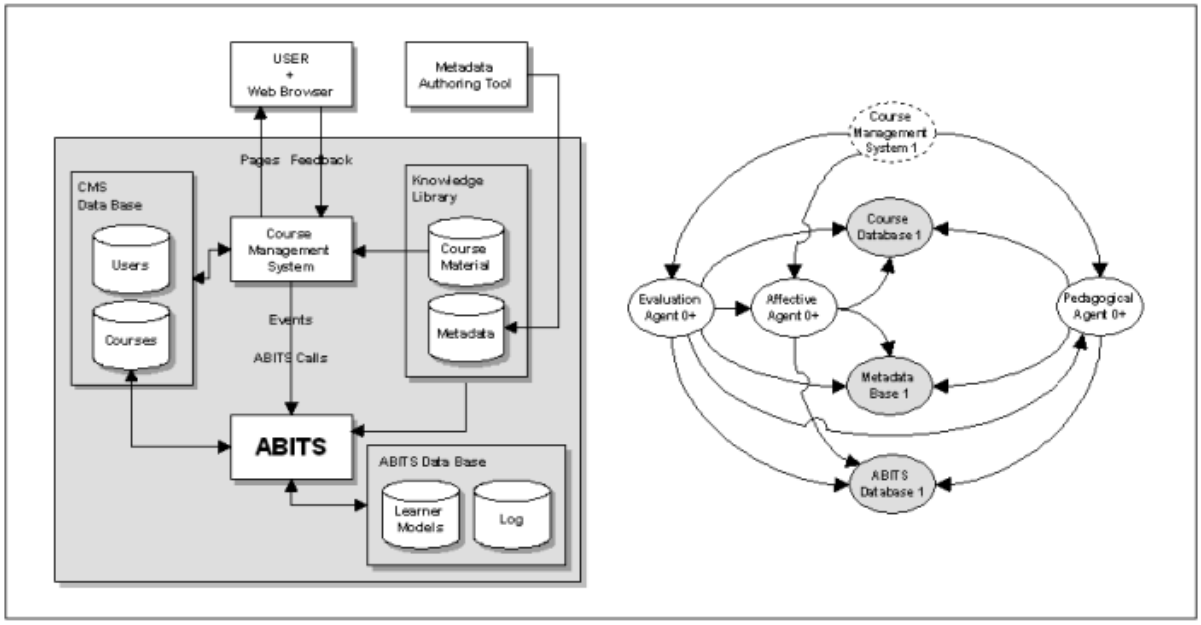
3.3.5 ABITS

Systém ABITS je inteligentní výukový systém pro distanční vzdělávání založený na bázi uměle inteligentních agentů [10]. Původně vzniknul jako modul většího ITS INTRASYS. Jedná se o webový systém se všemi výhodami, které z této skutečnosti vycházejí. U webových výukových systémů se využívají tři základní přístupy. V prvním případě se jedná se o statické systémy, kdy pedagog vytvoří obsahové části lekcí pomocí značkovacích jazyků a umístí je na web. Druhý je personalizovaný přístup, kdy učitel využívá nějakého softwaru při správě daného školení a studentů a je schopen prostřednictvím tohoto softwaru sledovat, ovlivňovat a testovat postup studentů předmětem i školení samotné. Třetím typem je adaptivní přístup, který rozšiřuje personalizovaný přístup o možnosti, které učiteli nabízí umělá inteligence systému. ABITS je systémem tohoto adaptivního typu.

Veškerý didaktický materiál je v systému obsažen v několika výukových objektech a uložen v souborovém systému učebních materiálů. Výukový objekt lze chápat jako kontejner učebních dat, kterým může být například probíraná látka lekce, simulace, test, apod. Tyto výukové objekty jsou indexovány a obsahují základní metadata, pomocí kterých lze určit jak s daným objektem nakládat při výuce.

Systém má propracované modelování studenta. Sleduje stav poznání probíraných lekcí a vědomostí studenta a také studijní předpoklady na základě jeho aktivit. Při tomto modelování je využito fuzzy výpočtů a fuzzy množin.

[1] Computer Science and Technology Center (Centro de Ciências e Tecnologias de Computação) (CCTC)



Obr.2. Architektura systému ABITS

Staženo z: http://www.capuano.biz/Papers/ITS%202000/ITS%202000.htm

Architektura systému je modulární (viz. Obr.2). Modul ABITS používá pro komunikaci s uživatelem modul course management systém, který má vlastní databázi. Oba systémy jsou napojeny na databázi znalostí a samotný modul ABITS má vlastní databázi. Systém je multi-agentní, což zaručuje širokou škálovatelnost. V systému existují tři základní typy agentů. Prvním typem jsou hodnotící agenti, kteří ohodnocují stav znalostí studenta a celý studentský model. Aby tuto činnost mohli vykonávat, přicházejí do styku s databází metadat, s ABITS databází a s ostatními dvěma typy agentů. Druhým typem jsou afektivní agenti, kteří monitorují citové stavy studentů. Podle výsledků upravují studijní předpoklady studentů. Interaktují s databází metadat a ABITS databází. Třetím typem jsou pedagogičtí agenti. Ti se starají o stav výuky. Interaktují s databází interakce, ABITS databází a databází kurzu.

3.3.6 CALO

Systém CALO je rozpoznávací agent, který má za úkol učit a organizovat (Cognitive Agent that Learn and Organize). Tento projekt vyvinula organizace DARPA (Defence Advanced Research Project Agency) [11]. Cílem projektu je výukový systém pro velitele ozbrojených složek, kteří se z něj učí například taktiku a zvládnutí bojových situací, pomoc při rozhodování u malých týmů, hledání nových cest ve vývoji software, podporu vědy zabývající se rozhodovacími procesy, učením, multi-modální interakcí, všímavostí, apod.

Systém CALO lze chápat jako asistenta s umělou inteligencí, který má pomoci člověku s vedením svého obchodního života. Výuková funkce systému se využívá spíše v armádě, kde se snaží vychovat důstojníky k tomu, aby správným způsobem vedli armádní záležitosti.

3.3.7 ICICLE

Projekt ICICLE je ITS, který se zabývá identifikací a opravou jazykových chyb. Zkratka znamená Interactive Computer Identification and Correction of Language Errors. Projekt je zaměřen na anglický jazyk [12]. Hlavním cílem projektu je využít přirozeného jazyka a generovat školení pro studenty spisovné angličtiny. Důraz je kladen na individuální vhodnost a správnost zpětné vazby pro každého studenta.

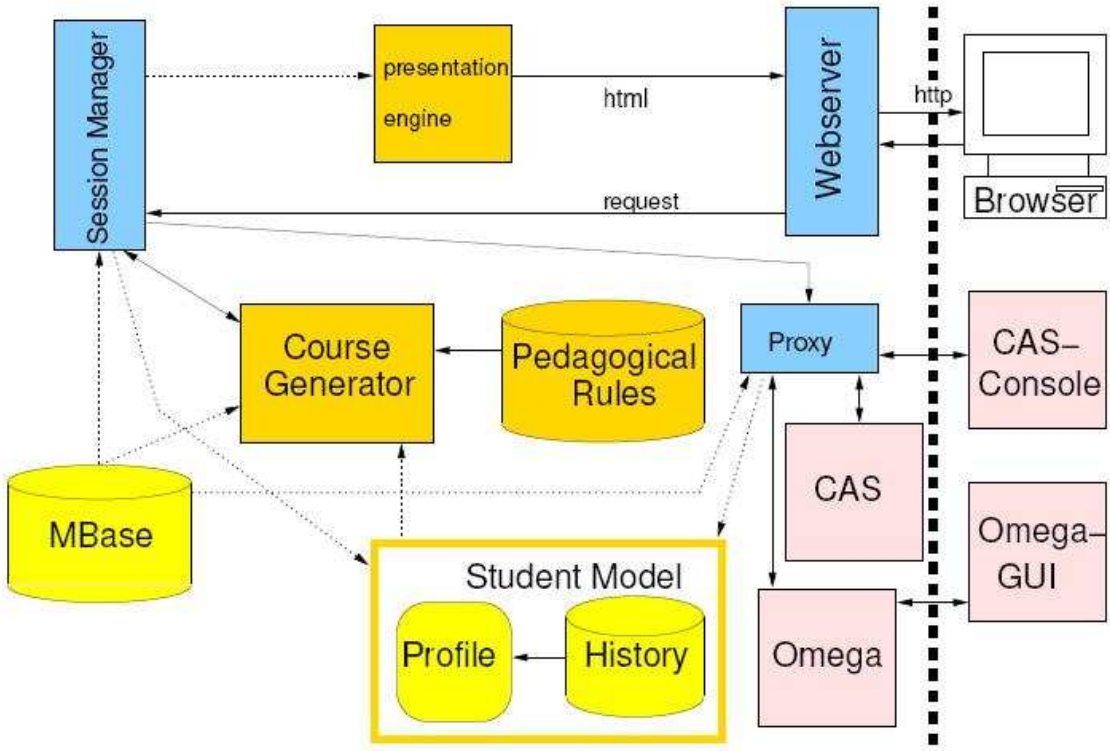
Interakce systému se studenty má podobu cyklu uživatelských vstupů a odpovědí systému. Tento cyklus začíná, když uživatel odešle ke kontrole kus napsaného textu. Systém na textu provede syntaktickou analýzu, určí případné chyby a vypíše odpovídající zprávu. Tyto zprávy obsahují vysvětlení gramatických jevů a rady, jak se podobných chyb vyvarovat. Tento systém má sloužit především anglicky mluvícím studentům, kteří potřebují vylepšit psanou spisovnou angličtinu.

3.3.8 ActiveMath

Tento webový inteligentní výukový systém je určen pro výuku matematiky. Systém je navržen jako velmi interaktivní a umožňuje volnou navigaci mezi kurzy [13]. Model studenta je modifikovatelný. Systém se pomocí adaptivního výběru obsahu kurzu snaží přizpůsobit výuku včetně obtížnosti příkladů znalostem a schopnostem studenta, čímž roste motivace a chuť do učení. Systém je vhodný pro dálkové učení prostřednictvím internetu a poskytuje adaptivní vzhled, adaptivní možnosti prezentace i adaptivní změny obsahu.

Architektura systému striktně odděluje reprezentované znalosti od funkcionality systému, stejně jako separuje různé druhy znalostí (viz. Obr.3.). Pedagogické znalosti jsou uloženy v bázi pedagogických pravidel, znalosti o studentovi ve studentském modelu a vzdělávací obsah v databázi MBase. Pro samotnou implementaci je použit model klient-server, přičemž typ klienta je omezen na webový prohlížeč.

Obsah každého kurzu je uživateli generován individuálně na základě jím vybraných učebních cílů. Generátor kurzů se spojí s databází znalostí a vytvoří graf s ID požadovaných komponent obsahu kurzu. To je převedeno do jazyka XML pomocí prezentačního stroje a takto se prezentuje v prohlížeči uživatele. Systém obsahuje i vnořené externí systémy jako jsou systémy počítačové algebry Maple a MuPad či plánovač důkazů Multi. Generovaný kurz se adaptivně přizpůsobuje prostředí studenta. Tedy jeho technickému vybavení, proměnným prostředí, životopisu, jazyku, oboru a zejména vzdělávacím potřebám. ActiveMath vede důkladný model studenta obsahující statický a dynamický profil každého studenta.



Obr. 3. Architektura systému ActiveMath

Vyjmuto z: http://www.activemath.org/pubs/Melisetal-Activemath-ICAISC-2004.pdf

ActiveMath byl vyvinut skupinou „The Intelligent Tutoring System Group“ na Univerzitě v Saarlandu.

4 Projekt automatického vyhodnocování e-learningových testů.

4.1 Popis projektu

4.1.1 Cíle projektu

Cílem projektu je vytvořit webovou aplikaci na architektuře klient server, která bude schopna vyhodnocovat testy obsahující víceslovné odpovědi. Tato technika je využitelná v prostředí elearningových systémů a inteligentních výukových systémů. Běžné e-learningové systémy využívají k testování znalostí studentů testy, ve kterých student pracuje s danými odpověďmi. Obvykle buď vybírá X odpovědí z Y možných, nebo odpovědi přiřazuje do dvojic k položeným souvislostem.

