

Slovenská technická univerzita v Bratislave
FAKULTA INFORMATIKY A INFORMAČNÝCH
TECHNOLÓGIÍ

Tvorba softvérového systému v tíme
Projektová dokumentácia
Tvorba testov s využitím L^AT_EXu



Vedúci projektu:

Ing. Valentino Vranič, PhD.

Členovia tímu:

Bc. Michal Koščák (SI)

Bc. Radoslav Menkyna (SI)

Bc. Martin Michálek (SI)

Bc. Stanislav Ochotnický (SI)

Bc. Pavel Paroulek (SI)

Kontakt: tsst@lists.kmit.sk

Dátum odovzdania: 15. november 2007

Zadanie

Súčasťou takmer každého univerzitného predmetu sú testy ako jeden zo spôsobov overenia vedomostí študentov. Tieto testy často obsahujú otázky s ponúknutými odpoveďami, na ktoré sa odpovedá výberom. Sú dva typy otázok s ponúknutými odpoveďami: otázky s jednou správnou odpoveďou a otázky s viacerými správnymi odpoveďami. Bodovanie závisí od typu otázky a od existencie trestných bodov za zlú odpoveď.

Odhladnuc od zložitosti tvorby samotných otázok, náročnou úlohou pri tvorbe testov s ponúknutými odpoveďami, ktorá by sa dala automatizovať, je preusporadúvanie otázok a odpovedí na ne za účelom obmedzenia možností odpisovania pri testovaní väčšieho počtu študentov súčasne. Pri preusporadúvaní otázok a odpovedí na ne vznikajú rôzne kľúče pre hodnotenie.

Ak prekladáme existenciu bázy otázok, testy by bolo možné zostavovať automaticky z náhodne vybraných otázok. Otázky pritom majú rozdielnú hodnotu a patria do rôznych okruhov, a väčšinou je potrebné, aby v teste boli zastúpené rôzne okruhy v stanovenom pomere (vzhľadom na hodnotu otázok).

Testy s ponúknutými odpoveďami je najlepšie realizovať na počítačoch. Niekedy to však nie je možné (napr. pre veľký počet študentov), takže zostáva potreba vytlačenia otázok. Za týmto účelom je vhodné použiť \LaTeX . \LaTeX umožňuje sadzbu dokumentov, ktoré vyzerajú profesionálne a zároveň v mnohom poskytuje flexibilitu programovacieho jazyka potrebnú pre zostavenie testu.

Analyzujte problematiku tvorby testov, ktoré obsahujú otázky s ponúknutými odpoveďami. Navrhnite systém na tvorbu takýchto testov, ktorý pomôže pri načrtnutých problémoch. Umožnite generovanie testu v \LaTeX u. Navrhnutý systém implementujte.

Obsah

1	Úvod	7
1.1	Prehľad dokumentu	7
1.2	Slovník pojmov a skratiek	7
1.3	Použitá notácia	8
2	Analýza problémovej oblasti	11
2.1	TestGen	11
2.2	Moodle	13
2.3	Analýza existujúcich testov	14
3	Špecifikácia požiadaviek a analýza systému	17
3.1	Špecifikácia systému	17
3.2	Dekompozícia systému	18
3.3	Otázky a kategórie	18
3.4	Obrázky	23
3.5	Testy	25
3.6	Export do Moodle a L ^A T _E Xu	28
3.7	Export a import databázy	30
4	Návrh systému	35
4.1	Výber technológií	35
4.2	Architektúra systému Genex	38
4.3	Logický model	39
4.4	Fyzický model	43
4.5	Akceptačné testy	45

Kapitola 1

Úvod

Tento dokument vznikol na základe zadania na predmete Tvorba softvérového systému v tíme. Predstavuje projektovú dokumentáciu softvérového systému na podporu tvorby testov v \LaTeX .

1.1 Prehľad dokumentu

Nasledujúci dokument sa skladá z troch hlavných častí: Analýza problémovej oblasti (kapitola 2), Špecifikácia požiadaviek a analýza systému (kapitola 3) a Návrh systému (kapitola 4).

Analýza problémovej oblasti zachytáva pohľad na doménu systému. Zameriava sa najmä na analýzu už existujúceho systému TestGen (časť 2.1). Ďalej obsahuje analýzu systému Moodle (časť 2.2), do ktorého bude umožnený export otázok a v neposlednej rade aj analýzu existujúcich testov vo formáte \LaTeX , ktoré bude potrebné do systému importovať (časť 2.3).

Špecifikácia požiadaviek a analýza systému (kapitola 3) zachytáva požiadavky na vytváraný systém. Systém bol najskôr rozdelený na logické časti (časť 3.2), ktoré sú následne podrobnejšie špecifikované jednotlivými prípadmi použitia alebo inými diagramami. (časti 3.3 - 3.7).

Návrh riešenia (kapitola 4) predstavuje hrubý návrh systému. Zaoberá sa zdôvodnením výberu použitých technológií (časť 4.1), architektúrou systému (časť 4.2), rozpracováva logický a fyzický model systému (časti 4.3 a 4.4). Záver kapitoly obsahuje návrh akceptačných testov (časť 4.5).

1.2 Slovník pojmov a skratiek

ACT - Activity (Diagram aktivít)

analyzátor - interpret \LaTeX kódu

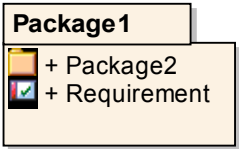
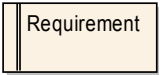
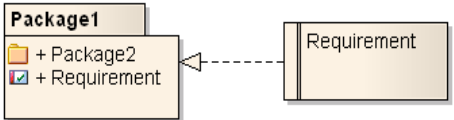
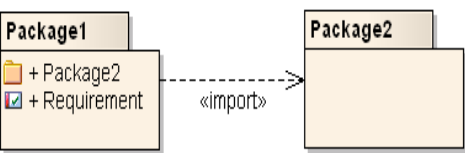
Base64 - Dátový formát zobrazujúci binárne dáta pomocou ASCII znakov

- Bytecode** - binárny zápis programu produkovaný Java kompilátorom
- HTML** - Hypertext Markup Language -značkový jazyk určený na vytváranie internetových stránok
- Java Virtual Machine** - interpret pre bytecode
- L^AT_EX** - typografický nástroj na sádzanie dokumentov
- MVC** - Model View Controller - softvérová architektúra, ktorá rozdeľuje systém na tri nezávislé komponenty. Dátový model, používateľské rozhranie a riadiacu logiku.
- SD** - Sequence Diagram (Sekvenčný diagram)
- Standalone aplikácia** - v tomto prípade aplikácia nezávislá od prostredia a iných aplikácií, vyžadujúca len Java interpret
- STM** - State Machine (Stavový diagram)
- UC** - Use Case (Prípád použitia)
- UML** - Unified Modeling Language -grafický jazyk na vizualizáciu, špecifikáciu, navrhovanie a dokumentáciu systémov
- XML** - eXtensible Markup Language -jazyk pre uchovávanie štruktúrovaných dát
- ZIP** - súborový formát pre kompresiu a archiváciu dát

1.3 Použitá notácia

Prevažná väčšina diagramov použitých v rámci dokumentu vychádza z jazyka UML a jeho štandardov. Výnimkou je diagram požiadaviek (Requirement diagram) (obr. 3.1), použitý na znázornenie dekompozície systému na základe požiadaviek (časť 3.2). V nasledujúcej tabuľke bude opísaná notácia diagramu požiadaviek.

Tabuľka 1.1: Notácia diagramu požiadaviek

 <p>Package1 + Package2 + Requirement</p>	Package (balík) reprezentuje časť systému
 <p>Requirement</p>	Requirement (požiadavka) predstavuje funkcionálnu požiadavku
 <p>Package1 + Package2 + Requirement</p> <p>Requirement</p>	Vzťah "realize" hovorí, ktorá časť systému zabezpečuje danú funkcionálnu požiadavku
 <p>Package1 + Package2 + Requirement</p> <p>Package2</p>	Vzťah <<import>> hovorí, že jeden balík je súčasťou iného balíka

Kapitola 2

Analýza problémovej oblasti

Táto časť dokumentu sa venuje analýze systému Testgen, Moodle a analýze existujúcich \LaTeX testov. Väčšia časť funkcionality požadovanej v zadaní projektu už bola implementovaná v systéme TestGen. Tento systém bol teda pre náš systém dobrým východiskom, preto bola potrebná jeho dôkladná analýza (časť 2.1). Medzi požiadavkami na náš systém bola aj požiadavka na preskúmanie možnosti prepojenia nášho systému so systémom Moodle, ktorý sa rýchlo presadzuje v akademickej sfére. Analýza systému Moodle je preto zameraná na práve túto požiadavku (časť 2.2). Ďalšia z požiadaviek na náš systém predstavovala možnosť importu existujúcich testov vo formáte \LaTeX . Tieto testy boli z tohto dôvodu analyzované (časť 2.3).