Rozšíření těchto možností o automatické vyhodnocování e-learningových testů má značné didaktické výhody, především zvyšuje vypovídací hodnotu testu o znalostech studenta.

4.1.2 Aplikační prostředí

4.1.2.1 Technologie pro vývoj aplikace

Aplikace je navržena na architektuře klient server. Tato architektura byla zvolena jako nejlépe odpovídající současnému trendu vývoje e-learningových systémů a inteligentních výukových systémů. Pro implementaci aplikace lze vybrat několik různých platforem a nástrojů. Hlavními součástmi projektu jsou aplikační server, databázový server a samotná aplikace.

Jako technologie pro samotnou aplikaci byla zvolena Java SE. Bylo zvažováno nasazení PHP nebo ASP.NET technologií, ale Java vyšla jako nejlepší při subjektivním hodnocení autora projektu. Java je velmi silný a bezpečný jazyk a pro webové systémy má velmi dobré vlastnosti a výkon. Velkou výhodou je multi-platformnost Javy a velmi vyspělé API pro vývoj webových aplikací. Velkou roli hrály také licenční podmínky a finanční dostupnost navazujících technologií jako je aplikační a databázový server. Pro vývoj webových aplikací se vhodnější použít Javu EE, ale rozšíření, která tato distribuce Javy nabízí navíc oproti standardní verzi, nebyla k vývoji aplikace nezbytná.

Jako aplikační server je pro aplikaci napsanou v jazyce Java možné využít různých možností. Společnost SUN Microsystems, která jazyk Java vyvinula, nabízí vlastní komerční řešení Java SUN Application Server. Výhodou tohoto řešení je vysoká stabilita, výkon a uživatelská podpora. Pro tento projekt bylo vybíráno vhodné nekomerční řešení, které bude dostatečně stabilní, výkonné a rozšířené. Architektura Javy je postavená na myšlence virtuálního stroje, který spravuje běžící aplikace a přitom běží na libovolné platformě. Pro běh webové aplikace je třeba, aby server podporoval Java Container pro zpracování Javy. Vhodné licenční podmínky mají open source projekty TOMCAT a Jetty. Tyto Java Containery lze provozovat jako pluginy pod open source serverem Apache. Server Apache patří k nejpoužívanějším a nejvýkonnějším serverům. Nejsilnější zvažovanou alternativou byl open source aplikační server GlassFish. Z nabízených containerů byl zvolen TOMCAT, protože je známější a rozšířenější. Rozšířenost má zásadní vliv na možnost a rychlost řešení možných problémů, které v průběhu práce s containerem nastávají.

Jako hlavní kritérium při výběru databázového serveru byly zvoleny vhodné licenční podmínky. Nejznámější a nejrozšířenější je databázový systém MySQL. Mezi dalšími zvažovanými systémy byly PostgreeSQL a Firebird. Při výběru databázového systému je třeba věnovat zvýšenou pozornost tomu, jakým způsobem a jestli vůbec k tomuto systému přistupuje technologie použitá pro vývoj samotné aplikace. Java využívá pro přístup k databázím standardizované rozhraní JDBC a pro každý databázový systém by měl existovat ovladač databáze, jehož prostřednictvím bude toto

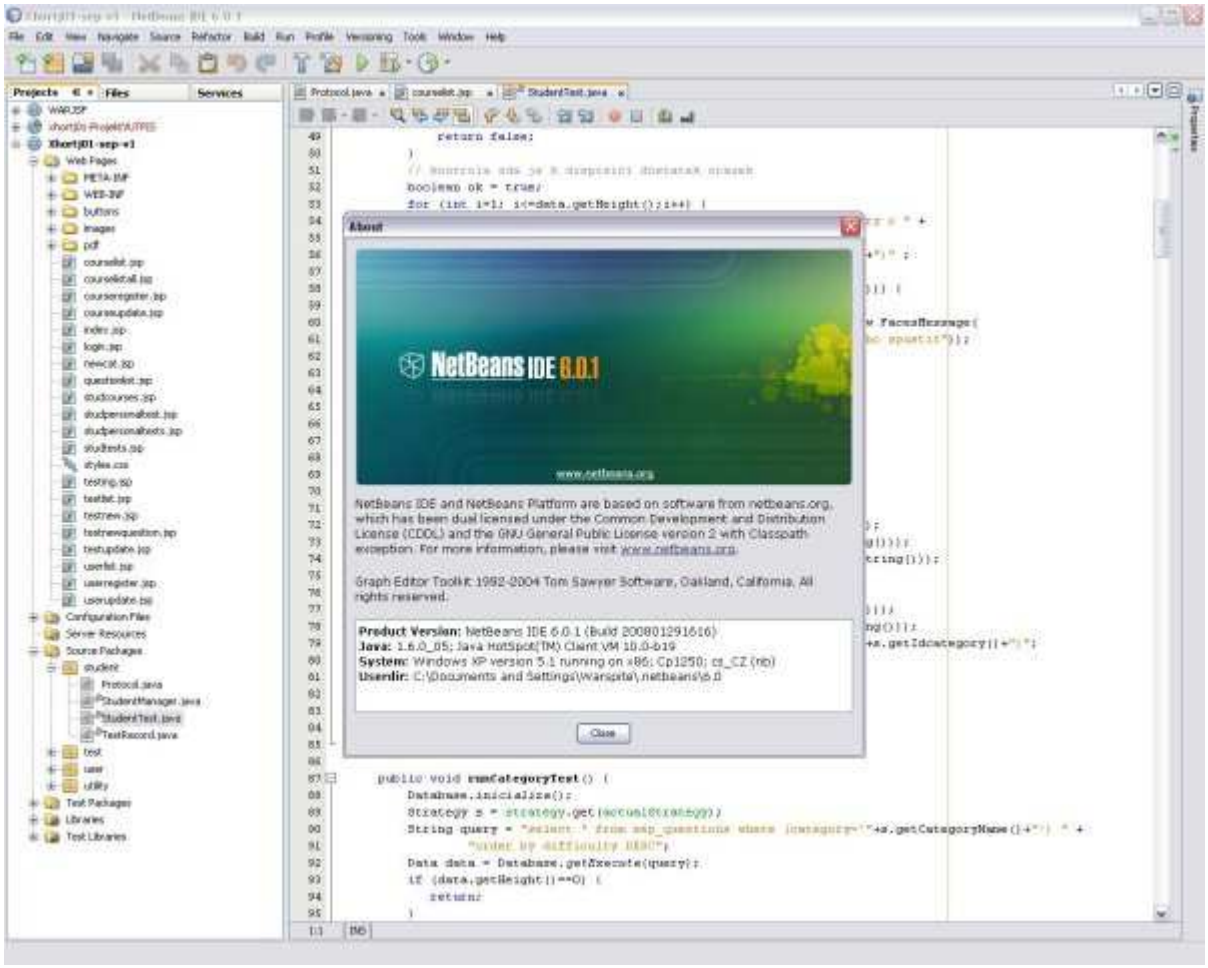
rozhraní s databází komunikovat. Pro MySQL existuje ovladač přímo od výrobce a je volně ke stažení ze stránek výrobce.

4.1.2.2 Podpůrné nástroje pro vývoj aplikace

Pro návrh i implementaci aplikace je vhodné využít nástrojů, které vývoj usnadní a urychlí. Pro samotný návrh se v současné době používá modelovací jazyk UML. Pro modelování v tomto jazyku existuje určité množství komerčních nástrojů, obvykle podporujících určitou metodiku vývoje softwaru. Známý je systém RATIONAL ROSE podporující Unified Process, či Enterprise Architect, který podporuje metodiku Select Perspective. Vzhledem k požadavkům na cenu softwaru se vybíralo z nabízených nekomerčních projektů a různých volných verzí komerčních systémů.

Z nekomerčních nástrojů pro UML je zřejmě nejlepší systém StarUML, kterému chybí některé důležité možnosti v oblasti návrhu diagramů, a proto nebyl použit. Z volně použitelných verzí se vybíralo ze systémů ArgoUML, Posseidon a Visual Paradigm s comunity licencí. Pro návrh systému byl použit Visual Paradigma, protože nabízel nejmenší omezení oproti ostatním systémům.

Pro samotnou implementaci systému je vhodné použít vhodné integrované vývojové prostředí . Existují komerční projekty společnosti SUN Microsystems SUN ONE Studio a společnosti Borland JBuilder. Velmi dobré vlastnosti mají i nekomerční systémy jsou NetBeans (viz. obr. č.4), Eclipse společnosti IBM a JDeveloper společnosti Oracle. Pro implementaci se použilo vývojové prostředí systému NetBeans z důvodu dobrých zkušeností autora projektu. Tento systém nabízí vše potřebné a vhodné pro vývoj podobných projektů. Ze systému NetBeans je odvozeno i komerční integrované vývojové prostředí SUN ONE Studio.



Obr.4. Integrované vývojové prostředí NetBeans.

Zdroj: Autor

4.1.3 Princip logické funkčnosti aplikace

K webové aplikaci se standardně přistupuje pomocí tenkého klienta, který je obvykle realizován jako webový prohlížeč. Koncovému uživateli se tedy bude systém jevit jako soubor webových stránek. Technologie Java umožňuje několik přístupů, jak logicky rozdělit aplikaci. Základní systém programování webových systémů v Java SE spočívá ve využití stránek jsp a servletů. Ty lze podpořit dalšími vhodnými technologiemi jako jsou JavaBeans a deklarace vlastních značek pomocí XML notace. Aplikace byla původně prototypována čistě na principu servletů a jsp stránek. Později bylo rozhodnuto o využití technologie java server faces a konečná verze aplikace byla podle toho překódována. S tím se částečně změnil i návrh systému. Veškerý aplikační kód je ukryt v java beans a k nim přidruženým třídám. Jsp stránky slouží pouze jako uživatelské rozhraní a nástroj pro prezentaci dat. Až na malé výjimky neobsahují výkonný kód.

Java má velmi dobře vyřešeno předávání parametrů a správu prostředků pro aplikaci při komunikaci mezi servlety či stránkami a servlety. Požadavky spojené s předáním parametrů jsou u servletu zapouzdřeny v objektu Request. Veškeré prostředky související s výstupem a s požadavky

směřujícími ze servletu jsou zapouzdřeny v objektu response. Ostatní prostředky, které souvisejí s nastavením servletu samotného, jsou zapouzdřeny v objektu ServletContext. Pomocí těchto objektů lze snadno operovat se všemi zdroji aplikace, jako jsou například nastavování sessions, cookies a atributů předávaných mezi servlety, stránkami, či požadavky, hlavičky stránek, i samotný výstup generované webové stránky.

Na stránkách jsp je Java kód ohraničen značkami <% a %>. Vzhledem k tomu, že Java kontejner ze stránky jsp generuje servlet, který na něm pak běží, může se Java kód libovolně prolínat s kódem značkovacího jazyka (například HTML, XHTML, atd.). Prostředky, se kterými se na webové stránce pracuje, jsou přístupné pomocí zvláštních systémových proměnných.

Správně navržená webová aplikace napsaná v Javě má použít jsp stránky tam, kde je málo Java kódu a hodně jiného výstupu. Servlety by se měly použít pro zpracování požadavků a následné reakce hlavně tam, kde je hodně Java kódu a současně jen málo, nebo žádný výstup. Při rozsáhlejších aplikacích, kde je potřeba důrazněji oddělit Java kód od designu aplikace, je vhodné využít vlastní třídy JavaBeans, nebo vlastní značky. Technologie Java EE nabízí i další vhodné vlastnosti, například perzistentní objekty a další výkonné technologie, ale vzhledem k rozsáhlosti projektu nejsou použity.

Technologie java server faces je určená pro prezentaci dat a pro snadnější implementaci pokročilého uživatelského rozhraní. Jedná se o technologii postavenou na javascriptu a ajaxu. Díky této technologii lze i na bezstavovém protokolu využít zpracování některých událostí. Aplikaci tak lze řídit bez toho, že by se při každé změně odesílal na server celý požadavek na změnu stránky a kompletně celá stránka se znovunačítala. Navigaci i samotný obsah stránky lze tak řídit mnohem jednodušeji a rychleji. Na server se vždy posílají jen nejnutnější data a je nejnutnější data se posílají ze serveru na klienta.

Samotné značky technologie java server faces využívají jmenných prostorů a notace XML. Z hlediska uživatelského rozhraní nabízejí stejné prvky jako běžný jazyk HTML. Prvky jazyka HTML lze také se značkami JSF míchat. Samotný zápis značek JSF je mírně složitější než zápis značek HTML, a proto se vyplatí jen pokud u prvků potřebujeme propojení na aplikační logiku. Příslušné Java třídy, které jsou použité jako řídící objekty aplikace (managed bean), je třeba zapsat do konfiguračního souboru faces.xml spolu s navigačními pravidly pro přesun mezi webovými stránkami a různými posluchači registrovanými do životního cyklu aplikace.

4.1.4 Bezpečnost aplikace

4.1.4.1 Důležitost bezpečnosti aplikace

Při návrhu i implementaci aplikace je nutné zabývat se bezpečností aplikace. Webová aplikace je přístupná prostřednictvím internetu a poměrně zranitelná. Rizikovými místy aplikace jsou všechny uživatelské vstupy a adresa URL. Útočník se obvykle snaží dostat různými způsoby do databáze nebo dostat do aplikační logiky vlastní fragmenty kódu. Je také možný útok za účelem zcizení zdrojového

kódu. Povaha projektu automatického vyhodnocování e-learningových testů není taková, aby bylo potřeba věnovat bezpečnosti vysokou míru pozornosti, a proto jsou provedeny jen základní dostačující ochrany proti běžným útokům.

4.1.4.2 Obvyklé útoky na webové systémy

Největší slabinou webového systému je nechráněný uživatelský vstup. Do tohoto vstupu se útočník může snažit dostat vlastní fragmenty kódu, který v systému plní nežádoucí funkce. Vzhledem k tomu, že uživatelské vstupy jsou často vstupem pro parametry použité v dotazech na databázi, může se útočník pokoušet vložit do nich vlastní fragment SQL dotazu, a tím ovládat databázi systému. Tato technika se nazývá SQL Injection a je uplatnitelná nejen na uživatelské vstupy, ale i na parametry předávané pomocí URL, tedy pomocí požadavku GET protokolu HTTP. Útoky na URL či vstupy je možné zautomatizovat pomocí různých exploitů.

4.1.4.3 Ochrana proti běžným útokům

V aplikaci, která má být odolná proti běžným útokům, je třeba pečlivě ošetřit uživatelské vstupy a parametry předávané pomocí URL adresy. U ošetření vstupů je potřeba především nepovolit znaky apostrofů, uvozovek a těchto „<, >, &“ znaků. Tato funkce se dá snadno implementovat pomocí filtrace html entit, tedy projít řetězec vstupů znak po znaku a nahradit příslušné znaky notacemi &amp pro „&“, &lt a &gt pro „< a >“ a &quot pro apostrofy. Omezením apostrof se ošetří riziko SQL injection, ošetřením ostatních znaků riziko vkládání vlastních fragmentů kódu. Pro případ, že by se útočník dostal do databáze, je třeba neukládat hesla k účtům v otevřené podobě, ale jen zašifrovaně. Pro tyto účely se vytváří tzv. HASH hesla. Java tuto funkci podporuje a v projektu je pro zašifrování hesel použit algoritmus SHA-1.

4.1.4.4 Ochrana proti útokům s využitím JSF

Technologie java server faces nabízí pokročilé možnosti ošetření uživatelských vstupů. Vzhledem k tomu, že vstupní formulářová pole jsou prostřednictvím řídících bean navázána na atributy tříd, které mají jasně definovány datové typy, prakticky nelze vložit jakýkoli nepředpokládaný vstup. Přidávání kódu do URL prohlížeče nemá efekt z důvodu jinak řešeného zpracování parametrů a odesílání dat na server. Navigace stránek probíhá podle navigačních pravidel a v posluchačích událostí registrovaných do životního cyklu aplikace lze implementovat různá omezení a další ochranné mechanismy. Pro zvýšení bezpečnosti i komfortu při práci s uživatelským rozhraním lze pomocí značek JSF svázat s uživatelskými vstupy různé konvertory či validátory nebo si vynutit vyplnění daného formulářového pole.

4.2 Návrh systému

4.2.1 Zpracování návrhu systému

Pro návrh systému je použit modelovací jazyk UML 2.0. Diagramy byly vypracovány v programu Visual-Paradigm Community Edition. Vzhledem k povaze projektu, rozsáhlosti projektu a velikosti vývojového týmu je návrh dopracován pouze do nutné míry. V této práci není uveden celý návrh, ale jen nutné informace a nejdůležitější diagramy.

4.2.2 Požadavky na projekt

Požadavky na projekt jsou součástí zadání a obvykle obsahují různé 3 dokumenty, které s různou mírou abstrakce upřesňují, co se od vyvíjeného systému očekává. Tyto dokumenty se nazývají prohlášením o cílech, dokumentem požadavků na systém a specifikačním dokumentem. Vzhledem k povaze a velikosti vyvíjeného systému byly požadavky pouze stručně sepsány způsobem, který je v souladu s doporučeními zkušených vývojářů z komunity UML.

4.2.2.1 Funkční požadavky

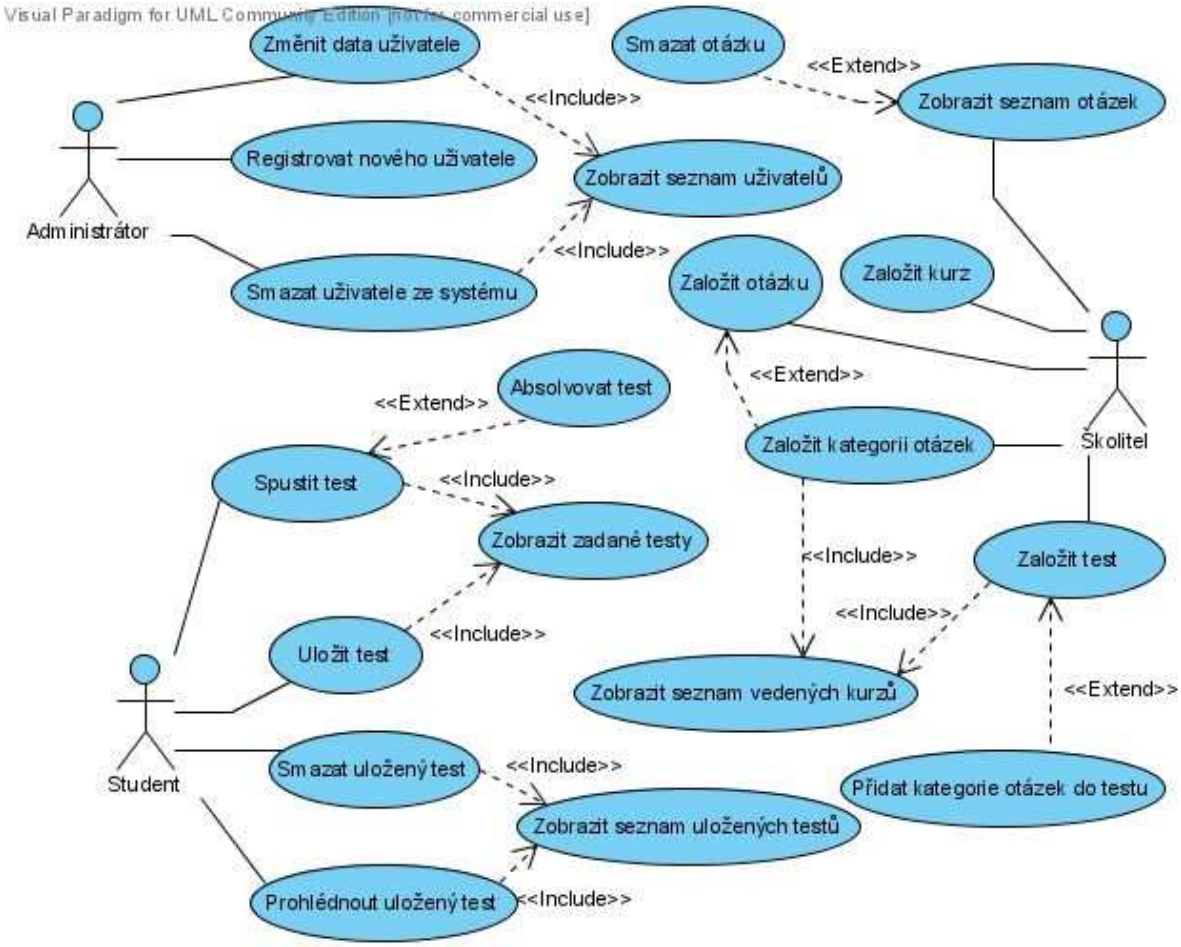
- Správa různých rolí uživatelů
- Zakládání různých testů a vedených předmětů.
- Specifikace různých kategorií otázek
- Testy s víceslovními odpověďmi
- Automatické vyhodnocení vypracovaných testů
- Podrobný záznam o průběhu testu

4.2.2.2 Non-funkční požadavky

- Webové prostředí, aplikace klient-server.
- Použití technologií Java, MySQL, XHTML
- Platforma Containeru TOMCAT

4.2.3 Modelování případů užití

Základním kamenem návrhu systému je diagram případu užití (USE CASE). Tento model popisuje základní případy užití systému různých rolí uživatelů (viz. Obr. 5). K diagramu patří modelování scénářů užití, ze kterých se později vyvíjejí diagramy tříd. Scénáře popisují posloupnost akcí které nastávají při realizaci případu užití. Tyto scénáře pro svůj rozsah nejsou v této práci prezentovány. Model případů užití využívá vysoké úrovně abstrakce a je jednoduchým vyjádřením toho, co se od systému očekává.



Obr.5. Diagram případů užití.

4.2.3.1 Scénáře případů užití

Scénáře případů užití jsou základním úkonem při návrhu systému. Popisují typové úlohy systému tak, jak je zadavatel zadal. Tyto scénáře je možné propojit s modelem aktivit a jsou výchozím bodem pro modelování analytických tříd. Z důvodu velké obsáhlosti je v práci otištěno jen několik vybraných scénářů. Jedná se o kompletní scénáře úkonů administrátora systému a nejdůležitější úkony školitele, těmi je registrace testů a registrace otázek.

ID: 1 Registrace nového uživatele

Role uživatelů: Administrátor systému

Vstupní požadavky: přihlášený administrátor systému.

Výstupní požadavky: V systému je zaregistrován nový uživatel.

1. Uživatel dá pokyn k registraci nového uživatele.
2. Systém zobrazí formulář registrace nových uživatelů.
3. Uživatel vyplní údaje nového uživatele. Těmi jsou: jméno, příjmení, uživatelské jméno, heslo a rok narození. Po vyplnění údajů formulář potvrdí tlačítkem.
4. Systém zkontroluje zadaná data.

4.1 POKUD některé z požadovaných dat chybí, nebo nesplňují požadovanou délku či datový typ, TAK Systém vypíše chybové hlášení u příslušného pole.

4.2 JINAK systém zjistí unikátnost zadávaného uživatelského jména.

4.2.1 POKUD je uživatelské jméno obsazeno, TAK Systém vypíše chybové hlášení a nepokračuje.

4.2.2 JINAK systém vloží data uživatele do datového úložiště a vypíše hlášení o úspěchu.

ID: 2 Zobrazení uživatelského seznamu

Role uživatelů: Administrátor systému

Vstupní požadavky: přihlášený administrátor systému.

Výstupní požadavky: Zobrazí se seznam uživatelů.

1. Uživatel dá pokyn k zobrazení seznamu uživatelů.

2. Systém načte data všech uživatelů a zobrazí je do přehledné tabulky. Tabulka obsahuje uživatelská jména, jména, příjmení a data narození uživatelů. Krom toho nabízí i funkce smazání uživatele a změnu uživatelských dat.