2.1 TestGen

Systém TestGen je jednoduchá aplikácia na podporu tvorby testov s využitím \LaTeX u. Navrhovaný systém a systém TestGen vychádzajú z rovnakého zadania. Je pre nás teda zaujímavé podrobnejšie sa zaoberať analýzou systému TestGen. Pri analýze sa zameriame na časti, ktoré sú zaujímavé najmä z pohľadu znovupoužitia. V prvej časti rozoberieme špecifikáciu, analýzu a návrh systému TestGen. Budú využité najmä poznatky získané z dokumentácie systému. V druhej časti bude popísané výsledné riešenie a jeho vlastnosti a konkrétna realizácia návrhu.

2.1.1 Analýza, špecifikácia, návrh

Pri tvorbe systému TestGen bola pomerne rozsiahla časť dokumentácie venovaná analýze existujúcich riešení, zameraná najmä na identifikovanie rôznych typov otázok. Na jej základe boli do systému TestGen vybrané nasledovné typy otázok:

Single choice - otázky s viacerými možnosťami výberu a jednou správnou odpoveďou. Hodnotí sa 0 alebo n bodmi.

Multi choice - otázky s viacerými možnosťami výberu a viac správnymi odpoveďami. Hodnotí sa 0 až n bodmi.

True-False - otázky vo forme výroku, ktorého správnosť je treba určiť. Hodnotí sa 0 alebo n bodmi.

Rank in order - otázky, kde je uvedených viacero možností, ktoré je potrebné správne zoradiť. Hodnotí sa 0 až n bodmi.

Sore finger - otázky s uvedenými viacerými možnosťami resp. výrokmi, z ktorých je potrebné niektoré vyradiť (nehodiace sa, nesprávne). Hodnotí sa 0 až n bodmi.

Fill in - text s vynechaným prázdny miestom alebo miestami. Nie sú poskytnuté odpovede, je potrebné doplniť na prázdne miesta správne pojmy. Hodnotí sa 0 až n bodmi, samostatne vzhľadom na každé doplnené slovo.

Numeric fill in - analogicky ako v predchádzajúcom bode, ale dopĺňajú sa číselné hodnoty.

Sentence - otázky, kde je potrebné odpovedať pojmom alebo krátkou vetou. Nie sú poskytnuté odpovede. Hodnotí sa 0 až n bodmi.

Essay - zadaním je nejaký problém, ktorý je potrebné vyriešiť. Odpoveďou je súvislejší text, ktorý môže obsahovať diagramy, vzorce a podobne. Hodnotí sa 0 až n bodmi.

Ďalšou zaujímavou časťou je definovanie kategorizácie otázok. TestGen má definovanú hierarchickú štruktúru kategórií. Otázka sa môže nachádzať na ľubovoľnej úrovni hierarchie, pričom jedna otázka môže patriť iba do jednej kategórie. Určitú formu kategorizácie otázok poskytuje TestGen aj prostredníctvom kľúčových slov. Každý otázke možno priradiť ľubovoľný počet kľúčových slov. Na rozdiel od kategórií sú kľúčové slová viazané na používateľa systému. Každý používateľ má teda svoju sadu kľúčových slov, množina kategórií je však zdieľaná všetkými používateľmi.

Samotné generovanie testov je rozdelené do dvoch častí: definovanie kritérií a generovanie testu s možnosťou následnej úpravy. Tieto dva základné body sú realizované prostredníctvom nasledovných krokov:

1. stanovenie tématického okruhu otázok definovaním kategórií, prípadne určením kľúčových slov,
2. stanovenie bodovej hodnoty testu,
3. definovanie percentuálneho zastúpenia typov a obtiažnosti otázok,
4. vygenerovanie prvej verzie testu systémom,

5. kontrola otázok používateľom a ich prípadná úprava,
6. vygenerovanie definovaného počtu permutácií testov systémom,
7. ponúknutie možnosti vygenerovania ďalších testov s danými kritériami a obmedzeniami, ktoré sú stanovené prvými tromi bodmi.

Bezpochyby najlepšie spracovanou časťou návrhu je logický model údajov. Plne uspokojuje požiadavky vyplývajúce zo špecifikácie a analýzy.

2.1.2 Riešenie

Z funkcionálneho hľadiska spĺňa systém TestGen všetky požiadavky dané špecifikáciou. Pri jeho používaní však boli zistené nasledovné nedostatky:

- Použitie databázy MySQL sa na prvý pohľad zdalo byť celkom dobrým riešením. Jeho nevýhodou je, že inštalácia a nastavenie databázy je nezávislé od samotného programu. Inštalácia MySQL Server 5.1 potrebná na spustenie TestGen pri tom zaberá takmer 110MB pamäti na pevnom disku. Ešte horšie sa ukázalo, že na spustenie programu je nutné poznať aj prostredie MySQL, čo nie je, ako sme sa mali možnosť presvedčiť, celkom triviálna úloha.
- Počas práce so systémom TestGen sme narazili na niekoľko závažných chýb v používateľskom rozhraní. Ako príklady môžeme uviesť:
 - Zámena funkcionality niektorých tlačidiel.
 - Pri odstránení otázky z testu (túto operáciu je možné vrátiť) sa systém opýtal či ju naozaj chceme odstrániť, naopak pri pokuse o odstránenie obrázka z databázy (natrvalo) systém bez dodatočného opýtania obrázkov odstránil.

2.1.3 Zhodnotenie

Z analýzy systému TestGen vyplynulo niekoľko zaujímavých poznatkov. Na jednej strane sú to nápady ktorými sa môžeme inšpirovať, na strane druhej chyby, ktorých sa treba vyvarovať.

2.2 Moodle

Systém Moodle je otvorený softvérový systém na podporu vzdelávania. Je pomerne rozsiahly a ponúka širokú škálu nástrojov. Z nášho pohľadu je zaujímavá najmä jeho funkcionality týkajúca sa práce s testami. Moodle poskytuje plnú funkcionality pre tvorbu, úpravu, zadávanie a vyhodnocovanie testov. V rámci tejto funkcionality je zahrnutá aj správa otázok, ich tvorba a úprava. Tieto testy sú primárne určené pre elektronické vypracovanie priamo v systéme.

Aby sme rozšírili funkcionality nášho systému, skúmali sme možnosť jeho prepojenia zo systémom Moodle. Zistili sme, že systém Moodle podporuje import otázok, z ktorých môže byť neskôr v tomto systéme vytvorený test. Z rôznych formátov, ktoré systém Moodle podporuje je najprístupnejší formát Moodle xml. Tento formát má jasne špecifikovanú štruktúru a syntax, preto je vhodný pre export otázok z nášho systému. Popri otázkach sa do systému Moodle dajú importovať taktiež obrázky, ktoré sú s niektorými otázkami prepojené. Obrázky však musia byť pripojené vo formáte Base64.

Systém Moodle podporuje všetky kategórie otázok, ktoré podporuje náš systém. Systém však štandardne vyžaduje pri otázkach s viacerými možnosťami odpovedí, zadávanie váhy jednotlivých možností. Toto systému následne umožňuje flexibilnejšie hodnotenie odpovedí. Túto funkcionality však náš systém štandardne nepodporuje, pretože je primárne určený pre tvorbu testov určených pre tlač. Preto budú všetky otázky exportované z rovnakou váhou odpovedí. Systém taktiež umožňuje vkladanie pomocných alebo vysvetľujúcich textov k otázkam, ktoré sa zobrazia pri nesprávnom vypracovaní otázky. Táto funkcionality zo spomínaných dôvodov nebude taktiež podporovaná.

2.3 Analýza existujúcich testov

Jednou z požiadaviek na systém bola požiadavka na import otázok z existujúcich testov vo formáte L^AT_EX. Tieto testy používajú rovnaké makrá. Obsahujú otázky, ku ktorým boli poskytnuté možnosti správnej odpovede, z ktorých práve jedna bola správna. Príklad otázky z poskytnutých testov:

```
\bq{1} Dá sa v Jave urobiť \emph{inštancia} rozhrania?
\eq
\begin{answers}
\item nie \label{q\theqnum:a}
\item áno
\item áno, ale len ak neobsahuje metódy
\item áno, ale nebudú sa dať zavolať jeho metódy
\item áno, ale len ak obsahuje výlučne statické prvky
\end{answers}
```

Úlohou systému bude z daných testov získať otázky a k nim prislúchajúce možnosti odpovedí. Takisto dôležité bude získať informáciu o správnosti danej odpovede. Keďže systém bude generovať testy vo formáte L^AT_EX, nebude potrebné odstraňovať použité príkazy.

Začiatok i koniec otázky je ľahko identifikovateľný, rovnako je to v prípade odpovedí. Každá z možností odpovede je uvedená na samostatnom riadku a začína príkazom

```
\item
```

teda tiež sa dá pomerne ľahko identifikovať. Správnosť odpovede je označená bezprostredne za textom odpovede príkazom

`\label{q\thecnum:a}`

Niektoré otázky obsahujú autorom definované makrá, ktoré bude potrebné buď rozbaľiť alebo uložiť do systému a pri generovaní testu vložiť do výsledného súboru. Po zvážení výhod a nevýhod sme sa rozhodli použiť skriptovací jazyk (aký, to sa rozhodne v neskorších fázach projektu) na “vytiahnutie” jednotlivých otázok z testov a ich následné uloženie do formátu, vhodného resp. zrozumiteľného pre náš systém.