ID: 3 Zobrazení uživatelského seznamu

Role uživatelů: Administrátor systému

Vstupní požadavky: přihlášený administrátor systému, zobrazený seznam uživatelů.

Výstupní požadavky: Smazaný uživatel.

0. Vložený scénář ID 2.

1. Uživatel dá pokyn ke smazání uživatele.

2. Systém uživatele smaže.

ID: 4 Změna dat uživatele

Role uživatelů: Administrátor systému

Vstupní požadavky: přihlášený administrátor systému, zobrazený seznam uživatelů.

Výstupní požadavky: Data uživatele byla změněna.

0. Vložený scénář ID 2.

1. Uživatel dá pokyn ke změně dat uživatele.

2. Systém zobrazí formulář s předvyplněnými daty měněného uživatele. Těmito daty jsou: uživatelské jméno, křestní jméno, příjmení, heslo a datum narození.

3. Uživatel provede zamýšlené změny a formulář potvrdí.

4. Systém zkontroluje zadaná data.

4.1 POKUD některé z požadovaných dat chybí, nebo nesplňují požadovanou délku či datový typ, TAK Systém vypíše chybové hlášení u příslušného pole.

4.2 JINAK systém zjistí unikátnost zadávaného uživatelského jména.

4.2.1 POKUD je uživatelské jméno obsazeno, TAK Systém vypíše chybové hlášení a nepokračuje.

4.2.2 JINAK systém vloží data uživatele do datového úložiště a vypíše hlášení o úspěchu.

ID: 11 Založení kategorie otázek

Role uživatelů: Školitel

Vstupní požadavky: Přihlášený školitel, zobrazený seznam testů nebo kurzů nebo registrace otázky.

Výstupní požadavky: Zobrazený formulář kategorií, případně zaregistrování nových kategorií.

1. Uživatel dá pokyn k zobrazení kategorií otázek.

2. Systém dohledá data všech kategorií svázaných s vybraným předmětem a zobrazí názvy kategorií v přehledné tabulce. Tabulka obsahuje tlačítko pro smazání kategorie (i všech jejích otázek) ze systému. Nad tabulkou se zobrazí formulář, který umožní do vedeného předmětu (který byl vybrán dříve) přidávat další kategorie. Formulář obsahuje pole pro název kategorie.

3. Uživatel vyplní a potvrdí název nové kategorie.

4. Systém zadaná data zkontroluje.

4.1 POKUD není kategorie zadána, nemá správnou délku nebo datový typ TAK: Systém zobrazí chybové hlášení a nepokračuje.

4.2 JINAK systém zapíše kategorii do datového úložiště. Název kategorie se zobrazí v tabulce kategorií pod formulářem.

ID: 12 Založení otázky

Role uživatelů: Školitel

Vstupní požadavky: Přihlášený školitel, zobrazený seznam testů.

Výstupní požadavky: Zaregistrována nová otázka.

1. Uživatel dá pokyn k registraci nové otázky

2. Systém zobrazí formulář pro registraci otázky. Tento formulář obsahuje rozbalovací seznamy pro zadání typu otázky (výčtová, posloupnost, množina), kategorie otázky a obtížnosti (10 typů). Dále obsahuje vstupní pole pro text otázky, správnou odpověď, minimální počet správně zodpovězených slov (jen u výčtových otázek) pro uznání otázky nebo procentuální hranici úspěšnosti pro uznání otázky (u posloupnosti a množiny). Posledním prvkem je vstupní pole pro hledané slovo. Při zadání hledaného slova se pod formulářem zobrazí seznam hledaných slov, umožňující přidávání synonym ke hledanému slovu a mazání těchto hledaných slov.

3. Uživatel vyplní všechny data otázky.

4. Uživatel zadá první hledaný výraz a potvrdí ho.

5. Systém zobrazí tabulku hledaných výrazů, ve které nabídne smazání výrazu a přidání synonym.

6. SOUBĚŽNĚ A, B, C

6.A.1 Uživatel zadá další hledané slovo.

6.A.2 Systém ho zobrazí do řádku tabulky.

6.B.1 Uživatel zadá synonymum k hledanému výrazu do vstupního pole na řádku tabulky hledaných výrazů a potvrdí.

6.B.2 Systém přidá slovo k řetězci hledaných výrazů. Jako oddělovač se použije čárka „ , “.

6.C.1 Uživatel dá pokyn ke smazání hledaného výrazu.

6.C.2 Systém odstraní záznam z tabulky.

7. Uživatel potvrdí formulář specifikace otázky

8. Systém zkontroluje všechna zapsaná data

8.1 POKUD otázka neobsahuje text otázky, nebo má specifikovaných méně hledaných výrazů, než je potřeba (minimálně jeden), TAK systém zobrazí chybovou zprávu a nepokračuje.

8.2 JINAK Systém založí otázku i hledané řetězce do datového úložiště a restartuje formulář.

ID: 13 Založení testu

Role uživatelů: Školitel

Vstupní požadavky: Přihlášený školitel, zobrazený seznam vedených kurzů.

Výstupní požadavky: Zaregistrován nový test.

1. Uživatel dá pokyn k registraci nového testu.

2. Systém zobrazí formulář se vstupními poli pro zkratku testu a název testu. Formulář dále obsahuje dva rozbalovací seznamy. První seznam nabízí procentuální nastavení hranice uznání testu jako úspěšně absolvovaného (0% až 100%). Druhý seznam nabízí registrované kategorie. Ty jdou pomocí tlačítek „Přidat kategorii“ přidat do testu.

3. Uživatel specifikuje zkratku a název testu. Dále specifikuje hranici úspěšnosti. Vybere kategorii otázek, kterou chce v testu mít a stiskne tlačítko pro přidání kategorie do testu. To lze provést vícekrát a kategorie se mohou opakovat.

4. Systém pod formulářem zobrazí podformulář s názvem kategorie a nastavením strategie průchodu testem. Tento podformulář obsahuje: vstupní pole pro specifikaci maximálního počtu otázek, které budou pro danou kategorii položeny a vstupní pole pro počet správně zodpovězených otázek, po jehož dosažení se testování přesune na další kategorii. Dále obsahuje rozbalovací seznam s nabídkou strategie výběru první otázky kategorie a druhý rozbalovací seznam s nabídkou strategie výběru dalších otázek. Podformulář obsahuje i funkci vyjmutí kategorie z testu.

5. Uživatel vloží do systému všechny kategorie, které chce v testu mít a u všech nastaví strategii. Pak celý formulář potvrdí.

6. Systém zkontroluje zadaná data

6.1 POKUD nemají některá data požadovanou délku, nebo datový typ, TAK Systém zobrazí příslušnou chybovou zprávu a nepokračuje.

6.2 JINAK zkontroluje unikátnost zkratky a jména testu.

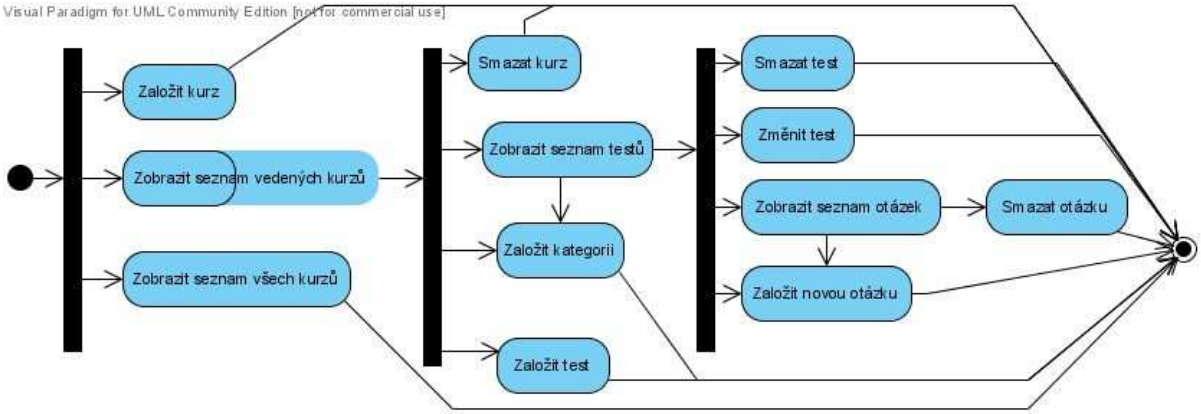
6.2.1 POKUD je zkratka i název testu unikátní, systém data testu vloží do datového úložiště a zobrazí zprávu o úspěchu.

6.2.2 JINAK Systém vypíše hlášení o obsazenosti zkratky nebo názvu testu a nepokračuje.

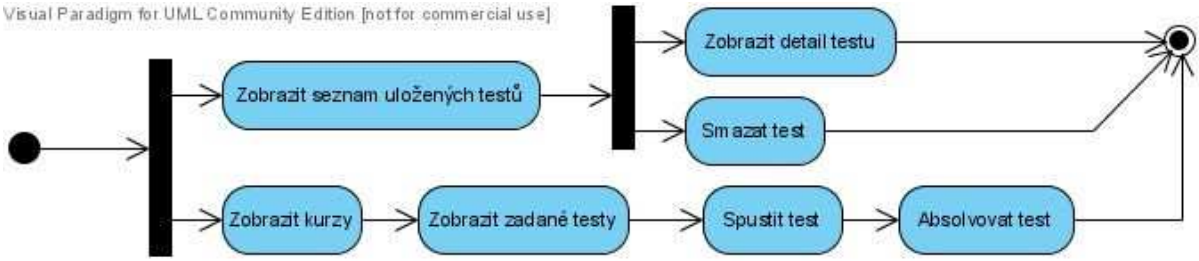
4.2.3.2 Návaznost případů užití

Přehled návaznosti jednotlivých případů užití ukazují diagramy aktivit rolí administrátora (viz. Obr. 6), školitele (viz. Obr. 7) a studenta (viz. Obr. 8).