Kapitola 3

Špecifikácia požiadaviek a analýza systému

Táto kapitola zachytáva proces špecifikácie požiadaviek a ich následnej analýzy. Najskôr boli zostavené požiadavky na systém vychádzajúc zo zadania projektu a projektových stretnutí (časť 3.1). Systém bol následne rozčlenený do menších celkov na základe funkcionality (časť 3.2). Takto získané celky mohli byť následne nezávisle analyzované (časti 3.3 - 3.7). Analýza je z väčšej časti je tvorená prípadmi použitia. V niektorých prípadoch v sekvenčných diagramoch alebo diagramoch aktivít. Sekvenčné diagramy boli použité, ak sa podarilo identifikovať podsystem, ktorý bude danú funkcionality vykonávať. Diagramy aktivít v prípade, ak takýto podsystem doposiaľ nebol identifikovaný. V diagramoch sa často používa účastník Učiteľ. Ten reprezentuje pedagogického pracovníka, ktorý používa tento systém.

3.1 Špecifikácia systému

Zo zadania projektu a projektových stretnutí bol vytvorený nasledovný zoznam požiadaviek na funkcionality systému:

Najvyššia priorita

- automatické generovanie testov,
- spravodlivé generovanie otázok čo sa týka náročnosti a bodov,
- kategorizácia otázok,
- vkladanie obrazových príloh,
- možnosť manuálnej opravy vygenerovaného testu v systéme, výstup .tex pre opravy mimo systém/archiváciu,
- variabilné spôsoby testovania,
- import a export bázy otázok (import z .tex, export do Moodle),
- zvýraznenie syntaxe.

Stredná priorita

- vyjadrovanie vzťahov medzi otázkami,
- generovanie výsledkových hárkov,
- neviazanosť na konkrétny predmet alebo odbor,

- zabezpečenie prístupu k otázkam iba oprávneným osobám (bezpečnosť databázy),
- parametrizácia otázok,

Najnižšia priorita

- kontrola pravopisu,
- zasadací poriadok,
- rôzne formy výstupov,
- optimalizácia pre väčší počet otázok.

Okrem už spomenutých funkcionálnych požiadaviek rozdelených podľa priorít boli identifikované aj nasledovné požiadavky:

- grafické jednopoužívateľské rozhranie,
- možnosť rozšírenia na viacpoužívateľské rozhranie.

3.2 Dekompozícia systému

Počas špecifikácie systému vznikla nutnosť rozdeliť systém do menších celkov na základe funkcionality tak, aby bolo možné pracovať na analýze týchto častí nezávisle. Rozdelenie systému je možné vidieť na obr. 3.1. Medzi niektorými z týchto častí existujú silné väzby, preto ich nemožno pokladať za moduly. Jedná sa o logické celky systému. Tieto časti budú v nasledujúcich častiach dokumentu špecifikované a analyzované. Na systém sa kladie viacero požiadaviek, tieto však boli identifikované ako dôležité, alebo kľúčové pre implementáciu systému.

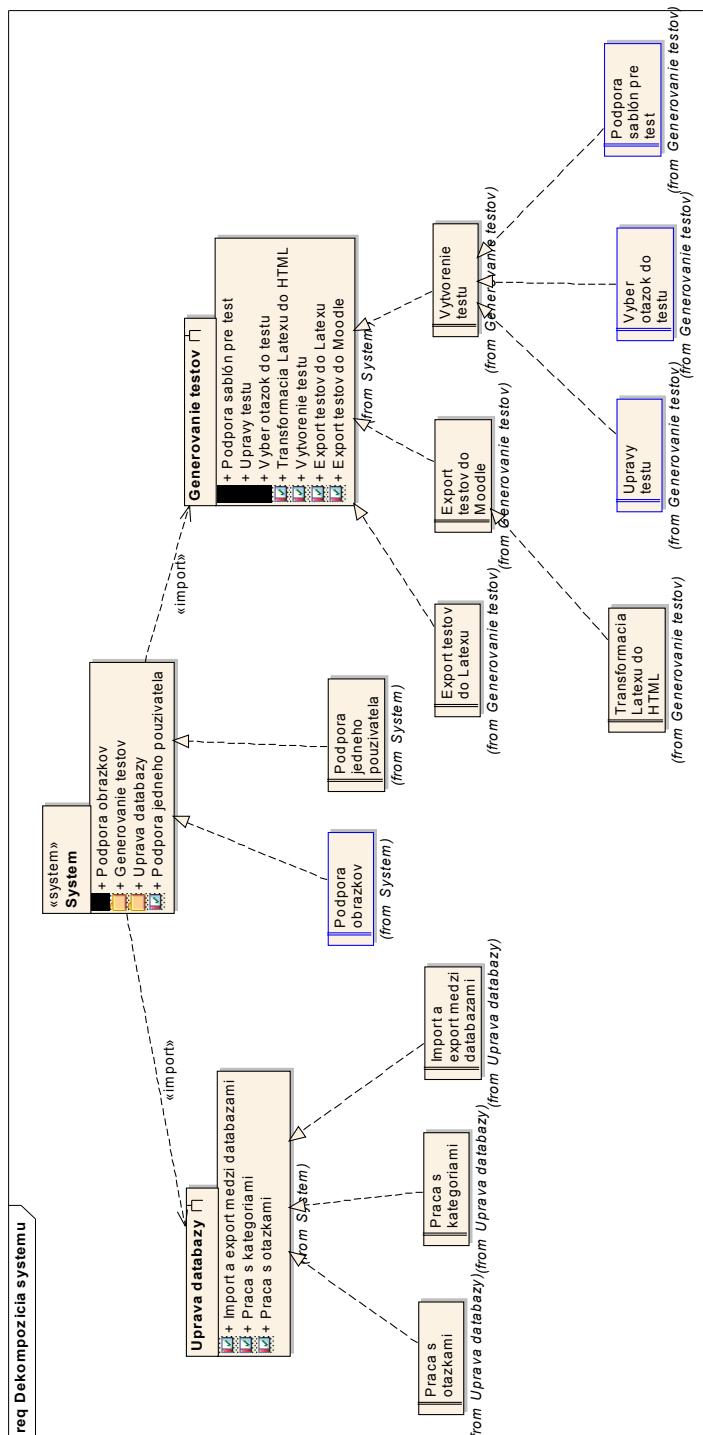
3.3 Otázky a kategórie

Úlohou systému v oblasti zaradenia príbuzných otázok do spoločných okruhov je vytvoriť takú hierarchiu, ktorá zabezpečí efektívne vyhľadávanie otázok pri tvorbe testov a umožní jednoduchú, prehľadnú orientáciu. Hierarchická štruktúra kategórií nie je obmedzená, otázka sa preto môže nachádzať v ľubovoľnej úrovni. Túto hierarchickú štruktúru si definuje samotný používateľ systému podľa vlastných potrieb. Ku každej otázke je možné vytvoriť niekoľko kľúčových slov, ktoré ju charakterizujú. Pomocou týchto kľúčových slov bude môcť používateľ ľahšie vyhľadávať otázky. Systém bude podporovať nasledujúce typy otázok:

- Single choice - otázky s viacerými možnosťami výberu a jednou správnu odpoveďou.
- Multi choice - otázky s viacerými možnosťami výberu a viac správnymi odpoveďami.
- True-False - otázky vo forme výroku, ktorého správnosť je treba určiť.
- Fill in - text s vynechaným prázdny miestom alebo miestami. Nie sú poskytnuté odpovede, je potrebné doplniť na prázdne miesta správne pojmy.
- Essay - zadaním je nejaký problém, ktorý je potrebné vyriešiť. Odpoveďou je súvislejší text.

Nasledujúca špecifikácia opisuje správu otázok a kategórií z pohľadu používateľa. Je rozdelená do 3 častí:

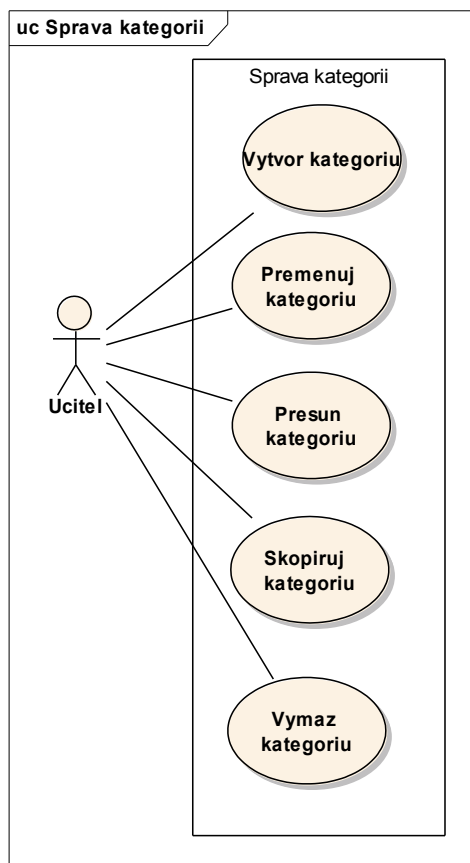
- správa kategórií,
- správa otázok,
- správa odpovedí.



Obrázok 3.1: Dekompozícia systému z pohľadu funkcionálnych požiadaviek. Rozdelenie je zobrazené pomocou diagramu požiadaviek.