Obr. č. 6. Aktivity administrátora systému



Obr. č. 7. Aktivity školitele v systému

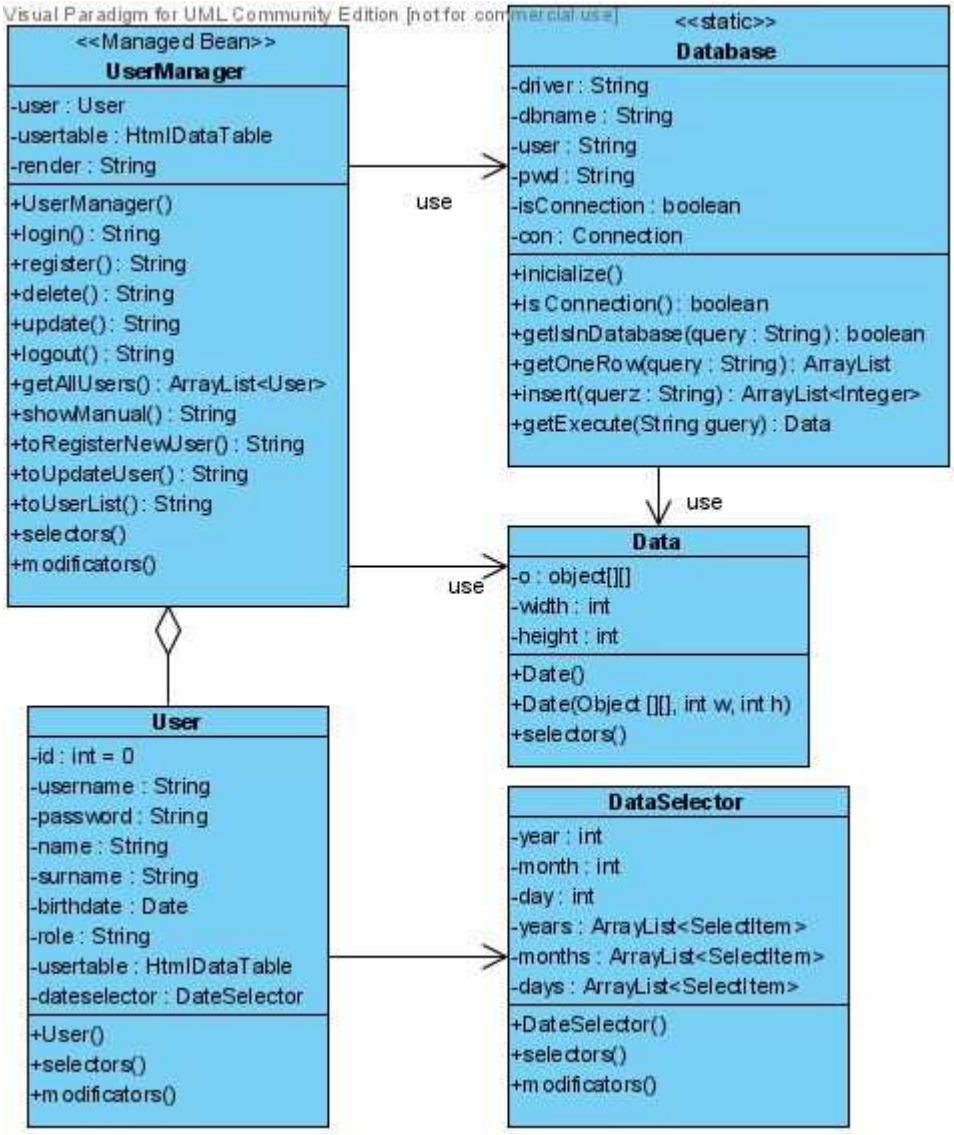


Obr. č. 8. Aktivity studenta v systému

4.2.4 Návrh architektury tříd systému

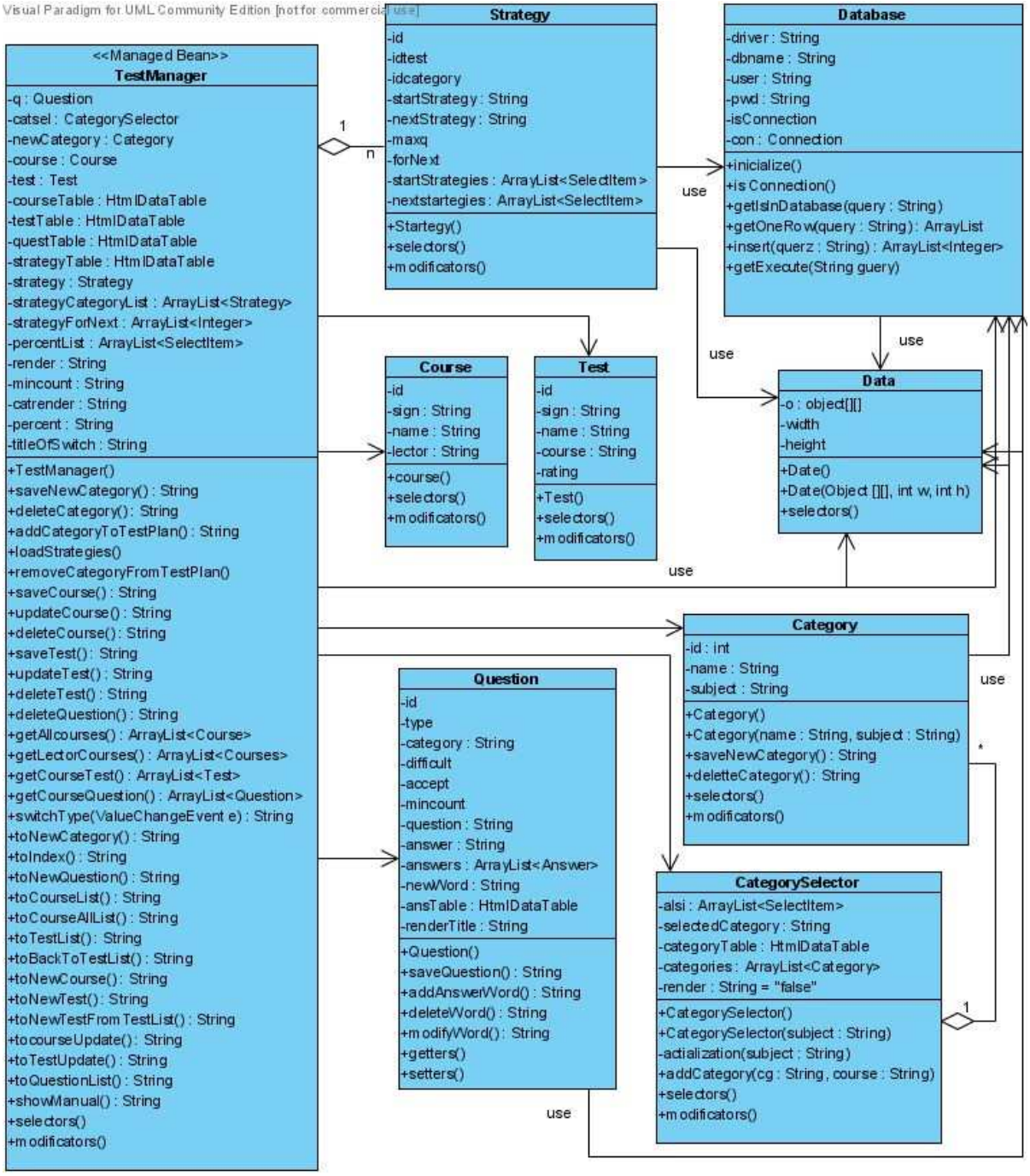
Třídy systému se navrhují v několika krocích a jsou stěžejní částí systému. Návrh diagramu tříd je náročná a složitá práce, která klade vysoké požadavky na analytické schopnosti vývojáře. Existuje více postupů, jak nalézt potřebné třídy a jejich obsah. Vhodnou metodou je analýza slov používaných v modelovaném reálném systému. Při této metodě se vyhledávají podstatná jména a slovesa a rozhoduje se o jejich umístění do tříd, atributů tříd a vztahů. Jiné výkonné metody, jakou je např. metoda štítků CRC,jsou vhodné při práci ve větším kolektivu. Pro přehlednost byl výsledný diagram rozdělen na tři diagramy tříd, které modelují třídy pro každou roli systému a které jsou mírně zjednodušené. Celkový diagram tříd by byl značně nepřehledný a nelze ho v této práci otisknout z důvodu vysokých prostorových nároků.

Nejjednodušší diagram tříd se zabývá modelováním tříd spojených s akcemi administrátora systému (viz obr. 9).



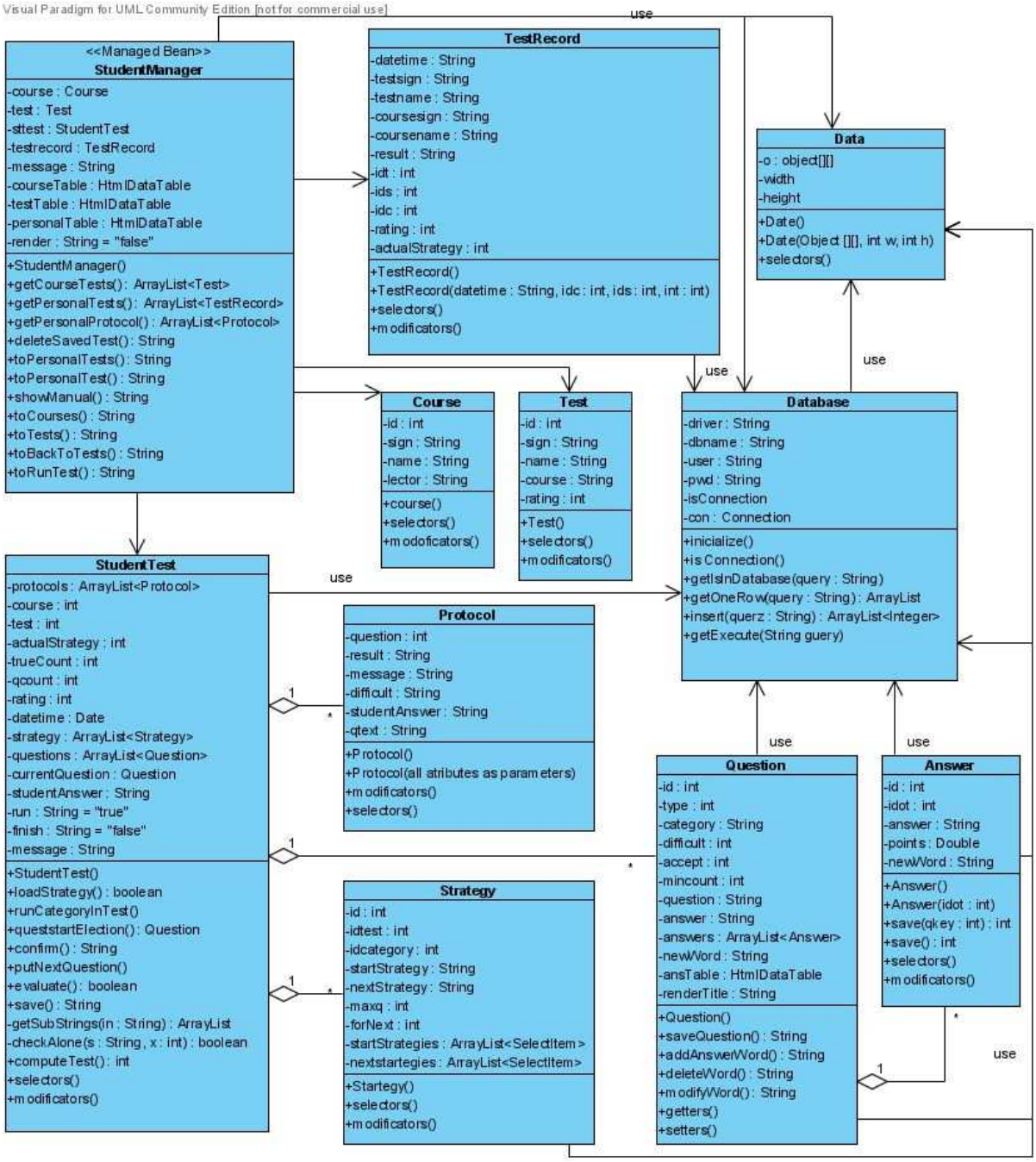
Obr. 9. Návrhový diagram tříd spojených s rolí administrátora systému.

Další diagram tříd se zabývá rolí školitele (viz. Obr.10).



Obr.10. Návrhový diagram tříd spojený s rolí školitele.

Třetí diagram tříd se zabývá rolí studenta.