3.3.1 Správa kategórií

Kategórie slúžia na triedenie príbuzných otázok. Toto triedenie zjednoduší vytváranie, vyhľadávanie a celkovú správu otázok v systéme. Každá kategória môže obsahovať rôzny počet podkategórií a otázok. Použitie kategórií môžeme prirovnať použitiu priečinkov v operačných systémoch.



Obrázok 3.2: Správa kategórií

UC-1 Vytvor kategóriu

Vytvorenie kategórie je jednou zo základných požadovaných funkcií systému. Kategóriu bude možné vytvoriť v ľubovoľnej existujúcej kategórii. Pri vytváraní bude potrebné zadať iba meno novej kategórie.

UC-2 Premenuj kategóriu

Premenovanie kategórie znamená jednoducho zmenu názvu vybranej kategórie.

UC-3 Presuň kategóriu

Presun kategórie znamená výber kategórie a jej následné vloženie do inej kategórie, vrátane všetkých jej položiek (podkategórií a otázok).

UC-4 Skopíruj kategóriu

Skopírovanie kategórie je podobné presunu kategórie. Najprv sa vyberie kategória na skopírovanie, následne sa určí cieľová kategória, v ktorej sa vytvorí kópia zdrojovej kategórie vrátane všetkých jej podkategórií i otázok.

UC-5 Vymaž kategóriu

Vymazanie kategórie znamená vymazanie vybranej kategórie spolu so všetkými podkategóriami i otázkami v nej. Pred samotným vymazaním si ešte systém vyžiada potvrdenie tejto operácie.

3.3.2 Správa otázok

Správa otázok zahŕňa tvorbu, úpravy, kopírovanie, presun, odoberanie a obnovu otázok používateľom. Ak sme kategóriu prirovnali k priečinku, tak otázku môžeme prirovať k súborom, ktoré sa nachádzajú v priečinkoch.

UC-6 Vytvor otázku

Vytvorenie otázky bude stále spojené s vybranou kategóriou. Vytvorená otázka bude automaticky zaradená do vybranej kategórie. Pri vytváraní bude možné zadať všetky potrebné parametre, vrátane možných odpovedí a popríklad aj obrázkov.

UC-7 Uprav otázku

Úprava otázky znamená zmenu jedného alebo viacerých parametrov vybranej otázky.

UC-8 Skopíruj otázku

Skopírovanie otázky znamená vytvorenie kópie zdrojovej otázky v inej kategórii.

UC-9 Presuň otázku do inej kategórie

Presun otázky do inej kategórie znamená zmenu kategórie danej otázky.

UC-10 Vymaž otázku

Vymazanie otázky znamená jej odstránenie z kategórie. Táto otázka sa však kvôli možnosti obnovy nevymaže zo systému celkom, len ju už nebude štandardne v kategórii vypísaná.

UC-11 Obnov otázku

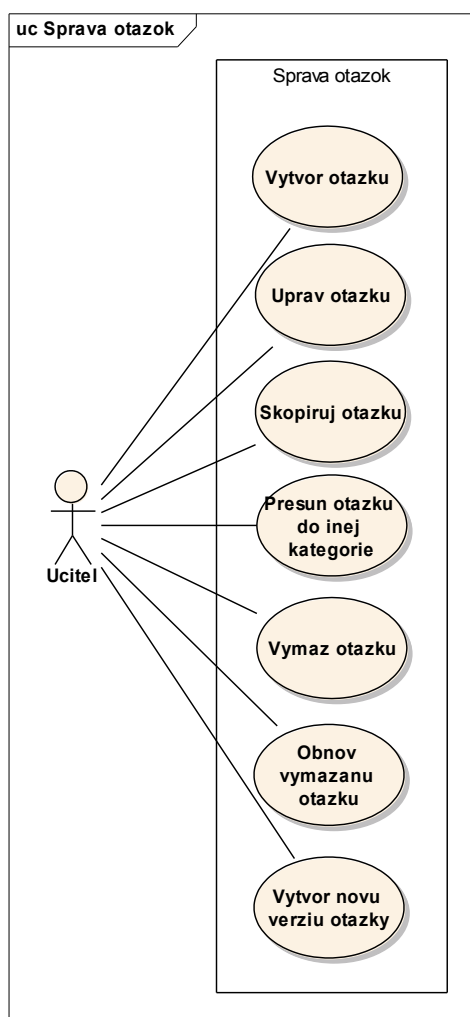
Obnovenie otázky znamená jej vrátenie do jej pôvodnej kategórie, z ktorej bola vymazaná.

UC-12 Vytvor novú verziu otázky

Vytvorenie novej verzie otázky znamená, že sa po úprave vybranej otázky neuložia vykonané zmeny, ale sa táto otázka uloží ako nová (upravená) verzia pôvodnej.

3.3.3 Správa odpovedí

Správa odpovedí znamená vytváranie, úpravu a odoberanie odpovedí konkrétnych otázok.



Obrázok 3.3: Správa otázok

UC-13 Vytvor odpoveď

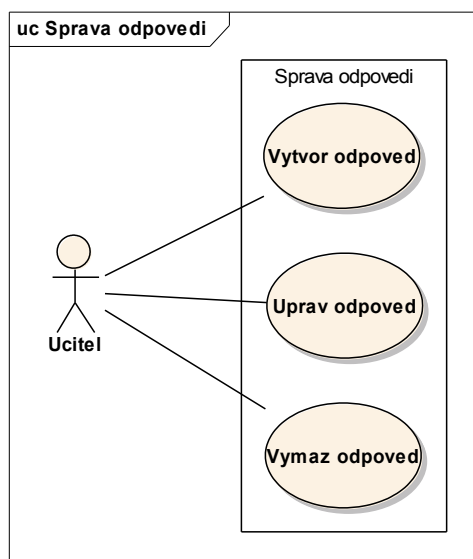
Vytvorenie odpovede znamená, že sa konkrétnej otázke pridá nová odpoveď, pričom sa označí jej správnosť.

UC-14 Uprav odpoveď

Úprava odpovede zahŕňa zmenu jedného alebo viacerých parametrov vybranej odpovede.

UC-15 Vymaž odpoveď

Vymazanie odpovede znamená vymazanie jednej odpovede jednej konkrétnej otázky.



Obrázok 3.4: Správa odpovedí

3.4 Obrázky

Potreba podpory obrázkov v systéme Genex je zrejmá. K niektorým otázkam je vhodné priložiť obrázky ako príklad, alebo ako možnú odpoveď.

Z používateľského hľadiska je potrebné zabezpečiť prácu s obrázkami v systéme na dvoch úrovniach:

- úprava databázy,
- generovanie testov.

Funkcionálne požiadavky pre prácu s obrázkami sa nachádzajú v diagrame prípadov použitia na obr. 3.5.

3.4.1 Úprava databázy

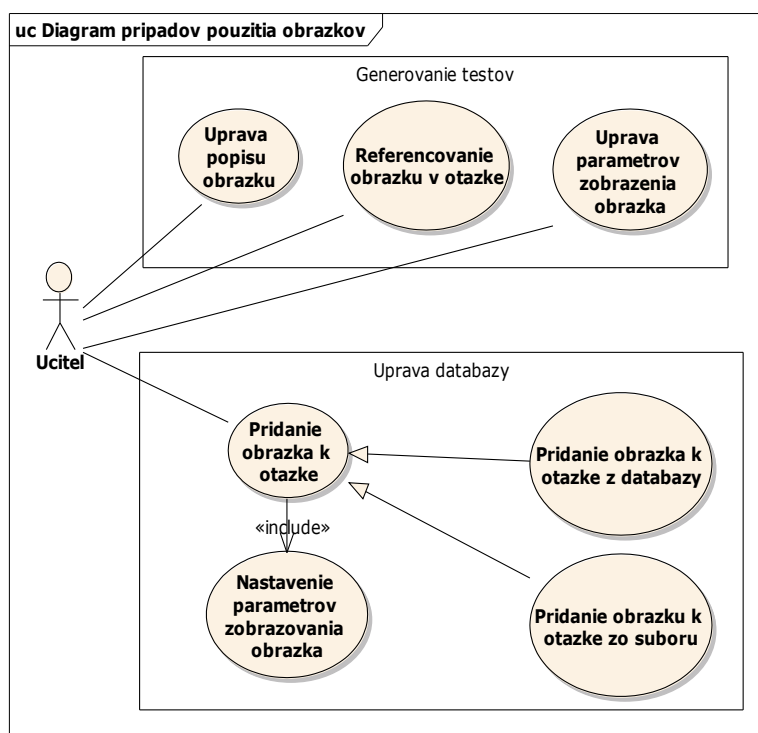
Úprava databázy v kontexte obrázkov predstavuje najmä pridávanie obrázkov k otázkam a modifikáciu rôznych parametrov zobrazovania obrázkov uložených v databáze spolu so samotnými dátami obrázkov.

UC-16 Pridanie obrázka k otázke

Prvým krokom v životnom cykle obrázka v systéme je jeho priradenie k niektorej z otázok. Pridelenie môže prebehnúť dvoma spôsobmi:

- ak bol vybraný nový obrázok zo súborového systému, vloží sa ako nová entita do databázy a priradí sa k vybranej otázke,
- ak bol obrázok vybraný z databázy, iba sa pridá nové prepojenie otázky s obrázkom.

Tento postup zabezpečí, že v systéme sa budú nachádzať iba obrázky súvisiace s niektorou z otázok. Uľahčí to tiež výber otázok z databázy, keďže ich bude možné triediť podľa kategórií podobne ako otázky (pozri 3.3).



Obrázok 3.5: Diagram prípadov použitia obrázkov

UC-17 Pridanie opisu obrázka

V niektorých prípadoch je potrebné zobraziť k jednotlivým obrázkom opis, ktorý môže objasniť obsah obrázka. Opis obrázka môže obsahovať \LaTeX syntax.

UC-18 Nastavenie parametrov zobrazovania obrázka

Systém automaticky nastaví základný spôsob zobrazovania obrázka ako je napríklad výška alebo šírka.

3.4.2 Generovanie testov

Časť generovania testov sa týka prípravy a vytvárania \LaTeX výstupu. Výber obrázkov je výsledkom výberu otázok v teste. V rámci generovania testov bolo identifikovaných viacero funkčných požiadaviek.

UC-19 Referencovanie obrázka v otázke

Možnosť odkázania sa v texte otázky na obrázok je potrebná z viacerých dôvodov. Hlavným dôvodom je umožnenie vkladania viacerých obrázkov k jednej otázke a ich jednoznačnú identifikáciu.

UC-20 Úprava parametrov zobrazenia obrázka

Možnosť ovplyvňovať spôsob zobrazenia môže byť v niektorých prípadoch dôležitá. Ide najmä o veľkosť obrázka, jeho umiestnenie na stránke a podobne.

UC-21 Úprava opisu obrázka

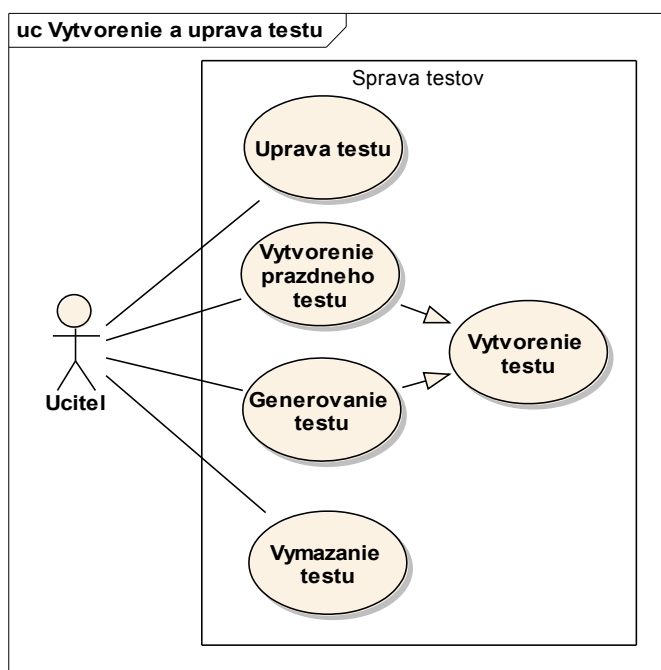
Opis obrázka, ktorý mohol byť zadaný ešte v databáze, je v niektorých prípadoch potrebné upraviť. Učiteľ si zvolí alternatívny opis obrázka, ktorý bude platný iba pre daný test.

UC-22 Export obrázkov do Moodle

Pri exporte otázok do systému Moodle sa spolu s otázkami exportujú aj obrázky.

3.5 Testy

Cieľom tohto modulu je umožniť používateľovi vytvoriť a upravovať test, ktorý môže byť následne použitý v procese výučby (obr. 3.6).



Obrázok 3.6: Správa testov

Prípady použitia

UC-23 Vytvorenie testu

UC-23 predstavuje proces inicializácie nového testu (obr. 3.7). V prvej fáze tohto procesu používateľ zadá meno testu a názov predmetu. Následne sa musí rozhodnúť, akým spôsobom chce test vytvoriť. V podstate existujú dva spôsoby ako vytvoriť test. Prvým je vygenerovanie testu na základe poskytnutej sady parametrov (prípád použitia UC-24) tak, aby potreba dodatočných úprav bola čo najmenšia.

Cieľom je tento proces plne automatizovať a tým minimalizovať úsilie a čas potrebný na vytvorenie testu. Druhý spôsob vytvorenia testu je jeho ručné zostavenie UC-25). Umožňuje získať plnú kontrolu nad zostavením testu na úkor času.

UC-24 Generovanie testu

Generovanie testu je automatický proces, pri ktorom systém na základe vstupných parametrov vyskladá test. Z hľadiska používateľa je v procese generovania zaujímavé najmä zadávanie parametrov, ktoré definujú filter pre výber otázok. Používateľ bude môcť zadať nasledovné parametre:

Kategórie - určujú tematický okruh otázok. Jednotlivé kategórie sú vybraté zo stromu kategórií.

Počet otázok - určuje koľko otázok bude vytvorený test obsahovať. Ak база otázok neobsahuje požadovaný počet otázok z danými vlastnosťami, systém upozorní používateľa zodpovedajúcim chybovým hlásením.

Počet bodov - určuje za koľko bodov test bude. Body pre jednotlivé otázky sú generované na základe váhy otázky, zložitosti testu a počtu otázok v teste. Tieto hodnoty je neskôr možné v teste upraviť.

Typy otázok - určuje aké typy otázok sa v teste budú vyskytovať. Výber z množiny podporovaných typov otázok (časť 3.3).

Zložitosť - určuje zložitosť testu. Pri vyššej zložitosti sa zvyšuje pravdepodobnosť výskytu otázok s vyššími váhami, pričom je týmto otázkam pridelený nižší počet bodov ako pri teste s nižšou zložitosťou.

Používateľ z nich musí zadať aspoň kategórie a počet otázok, z ktorých chce test generovať. Po ukončení generovania sa vygenerovaný test automaticky uloží medzi existujúce testy.

UC-25 Vytvorenie prázdneho testu

Vytvorenie prázdneho testu sa využíva ak chce používateľ celý test vytvoriť ručne, postupným pridávaním otázok. V tomto prípade je systémom vytvorený prázdny test, pričom pridávanie otázok do testu je vlastne úprava prázdneho testu (prípád použitia UC-27). Vytvorený test je automaticky uložený medzi existujúce testy.

UC-26 Odstránenie testu

Pred samotným odstránením musí používateľ test označiť. Následne je možné test vymazať.

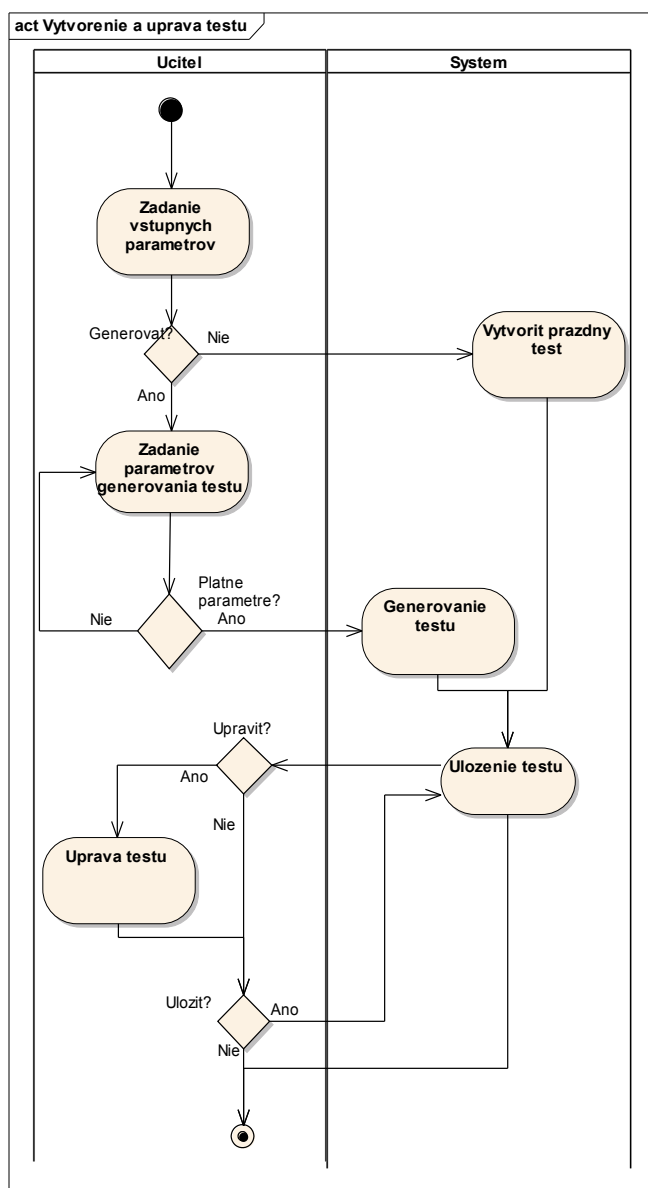
3.5.1 Úprava testu

Po ukončení inicializácie testu, môže používateľ pokračovať druhou fázou, v ktorej môže upraviť test vytvorený v prvej fáze. Úprava testu bude podrobnejšie opísaná prípadom použitia UC-27. Po ukončení úprav si môže používateľ test uložiť (obr. 3.7).