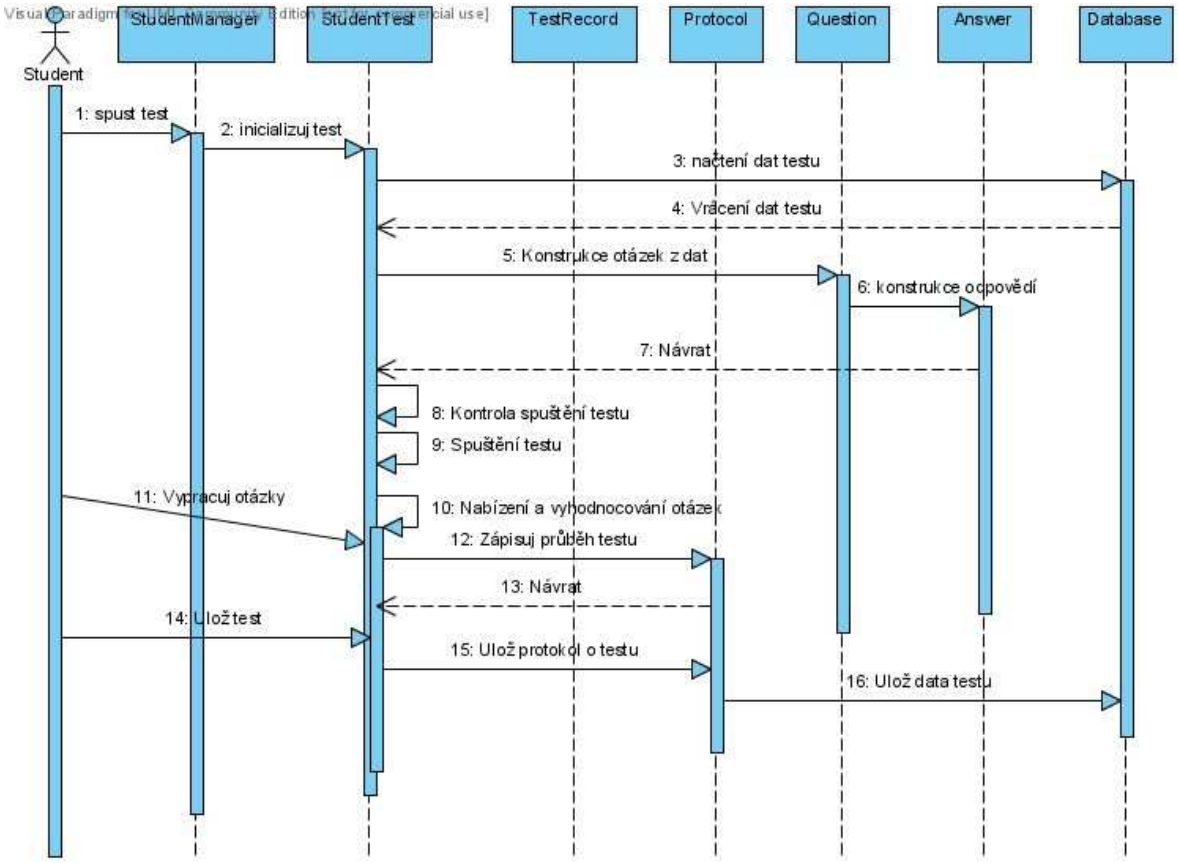


Obr.11. Návrhový diagram tříd spojený s rolí studenta.

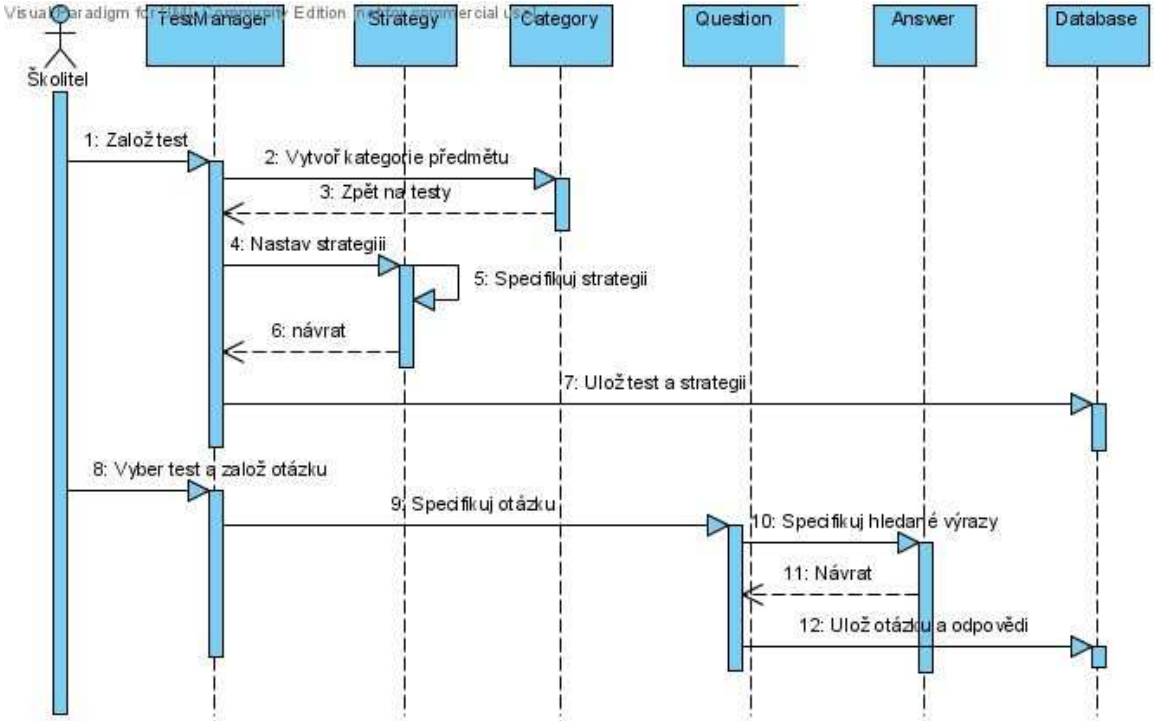
4.2.5 Návrh objektové spolupráce objektů

Jazyk UML umožňuje různé způsoby, jak modelovat objektovou spolupráci objektů. Ve své podstatě jde o návrh vnitřního logického chování systému. Toto chování lze modelovat jednak pomocí diagramů aktivit, stavovými stroji, diagramy interakce, komunikačními diagramy nebo sekvenčními diagramy. Při návrhu systému se vytvoří velké množství diagramů, které nějakým způsobem modelují interaci tříd. Pro prezentaci důležitých procesů byly vybrány některé klíčové sekvenční diagramy.

Jedná se o zjednodušený diagram testování studenta (viz. Obr. 12) a zjednodušený diagram zakládání testů, kategorií otázek a otázek (viz. Obr. 13).



Obr. 12. Sekvenční diagram testování studenta.



Obr. 13. Sekvenční diagram zakládání testů, kategorií a otázek.

4.2.6 Rozvržení návrhu do balíčků

Třídy, ze kterých se systém skládá, jsou rozděleny do čtyř balíků. Balíky jsou obsahově orientovány podle rolí uživatelů v systému, protože z hlediska aplikační logiky se tento přístup jevil vhodný. Projekt je zabalen do webového archivu způsobem obvyklým pro webové projekty jazyka Java.

4.2.6.1 Balík tříd student

Balík student obsahuje třídy související se samotným testováním studentů. Obsahem balíku jsou třídy:

- StudentManager (řídící bean)
- StudentTest (hlavní třída testů)
- TestRecord (záznam o uloženém testu)
- Protocol (záznam o hodnocení v průběhu testu)

4.2.6.2 Balík tříd test

Balík tříd test obsahuje největší množství tříd. Jedná se o třídy související se zadáváním testů, určováním strategií pro testy, zadáváním otázek, odpovědí, zakládáním kategorií a podobně. Obsahuje tyto třídy:

- TestManager (řídící bean)
- Category (kategorie otázek)
- CategorySelector (třída výběru kategorií na stránce JSF)
- Course (data kurzu)
- Question (data otázky)
- Strategy (data strategie)
- Test (data testu)
- Answer (data hledaných výrazů)

4.2.6.3 Balík tříd user

Balík user obsahuje třídy spojené s prací s uživateli a řízením přístupu.

- UserManager (řídící bean)
- User (data uživatele)

4.2.6.4 Balík tříd utility

Poslední balíček obsahuje třídy použitelné na mnoha místech systému.

- Data (třída zastřešující obecná data načítaná z databáze)
- Database (vykonávání dotazů nad databází)
- DateSelector (výběr data na stránce JSF)
- autentificationListener (naslouchač který, implementuje řízení přístupu)

4.3 Implementace systému

4.3.1 Výpis konkrétních nástrojů použitých pro implementaci

Pro implementaci systému byl zvolen jazyk Java SE verze 1.6 [z1]. K jazyku byly připojeny knihovny Java Server Faces. Jako integrované prostředí byl použit systém NetBeans 6.0.1 [z2] s pluginy pro vývoj webových aplikací. Pomocí containeru TOMCAT, který byl pro tento účel integrován do prostředí NetBeans.. Jako databáze bylo použito MySQL 5.0 [z3] a administrátorské prostředí pro tento systém MySQL GUI TOOLS [z4]. Dále byla využívána dokumentace k jazyku JAVA [14] a kniha Java Servlety a stránky JSP. [15] Jako základ webových stránek systémy byl použit značkovací jazyk XHTML 1.0 Strict [z5]. Pro formátování webových stránek je použita technologie CSS [z6]. Pro určitou funkci na formulářích zadávání otázek je použit JavaScript [z7].

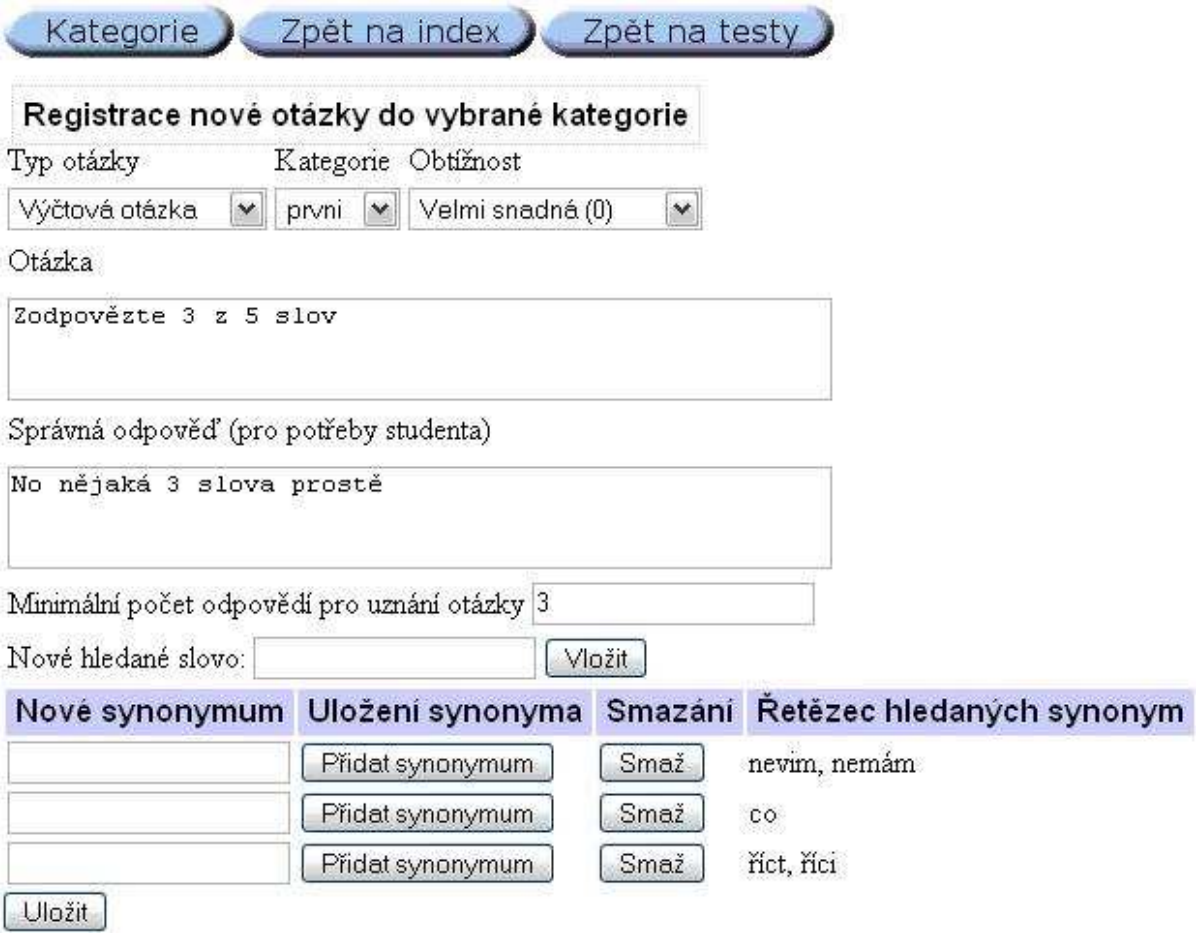
4.3.2 Popis algoritmu vyhodnocování testů

4.3.2.1 Definice otázky

V systému existují 3 různé typy otázek. Prvním typem jsou otázky typu „výčet“, což je typ, který definuje nutnost nalezení určitého počtu odpovědí z určitého vyššího možného počtu. Druhým typem jsou otázky typu „posloupnost“, kde se vyhledávají víceslovné odpovědi s pevně definovaným pořadím. Třetím typem otázek je „množina“, kde se vyhledávají víceslovné odpovědi a kde nezáleží na pořadí nalezených slov.