UC-27 Úprava testu

UC-27 reprezentuje proces úpravy testu (obr. 3.8). Predtým ako môžeme upravovať musíme vybrať test, ktorý chceme upraviť. Následne sú používateľovi k dispozícii štyri typy úprav:

- pridanie otázky do testu,



Obrázok 3.7: Vytvorenie a úprava testu

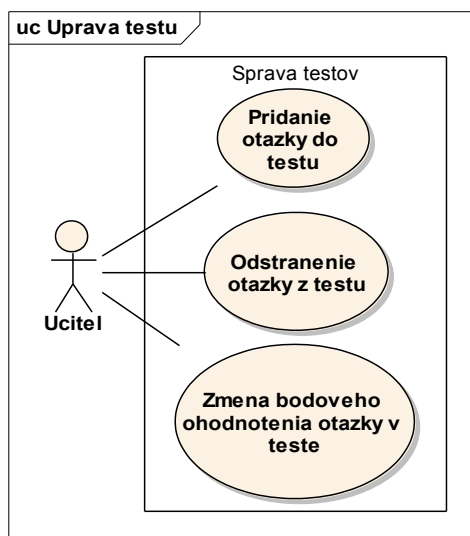
- odobratie otázok z testu,
- zmena bodového ohodnotenia otázok v teste.

UC-28 Pridanie otázky do testu

Používateľ si vyberie otázku pomocou filtra otázok, označí ju a pridá do testu.

UC-29 Odobratie otázok z testu

Používateľ odstráni označenú otázku z testu.



Obrázok 3.8: Diagram prípadov použitia pre úpravu testu

UC-30 Zmena bodového ohodnotenia otázok v teste

Používateľ môže upraviť predvolené bodové ohodnotenie otázky. Zmena bodového ohodnotenia nemá na rozdiel od úpravy otázky vplyv na samotnú otázku v databáze. Bodové ohodnotenie je reprezentované ako ohodnotenie spojenia medzi otázkou a testom (kapitola 4.3).

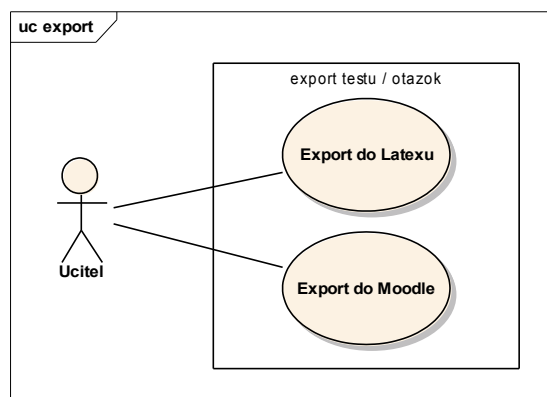
3.6 Export do Moodle a \LaTeX u

Export údajov je jedna zo základných funkcií úspešného softvérového systému. Náš systém okrem exportu celej databázy podporuje aj export testov, alebo zadefinovanej množiny otázok, do vybraných formátov. Systém podporuje priamu konverziu do formátu \LaTeX a do formátu Moodle xml (obr.3.9).

UC-31 Export do \LaTeX u

Prípado použitia zachytáva potrebu používateľa exportovať vygenerovaný test do formátu, z ktorého je následne možná jeho tlač. Počiatočný predpoklad tohto prípadu použitia je existencia zostaveného testu, či už vygenerovaného alebo otvoreného z databázy. Výstupom je súbor, prípadne viac súborov vo formáte \LaTeX . Ak test obsahuje obrázky, výstupom sú aj súbory reprezentujúce tieto obrázky.

Export testov do formátu \LaTeX je jedna zo základných požiadaviek na náš systém. Z formátu \LaTeX je možno jednoducho vytvoriť výstup do formátu PDF, ktorý je vhodný pre následnú tlač vygenerovaných testov. Náš systém podporuje zadávanie otázok a odpovedí priamo vo formáte \LaTeX . Export do formátu \LaTeX podrobnejšie zachytáva diagram prípadov použitia (obr.3.10). Prípady použitia, ktoré sú pre tento proces kľúčové sú UC-32 a UC-33.



Obrázok 3.9: Diagram prípadov použitia - export do L^AT_EXu a Moodle xml

UC-32 Výber šablóny

Proces predstavuje možnosť používateľa vybrať si jednu z prednastavených výsledných rozložení komponentov testu. Vstupom pre tento prípad použitia je vyjadrenie preferencie daného rozloženia testu prostredníctvom používateľa. Výstupom je šablóna dokumentu vo formáte L^AT_EX. Táto šablóna je neskôr modifikovaná v prípadom použitia úprava šablóny UC-33.

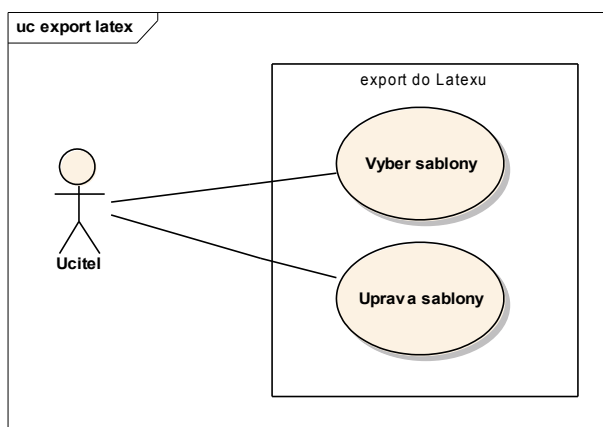
UC-33 Úprava šablóny

Prípad použitia predstavuje proces, pri ktorom sa do šablóny rozloženia komponentov testu dopĺňajú samotné komponenty. Vstupom pre prípad použitia je šablóna rozmiestnenia komponentov. Do tejto šablóny sa vždy automaticky pridávajú aktuálne definície makier, ktoré sa môžu vyskytnúť v texte otázok a odpovedí. Následne sa podľa potreby pridávajú do šablóny ostatné prvky. Vstupom pre prípad použitia sú údaje zadané používateľom alebo údaje prítomné v systéme. Výstupom je modifikácia existujúcej šablóny rozmiesnenia komponentov. Komponenty sú na správne miesto v šablóne umiestňované na základe interných značiek. Ako značky budú použité zakomentované kľúčové slová.

UC-34 Export do Moodle xml

Export do Moodle xml je proces prenosu otázok vygenerovaných, alebo označených v našom systéme do systému Moodle. Vstupom pri tomto prípade použitia je vygenerovaný test alebo kategórie otázok označené v systéme. Výstupom je súbor vo formáte Moodle xml, ktorý obsahuje danú množinu otázok. Tieto otázky môžu byť v systéme Moodle neskôr ľubovoľne modifikované, čiže obsiahnuté v rôznych možných testoch, zostavených v tomto systéme. Do systému Moodle sa exportujú samostatné otázky bez prihliadnutia na formát alebo nastavenia vygenerovaného testu.

Proces, ktorý tento prípad použitia predstavuje je pomerne zložitý, a preto bol podrobnejšie zachytený v sekvenčnom diagrame. Sekvenčný diagram zachytáva identifikované časti systému, ktoré danú funkcionálnu vykonávajú, ako aj čas-

Obrázok 3.10: Diagram prípadov použitia - export do \LaTeX u

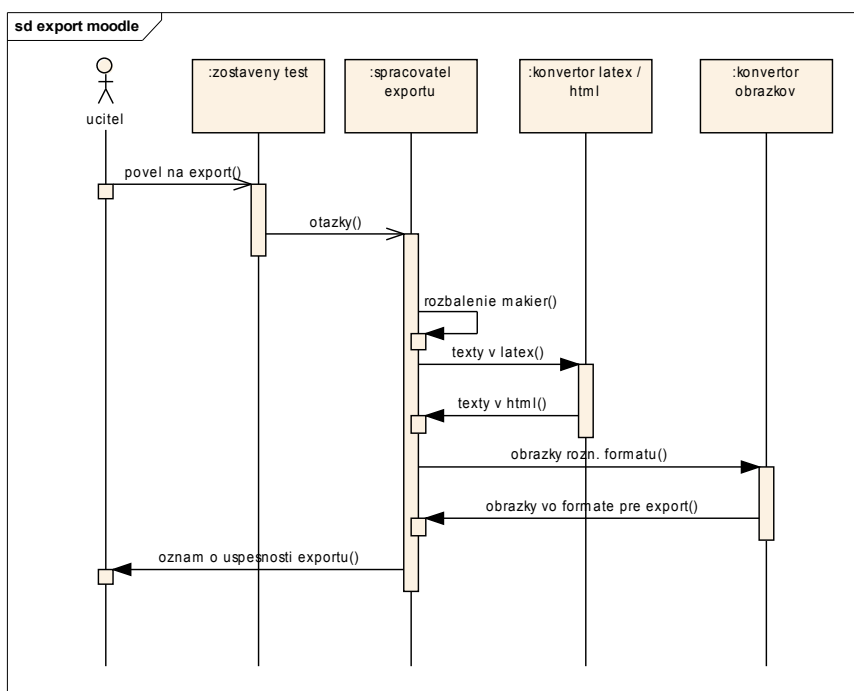
vú súslednosť jednotlivých krokov exportu (obr.3.11). Aby sa zachovalo prípadné formátovanie otázok a odpovedí, otázky musia byť preložené do jazyka HTML, ktorý je podporovaný systémom Moodle. Tento proces vykoná nato určený konvertor. Pred samotnou konverziou však musia byť nahradené používateľom definované makrá štandardnými \LaTeX príkazmi. Pri tejto konverzii \LaTeX / HTML bude podporovaná konečná množina \LaTeX symbolov, ktorú bude môcť používateľ rozšíriť. Následne, ak otázky obsahujú nejaké obrázky, tieto musia byť skonvertované do formátu Base64. O konverziu do tohto formátu sa postará taktiež špeciálny konvertor. Po týchto konverziách systém vytvorí už konečný dokument vo formáte xml, ktorý môže byť následne exportovaný do systému Moodle.

3.7 Export a import databázy

Podsystém na export a import databázy je dôležitou časťou systému. Bude umožňovať jednoduchú výmenu otázok medzi dvoma databázami. Pod pojmom *súbor databázy* budeme rozumieť ZIP archív, ktorý bude obsahovať súbor s \LaTeX makrami a súbor s databázovými údajmi. Importované údaje sa budú ukladať do podkategórie *Import*. Kategória *Import* bude jednou z koreňových kategórií v systéme.

UC-35 Premenovanie importovaných \LaTeX makier

Pri importe údajov zo súboru databázy môže dôjsť ku konfliktu názvov importovaných makier a makier, ktoré má používateľ definované pre otázky v databáze. Prípád použitia umožňuje importovaným makrám rozšíriť názov tak, aby bol pre celý systém unikátny. Názvy makier sa zmenia aj v importovaných otázkach. K premenovaniu dôjde až počas načítavania súboru databázy do systému. K premenovaniu nedôjde v súbore databázy. Ak dôjde k premenovávaniu makier, ich názvy sa zmenia aj v importovaných otázkach.



Obrázok 3.11: Sekvenčný diagram- export do Moodle xml

UC-36 Import údajov zo súboru databázy

Prípado použitia umožňuje skopírovať údaje zo súboru databázy do podkategórie kategórie *Import* a začleniť ich tak do databázy.

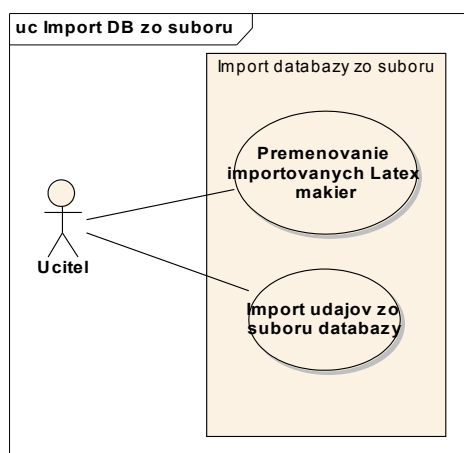
Podsystem slúžiaci na import údajov exportovaných z inej databázy bude tvoriť konzistentný celok. Na obr. 3.12 je zobrazený diagram prípadov použitia importu do databázy. Pri importe otázok systém prekopíruje údaje zo vstupného súboru databázy do podkategórie koreňovskej kategórie *Import*. Táto podkategória bude mať unikátny názov (*import-dátum-číslo*) a bude koreňovou kategóriou pre importovanú stromovú hierarchiu otázok (ďalej len importovaná hierarchia). V tejto podkategórii bude môcť používateľ už ľubovoľne meniť otázky a názvy kategórií. Ďalej bude možné prenášať otázky a kategórie v rámci existujúceho hierarchického systému otázok v databáze.

UC-37 Určenie kategórie na export

Prípado použitia umožňuje používateľovi určiť jednu alebo viac kategórií, ktoré sa majú exportovať.

UC-38 Export údajov z databázy do výstupného súboru databázy

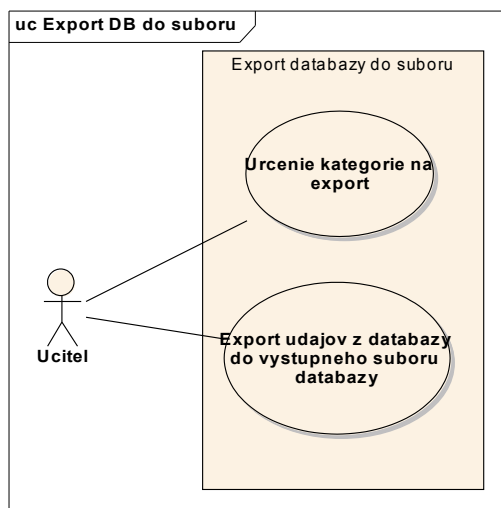
Prípado použitia umožňuje exportovať (uložiť) údaje z kategórií určených na export (UC-37) do vybraného súboru databázy.



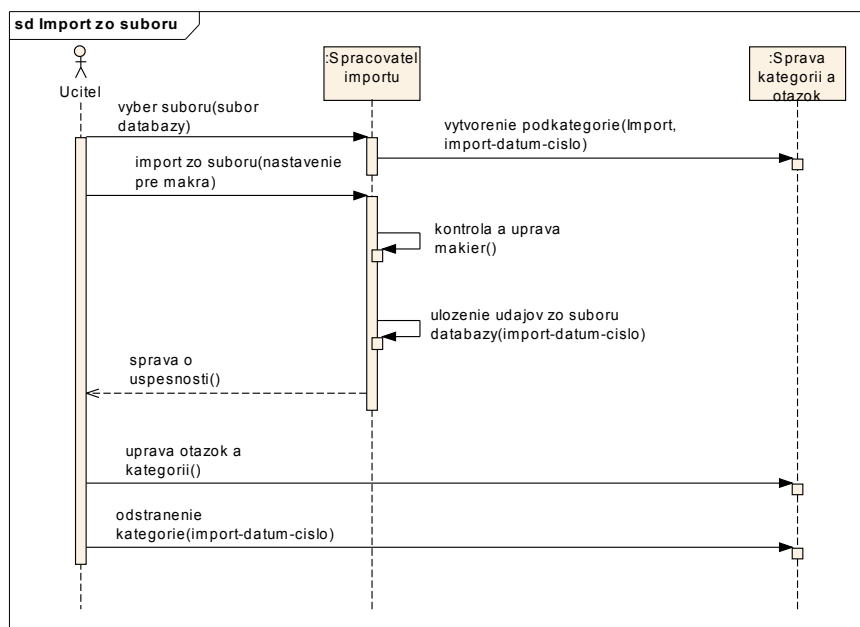
Obrázok 3.12: Import zo súboru databázy do databázy

Logiku importovania údajov do databázy najlepšie znázorňuje sekvenčný diagram. Na obr. 3.14 je znázornený sekvenčný diagram podsystemu importovania údajov z inej databázy pomocou súboru databázy. Používateľ vyberie súbor databázy. Následne nechá importovať údaje zo súboru databázy. Spracovateľ pre import databázy vytvorí podkategóriu *import-dátum-číslo* kategórie *Import*. Dátum bude aktuálny dátum importu a číslo bude prirodzené číslo, ak by používateľ importoval viac ako jedenkrát za deň. Následne skontroluje a prípadne zmení názvy importovaných makier, podľa používateľových nastavení. Po spracovaní makier dôjde k načítaniu údajov z databázového súboru do podkategórie *import-dátum-číslo*. Používateľ už môže s týmito importovanými údajmi pracovať podľa vlastného uváženia v rámci celej hierarchickej štruktúry svojich otázok. Ak nebude podkategóriu *import-dátum-číslo* naďalej potrebovať, môže ju odstrániť (viď UC-5).

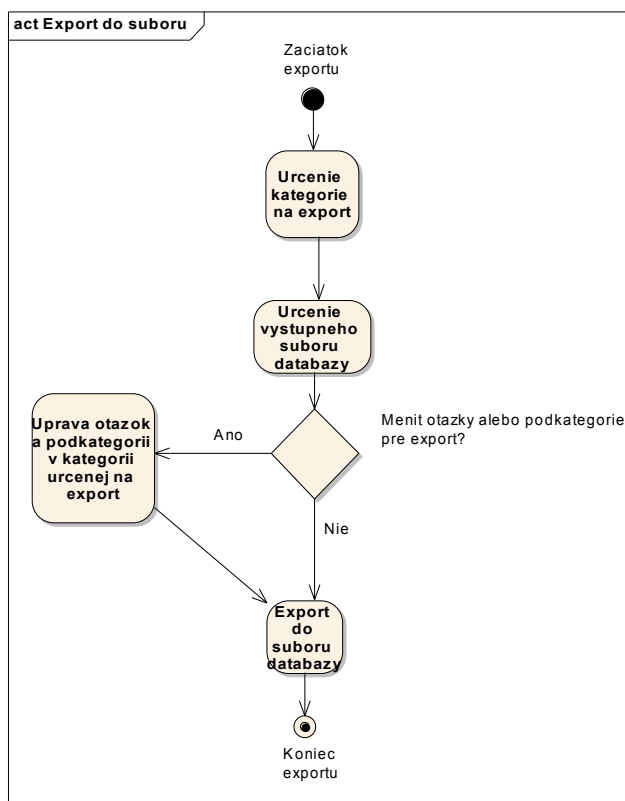
Na obr. 3.15 je znázornený diagram aktivít zachytávajúci scenár použitia podsystemu exportovania údajov z databázy do súboru databázy. Používateľ vyberie výstupný súbor databázy a kategóriu, ktorú chce exportovať (viď. 37). Vybratú kategóriu môže exportovať, alebo upraviť podľa svojich požiadaviek a až následne exportovať. Toto je výhodné ak používateľ vytvorí novú kategóriu na export a potom ju upraví podľa svojich požiadaviek.