Otázku je třeba přiřadit do určité kategorie, která musí být založená. Pokud jsou k danému předměty definovány kategorie, pak je systém automaticky dohledá a nabídne. Další nastavení u otázky je obtížnost. Obtížností je celkem deset a slouží hlavně při určování strategie průchodu kategorií při testování. Další položkou je samotná otázka. Je vhodné definovat i správnou odpověď pro studenta, která se zobrazí v protokolu. U výčtových otázek se nastavuje minimální počet slov který je třeba odpovědět pro uznání otázky. U ostatních typů je tento element vyjádřen procentuálním poměrem.

Ke každé otázce je třeba definovat množinu hledaných výrazů. To se dělá postupně přikládáním slov, ke kterým lze dále definovat různá synonyma. U výčtových otázek je třeba, aby se počet hledaných odpovědí minimálně rovnal počtu odpovědí definovaných pro správnou odpověď. Ostatní otázky musí mít definováno alespoň jedno hledané slovo.



Obr. 14. Zadávání otázek

4.3.2.2 Obecný princip algoritmu

Algoritmus je založen na myšlence hledání výrazů, které obsahuje správná odpověď, v textu odpovědi studenta. Protože každý hledaný výraz může mít více synonym a alternativních vyjádření, je třeba definovat a vyhledávat i tyto alternativní výrazy. Algoritmus neumí sám automaticky zpracovávat výrazy odvozené skloňováním a jinými gramatickými operacemi českého jazyka.

Při konstrukci testu jsou do testy vloženy kategorie otázek a pro každou z nic se určuje strategie průchodu. Plánování strategie je jednoduché a zahrnuje zadání maximálního počtu otázek, které jsou z dané kategorie položeny, minimálního počtu správně zodpovězených otázek, po kterých následuje posun na další kategorii (nemá smysl ptát se podrobně na kategorii, ve které si je student jistý), strategii výběru první otázky kategorie a nakonec strategie výběru dalších otázek. První otázka kategorie se může vybírat náhodně nebo podle obtížnost z lehčích, středně těžkých nebo těžkých otázek. Výběr dalších otázek je možné volit náhodně nebo se mohou otázky při správných odpovědích zlehčovat a při dobrých ztížit nebo naopak.

Základním kamenem algoritmu je vždy rozhodnout, zda je odpověď zcela správná, nebo zcela špatná. Body za částečné odpovědi se nepočítají a nepočítají se ani za otázky, protože test může mít různé množství otázek, které se určuje na základě znalostí studenta.

4.3.2.3 Typ otázky výčet (n odpovědí z m možných)

1, K položené otázce se vyhledá seznam všech hledaných řetězců synonym.

2, Seznam hledaných synonym se postupně prochází:

2.1, Aktuálně vybraný řetězec synonym se rozloží na kolekci jednotlivých výrazů.

2.2, Celá kolekce výrazů se postupně prochází

2.2.1, Každý výraz se převede na malá písmena.

2.2.2, Provede se vyhledání výrazu v řetězci odpovědi studenta.

2.2.2.1, POKUD se nalezne, provede se test samostatnosti nalezeného slova.

2.2.2.1.1 POKUD je slovo samostatné, bere se výraz za nalezený a započítá se.

2.2.2.1.2 JINAK se slovo za nalezené nepočítá.

2.2.2.2 JINAK se pokračuje dále dalším cyklem

3, Systém porovná počet nalezených odpovědí s uznatelným počtem u otázky.

3.1, POKUD je uznatelný počet nižší, otázka se započítá jako správná.

3.2 JINAK se jako správná nezapočítá.

4.3.2.4 Typ otázky posloupnost – víceslovná odpověď – záleží na pořadí

1, K položené otázce se vyhledá seznam všech hledaných řetězců synonym.

2, Seznam hledaných synonym se postupně prochází:

2.1, Aktuálně vybraný řetězec synonym se rozloží na kolekci jednotlivých výrazů.

2.2, Celá kolekce výrazů se postupně prochází

2.2.1, Každý výraz se převede na malá písmena.

2.2.2, Provede se vyhledání výrazu v řetězci odpovědi studenta.

2.2.2.1, POKUD se nalezne, provede se test samostatnosti nalezeného slova.

2.2.2.1.1 POKUD je slovo samostatné, bere se výraz za nalezený a započítá se. Zapíše se pořadí.

2.2.2.1.2 JINAK se slovo za nalezené nepočítá.

2.2.2.2 JINAK se pokračuje dále dalším cyklem

3, Systém porovná poměr nalezených odpovědí k nalezitelným s uznatelným poměrem u otázky.

3.1, POKUD je uznatelný počet nižší, zkontroluje se pořadí.

3.1.1 POKUD pořadí odpovídá, je otázka uznána.

3.2 JINAK se jako správná nezapočítá.

4.3.2.5 Typ otázky množina - víceslovná odpověď – nezáleží na pořadí

1, K položené otázce se vyhledá seznam všech hledaných řetězců synonym.

2, Seznam hledaných synonym se postupně prochází:

2.1, Aktuálně vybraný řetězec synonym se rozloží na kolekci jednotlivých výrazů.

2.2, Celá kolekce výrazů se postupně prochází

2.2.1, Každý výraz se převede na malá písmena.

2.2.2, Provede se vyhledání výrazu v řetězci odpovědi studenta.

2.2.2.1, POKUD se nalezne, provede se test samostatnosti nalezeného slova.

2.2.2.1.1 POKUD je slovo samostatné, bere se výraz za nalezený a započítá se.

2.2.2.1.2 JINAK se slovo za nalezené nepočítá.

2.2.2.2 JINAK se pokračuje dále dalším cyklem

3, Systém porovná poměr nalezených odpovědí k nalezitelným s uznatelným poměrem u otázky.

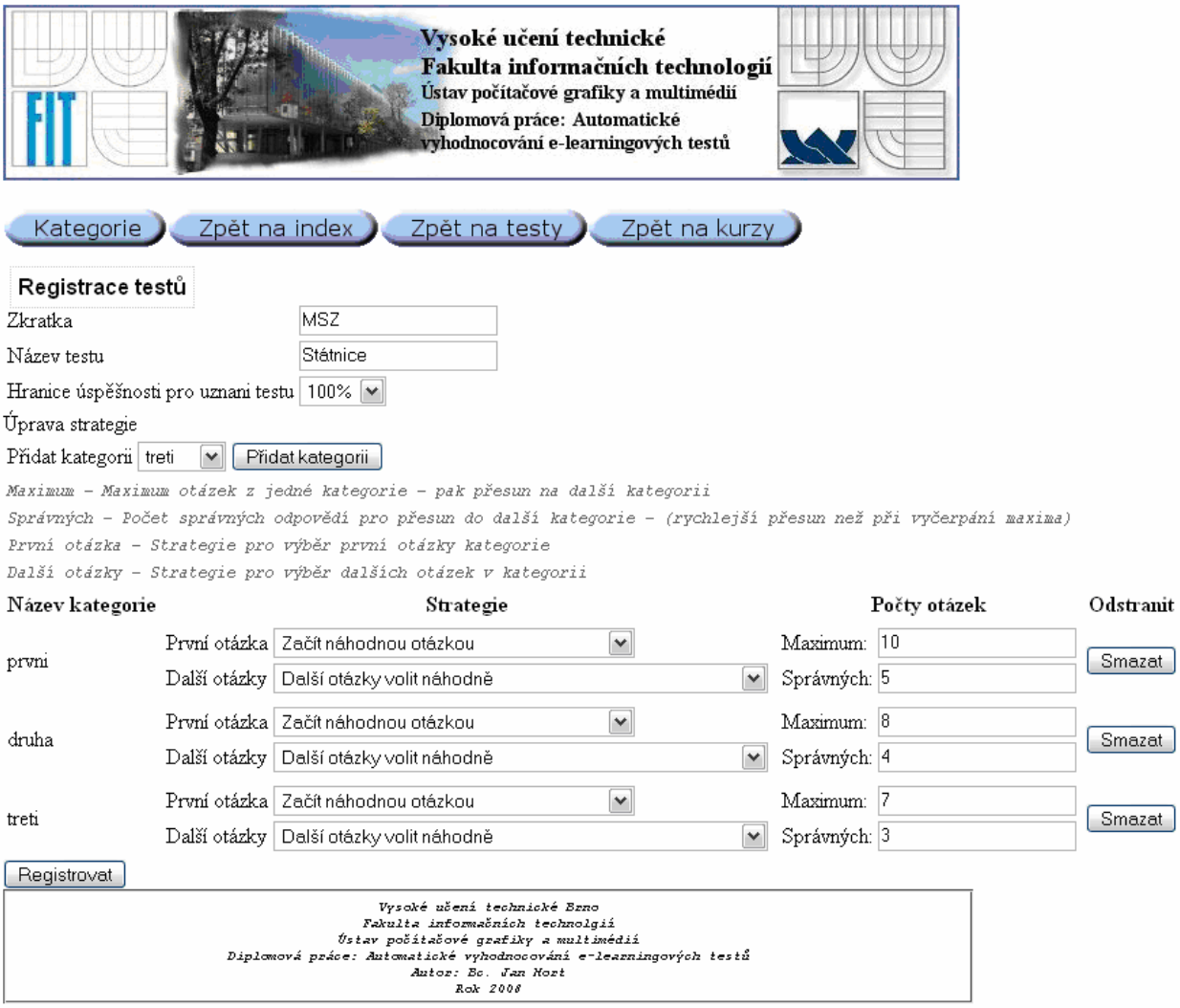
3.1, POKUD je uznatelný počet nižší, otázka se zapíše jako správná.

3.2 JINAK se jako správná nezapočítá.

4.3.3 Výsledný systém

Výsledkem diplomové práce je systém pro zpracování testů s víceslovními odpověďmi. Podařilo se vyřešit správné dohledávání slov a navrhnout jednoduchý, ale účinný způsob, jak specifikovat strategii průchodu grafem. Byly zkoušeny i složitější přístupy spočívající v počítání plovoucích ratingů a odvozování délky zkoušení jednotlivých kategorií z nich, ale první testy ukazovaly na problémy s pochopením uživatelů. Díky technologii JSF se podařilo implementovat poměrně bezpečné a přehledné grafické uživatelské rozhraní. Testy ukazují na poměrně rychlé zvládnutí systému uživateli, kteří s ním nikdy dříve nepřišli do styku. Celkové hodnocení od uživatelů je dobré.

V diplomové práci se nepovedlo vyřešit zpracování gramatiky, která by napomohla při generování synonym hledaných výrazů. Potřebné techniky byly zkoumány a nebyla nalezena žádná jednoduchá metoda, která by poskytovala dobré výsledky a usnadňovala zadávání hledaných výrazů do otázek. Z časových důvodů a důvodu náročnosti bylo nakonec od plánu udělal komponentu pro řešení tohoto problému upuštěno.



Obr. 15. Screenshot z vyvinutého systému.

Zdroj: autor

4.3.4 Hlavní problémy při implementaci

4.3.4.1 Problémy související s vývojem aplikace

Jedním z hlavních problémů při vývoji aplikace byl návrh samotného algoritmu pro vyhodnocování víceslovních odpovědí a souvisejících problémů. Předně bylo třeba dobře navrhnout způsob zadávání odpovědí k otázkám a jejich uložení do databáze. Nakonec bylo rozhodnuto o zadávání různých řetězců synonym hledaných slov, která byla ukládána do databáze jako celek a při hodnocení testu načítána hodnotícím algoritmem a rozkládána. Dalším problémem bylo vhodné rozvržení aplikace. Java nabízí mnoho možností, jak implementovat webovou aplikaci co se týče její vnitřní logiky, a bylo třeba dobře rozmyslet rozsah a vhodnost použitého přístupu. Další problémy, jako je pokročilé zpracování jazyka a adaptivní průchod grafem testu, byly odloženy pro další vývoj při řešení diplomové práce.

Třetím velkým problémem bylo navržení postupu definice strategie průchodu testem pro každý test. Bylo zkoušeno několik složitých přístupů, které se neosvědčily při testech s běžným uživateli. Nakonec došlo k výraznému zjednodušení konceptu tak, aby se udržela myšlenka inteligentního průchodu zkoušením studenta.

Největším a nevyřešeným problémem bylo zpracování gramatiky a vytvoření komponenty, která by generovala k zadanému hledanému výrazu synonyma, a tím usnadnila zadávání otázek.

4.3.4.2 Problémy s vývojovými nástroji

Největším problémem při vývoji aplikace bylo zpracování českého prostředí. Tento problém souvisel s chybou v implementaci v servlet containeru TOMCAT 5.5, který byl pro vývoj původně používán při zpracování znakové sady středoevropských jazyků iso-8859-2. Projevoval se při předávání parametrů komolením českých znaků. Tato chyba byla řešena nouzově tím, že byl nalezen a stanoven funkční postup zpracování předávaných parametrů. Předávanému formuláři byla nastavena znaková sada iso-8859-1 a po odeslání formulářových dat byla data z objektu požadavku Request převedena na pole Bytů u kterého se specifikovala znaková sada jako iso-8859-2 (čeština) a které se předalo jako parametr do konstruktoru objektu řetězce String daného atributu. Zajímavé je, že tento postup fungoval i při deklaraci obecné znakové sady, tedy znaky byly také české. Tento postup byl testován pouze na containeru TOMCAT verze 5.5 a nelze zaručit jeho funkčnost u jiných verzí containeru. Od verze TOMCAT 6.0 je zpracování češtiny stabilní a nevyskytly se větší problémy.

Problémem byla také migrace systému z prostředí localhostu, kde byl systém vyvíjen, na prostředí externího hostingu. Odlišné technologie a jazykové prostředí technologií použitých na straně hostingu mohou způsobit problémy se zpracováním češtiny a vykonávání kódu aplikace z důvodu například starší verze virtuálního stroje Javy. I komerční hosting měl problémy se zpracováním webové aplikace, ačkoli podle specifikace měl Javu podporovat. Nakonec byl systém nasazen na školní server, což přineslo některé nečekané problémy s nefunkčností některých komponent JSF, které byly nacházeny a řešeny v asi průběhu dvou dní.

4.4 Testování

4.4.1 Způsob testování systému

4.4.1.1 Testovací skupina a účel testování

Systém byl testován pomocí několika anonymních uživatelů různého věku a různého pracovního zaměření. Množina testerů zahrnovala středoškolské i vysokoškolské studenty, matku v domácnosti, učitelku, poštmistra, doktorku a informatika. Celkem skupina zahrnovala 8 osob. Testeři měli subjektivně zhodnotit to, zda systém odpovídá specifikaci. Tedy zda podle nich splňuje účel simulace

zkoušení formou psané písemné práce a další aspekty jako vzhled systému, jeho ovládání a celkový dojem.

4.4.1.2 Testovací dotazník

Pro hodnocení byl použit tento dotazník, kde testující osoby vyjádřily své dojmy ze systému.

Dotazník pro hodnocení systému

Zaměstnání či aktuální životní pozice hodnotícího:

Datum:

Zvýrazněte nebo jinak označte 1 možnost z testu

1, Grafický vzhled a ovládání systému shledávám:

A, Skvělé.

B, Dobré.

C, Dostatečné .

D, Špatné.

E, Vážně hrozné.

2, Způsob plánování testů (strategie průchodu testem) shledávám:

A, Skvělé, lepší mít nemůže.

B, Je dobré, některé věci bych změnil/a.

C, Docela dostatečné, mohlo být lepší.

D, Není dobré, je to nedomyšlené.

E, Vůbec se mi to nelíbí. Celé.

3, Způsob zadávání otázek shledávám:

A, Skvělý, je tam vše co je potřeba a jednoduše.

B, Docela dobré, mohlo by být i lepší.

C, Ujde to, chtělo by to další pomocné funkce.

D, Nelíbí se mi, není to domyšlené.

E, Je to vážně hrozné. Celé.

4, Způsob testování studenta shledávám:

A, Skvělým, skvěle provedeným.

B, Dobré, něco tomu chybí.

C, Je to dostatečné vzhledem k tomu, co to má dělat.

D, Je to nedostatečné, čekal/a jsem víc.

E, Úplně hrozné a špatně provedené.

5, Celkový systém shledávám:

A, Skvělým. Co dodat?

B, Je to trochu jednoduché, ale na to, co to má dělat, asi dobré.

C, Je to dosti jednoduché, chce to více funkcionality.

D, Není to moc povedené, skoro až špatné.

E, Vůbec se mi to nelíbí.

Napište slovní subjektivní hodnocení

6, Slovně zhodnoťte systém. To, jak dobře simuluje písemný test a jak jste spokojeni se správností vyhodnocování otázek. Také dodejte připomínky:

4.4.2 Dosažené výsledky

4.4.2.1 Hodnocení při testech

Výsledky testování dopadly poměrně dobře. Hodnocení systému mohlo být mírně ovlivněno vztahem testerů k autorovi systému a mírně ho ovlivnit směrem k lepšímu nebo horšímu. Všichni testeři však podle svých vyjádření hodnotili systém objektivně podle svého nejlepšího svědomí, tak jak byli požádáni autorem systému. Testeři jsou anonymní a pro rozlišení byly použity jejich iniciály. Dosažené výsledky zobrazuje tabulka TAB1.

Tester	***Otázka 1***	***Otázka 2***	***Otázka 3***	***Otázka 4***	***Otázka 5***
MW1	A	C	A	B	B
MW2	A	B	B	B	B
FK	A	C	A	B	B
FH	B	D	C	B	C
JP1	B	C	A	C	B
JP2	A	B	B	B	B
ŠN	C	C	B	B	B
MN	A	A	A	B	A

TAB1 – tabulka hodnocení systému testery.

4.4.2.2 Analýza hodnocení

Analýzu hodnocení ukazuje tabulka TAB2

	Otázka 1	***Otázka 2***	***Otázka 3***	***Otázka 4***	***Otázka 5***
Průměrné hodnocení	1,5 (A-)	2,65 (C+)	1,625 (B+)	2,125 (B)	2 (B)

Z analýzy hodnocení vyplývá, že systém navzdory své jednoduchosti dopadl docela dobře. Oproti prototypu, který byl testován jen dvěma testery, se hodně zlepšilo zejména uživatelské rozhraní systému, zvláště v kritických procesech jako je zadávání otázek.

4.5 Možnosti dalšího výzkumu

V projektu jsou velké možnosti pro další výzkum a vývoj. Dobrým směrem vývoje může být vypracování pokročilého definování strategie procházení testy, které by uspokojily profesionální náročnější školitele tak, aby test byl inteligentnější. Stávající systém je navržen spíše pro běžné učitele. Druhým zajímavým směrem vývoje je výzkum možností automatického zpracování výrazů odvozených z hledaných slov pomocí gramatických jevů českého jazyka. Systém lze také významně rozšířit o moduly zaměřené na vysvětlování a probírání látky. Určité možnosti dalšího vývoje jsou ve zlepšování uživatelského prostředí, zvláště doplnění pokročilé líbivé grafiky.

4.6 Závěr

Systém automatického vyhodnocování e-learningových testů zvládá vyhodnocování víceslovních odpovědí studenta při online prováděných testech. Projekt je využitelný v oblasti elearningu a počítačem řízeného vzdělávání. Systém používá jednoduché principy při průběžném plánování průchodu testem studenta, které mají za cíl simulovat inteligentní zkoušení od živého školitele. Program projektu má v současné době přibližně 5500 řádků kódu psaných autorem.

Podle testů uživatelů systém odpovídá zadání pro které byl navržen a implementován. Oproti prototypu, který byl vytvořen v kurzu Semestrální projekt, došlo k výraznému zjednodušení a zlepšení uživatelského rozhraní. Dalším rozšířením je zavedení nutnosti stanovit strategie pro procházení jednotlivých kategorií otázek v testu. V původním prototypu se test generoval celý na jeho začátku. V nynějším projektu se test prochází adaptivně podle toho, jak dobře student odpovídá na jednotlivé otázky.

Přínosem autora je implementace systému, který umí vyhodnocovat víceslovné odpovědi a adaptivně přizpůsobuje způsob testování studenta. Systém má dobře zvládnuté uživatelské rozhraní a je snadno rozšiřitelný o další možné funkce. Těmito rozšířeními by mohly být moduly pro vysvětlování učiva či běžné výběrové testy.

Během řešení projektu bylo zjištěno, že uživatelé dávají přednost jednoduchým modelům inteligentního chování, které jsou intuitivně pochopitelné a zvládnutelné. Pokročilé modely, založené na detailnějším modelování studenta během testu, připadaly uživatelům nesrozumitelné a neobešly se bez dlouhého vysvětlování. Proto bylo definování strategie průchodu testem maximálně zjednodušeno a stejně optimalizováno bylo i zadávání otázek a odpovědí, ze kterých byl student zkoušen.

Tento diplomový projekt nemá vazby na jiné právě dokončené projekty.

Literatura

Použitá literatura

[1] Kontis E-Learning, AICC, http://www.e-learn.cz/uvod_standardy_aicc.asp?menu=elearning&submenu=standardy&subsubmenu=aicc

[2] Wikipedie, pojem SCORM, http://cs.wikipedia.org/wiki/SCORM

[3] Kontis E-learning, SCORM, http://www.e-learn.cz/uvod_standardy_scorm.asp?menu=elearning&submenu=standardy&subsubmenu=scorm

[4] Reva Freedman, What is an Intelligent Tutoring System?, http://www.cs.niu.edu/~freedman/papers/link2000.pdf

[5] H. Chad Lane, Intelligent tutoring systems: Prospects for Practice and Effective Learning, http://people.ict.usc.edu/~lane/papers/ITSProspectsLane-Aug06.pdf

[6] Tutoring system SYSPROS a POINTRA, http://wwwbruegge.in.tum.de/projects/intusys/intusys_eng.html

[7] CIRCSM-Tutor Project, http://www.cs.iit.edu/~circsim/

[8] ANDES: An intelligent tutoring system for physics, http://w5.cs.uni-sb.de/website_old/UM97/gertner.html

[9] ITAAM - Intelligent Tutoring Agents in the Area of the Medicine, http://cctc.di.uminho.pt/projects/itaam-intelligent-tutoring-agents-in-the-area-of-the-medicine/

[10] ABITS: An Agent Based Intelligent tutoring System for Distance Learning, http://www.capuano.biz/Papers/ITS_2000.pdf

[11] Cognitive Assistant that Learns and Organizes (CALO), http://godel.stanford.edu/twiki/bin/view/Public/SemlabProjects

[12] The ICICLE Project: An Intelligent Written English tutoring System for Deaf Students, http://www.eecis.udel.edu/research/icicle/

[13] ActiveMath: An Intelligent Tutoring System for Mathematics, http://www.activemath.org/pubs/Melisetal-Activemath-ICAISC-2004.pdf

[14] Java Reference API, http://java.sun.com/javase/6/docs/api/

[15] Hall, Marty. Java Servlety a stránky JSP, Praha: Neocortex spol. s.r.o., 2001, ISBN 80-86330-06-0

Použité zdroje

[z1] Java SE 1.6 - zdroj: http://java.sun.com/javase/downloads/index.jsp

[z2] NetBeans 5.5 - zdroj: www.NetBeans.org

[z3] MySQL - zdroj: http://dev.mysql.com/downloads/

[z4] MySQL GUI TOOLS - zdroj: http://dev.mysql.com/downloads/gui-tools/5.0.html

[z5] XHTML 1.0 - zdroj: http://www.w3.org/TR/xhtml1/

[z6] CSS - zdroj: http://www.w3.org/Style/CSS/

[z7] JavaScript - zdroj: http://www.w3schools.com/js/js_obj_htmldom.asp

Seznam příloh

Příloha 1: Text diplomové práce (pdf), CD

Příloha 2: Zdrojové kódy aplikace, CD

Příloha 3: Vypracované testy, CD

Příloha 4: Uživatelské příručky, CD