Obrázok 3.13: Export databázy do súboru databázy



Obrázok 3.14: Sekvenčný diagram importu zo súboru databázy do databázy



Obrázok 3.15: Diagram aktivít exportu databázy. Diagram znázorňuje postup činností pri exporte

Kapitola 4

Návrh systému

Táto kapitola predstavuje hrubý návrh nášho systému. Tento návrh tvorí základ pre ďalšiu prácu na projekte. Niektoré časti budú spresnené alebo prepracované v podrobnom návrhu, ktorý bude vychádzať z nasledujúcich častí tohto dokumentu. Kapitola pred samotným návrhom uvádza prehľad uvažovaných technológií (časť 4.1). Nasleduje popis architektúry systému Genex (časť 4.2). Ďalej je návrh spresnený logickým a fyzickým modelom dát (časti 4.3 a 4.4). V Závere kapitoly sú uvedené navrhnuté akceptačné testy (časť 4.5).

4.1 Výber technológií

Pred samotným návrhom riešenia prebehla analýza použiteľných technológií na riešenie tohoto projektu. V tejto časti dokumentu je skrátený prehľad technológií, ktoré boli analyzované spolu s vyhodnotením a určením ich použitia v tomto projekte.

4.1.1 GlassFish

GlassFish je voľne dostupný aplikačný server, ktorý implementuje najnovšiu funkcionálnu zahrnutú v platforme Java EE 5. Táto platforma zahŕňa posledné verzie rôznych technológií ako napríklad JavaServer Pages (JSP) 2.1, JavaServer Faces (JSF) 1.2, Servlet 2.4, Enterprise JavaBeans 3.0, Java API for Web Services (JAX-WS) 2.0, Java Architecture for XML Binding (JAXB) 2.0, Web Services Metadata for the Java Platform 1.0 a mnohé iné. GlassFish bol a je vyvíjaný pod licenciou Common Development and Distribution License (CDDL), ktorá umožňuje vývojárom voľne upravovať jeho zdrojové kódy.

Z hore uvedeného popisu vyplýva, že GlassFish je veľmi silný a robustný nástroj na vývoj a správu aplikácií. Jeho inštalácia a manažovanie však nie je triviálne a vyžaduje si aj viaceré prostriedky. Preto zastávame názor, že pre potreby našej aplikácie by bolo použitie GlassFish zbytočnou komplikáciou, najmä čo sa týka efektívneho využívania prostriedkov a nasadzovania.

4.1.2 Apache Derby

Apache Derby je relačný databázový systém napísaný v Jave a postavený na štandardoch JDBC a SQL99 s čiastočnou podporou pre SQL-2003. Medzi hlavné výhody tohoto riešenia patrí možnosť použitia dvoma spôsobmi:

- vnorený (embedded) mód,
- klient-server mód.

Vnorený mód umožňuje použitie plne funkčnej databázy v rámci Java aplikácií bez nutnosti inštalovať akýkoľvek server. Zjednodušuje tak nasadenie, testovanie a aj samotné používanie. Nevýhodou vnoreného módu je nemožnosť viacerých simultánnych pripojení. Túto nevýhodu je možné obísť použitím klient-server módu. V tomto prípade je ale situácia mierne skomplikovaná, keďže je nutné najprv nainštalovať samotný server a následne nastaviť pripojenie v klientských aplikáciách. Databáza je v oboch prípadoch obsiahnutá v jedinom súbore na súborovom systéme. V klient-server móde je možné zabezpečiť prístup k databáze pomocou štandardných autentifikačných metód. Je možná aj implementácia pomocou LDAP, prípadne iných mechanizmov.

Identifikované vlastnosti a obmedzenia boli zobrazené do úvahy pri výbere databázového systému Apache Derby pre zabezpečenie úschovy dát v tomto projekte.

4.1.3 Groovy

Jazyk Groovy je jazyk so všetkými rysmi moderných skriptovacích jazykov. Pre svoj chod využíva Java Virtual Machine, takže sa dá ľahko prepojiť s ľubovoľnou Java knižnicou či frameworkom. Vďaka tomu, že Java podporuje UNICODE, nemá Groovy problémy s diakritikou ako Ruby či Python. Takisto má Groovy vďaka prepojeniu s Javou k dispozícii oveľa viac knižníc než Ruby alebo Python. Stručne povedané Groovy je dynamický typovací jazyk so syntaxou Javy a expresivitou Pythonu spolu s ďalšími prvkami od iných jazykov. Už na prvý pohľad vyzerá veľmi pekne, elegantne a nechýba mu ani správna dávka pragmatickosti, čo je vidieť hlavne pri práci s databázami.

Medzi ďalšie výhody Groovy patrí:

- natívna podpora regulárnych výrazov,
- natívna podpora rôznych značkovacích jazykov ako XML, HTML, SAX, W3C DOM, Ant a iné,
- reťazce je možné spájať priamo operátorom "+",
- regulárne výrazy sa môžu používať priamo v reťazcoch,
- statické a dynamické určovanie typov,
- množstvo nových pomocných metód v JDK.

Medzi nevýhody Groovy patrí najmä fakt, že ide o pomerne nový jazyk a nie je ešte celkom zdokumentovaný. Použitie tohto nástroja sme zvažovali hlavne v súvislosti s možnosťou, že chceme vytvárať aplikáciu typu klient-server. V takomto prípade by výhody Groovy stáli za riziko použitia pomerne novej technológie. Ale keďže ideme vytvárať klasickú "standalone" aplikáciu, rozhodli sme sa ju implementovať v jazyku Java. Toto rozhodnutie podporil i fakt, že predpokladáme, že v tomto projekte bude v budúcnosti pokračovať ďalší tím a Java je predsa len rozšírenejší jazyk než Groovy.

4.1.4 L^AT_EX analyzátoary

Latex analyzátoary sú pre náš systém dôležité z hľadiska exportu testov do systému Moodle. Analýza je potrebná pre zachovanie formátovania otázok a odpovedí, ktoré sú zadávané vo formáte L^AT_EX. Takto analyzované otázky sú následne konvertované do formátu HTML, ktorý je už priamo podporovaný systémom Moodle.

4.1.5 T_TH prekladač

Program T_TH (Tex To HTML) umožňuje transformáciu L^AT_EXu do formátu HTML4.0. Program pracuje ako filter, načítava štandardný textový vstup a ten transformuje na textový výstup v HTML. Dokáže rozpoznávať základné textové zvýraznenia `\it` `\sl` `\bf`. Odhaduje približný typ písma. Podporuje definície makier `\gdef`, `\xdef`, `\global`, `\def`, `\global`, `\edef`, `\def`, `\edef`. Rozpoznáva základné L^AT_EX príkazy. Nerozpoznáva žiadne rozšírenia L^AT_EXu a príkaz `\usepackage`. Zoznam podporovaných príkazov je možné nájsť na domovskej stránke programu. Program bol vytvorený s využitím nástroja *flex*.

Program je implementovaný v jazyku C. Vzhľadom na to, že pri vytváraní T_TH bol použitý nástroj *flex* je zdrojový kód menej prehľadný a jeho modifikácia by bola zložitá. Tento fakt podporuje aj štruktúrovanie programu, ktorý má skoro 28000 riadkov a nachádza sa v jednom zdrojovom súbore. Program T_TH by bolo preto vhodné použiť v stave ako je, alebo nepoužiť vôbec.

Kladnými vlastnosťami T_TH je pomerne dobrá rozpoznávacía schopnosť L^AT_EXového kódu. Príkazy HTML (*tags*) sú ľahko modifikovateľné, T_TH ich definuje ako makrá jazyka C. Ako bolo uvedené vyššie nevýhodou T_TH je slabá možnosť rozšírenia podporovaných príkazov.

4.1.6 latex2html

Prekladač latex2html je napísaný v jazyku Perl a slúži na konvertovanie L^AT_EX dokumentov do formátu HTML3.2.

Inštalácia vyžaduje jazyk Perl verzie 5 alebo vyšší, program Ghostscript, kompilátor L^AT_EX a program netpbm. Nevýhodou je veľa závislostí. Program latex2html má podporu veľkej časti L^AT_EXu a sú vytvorené moduly na podporu rôznych L^AT_EX balíkov. Program má modul zo značnou podporu štýlu `html.sty`. Podporuje definície nových príkazov a makier a bezchybne zvláda tabuľky.

Jednou z vlastností programu latex2html je, že ak narazí na L^AT_EX konštrukciu, ktorá nemá HTML ekvivalent, transformuje túto časť na bitmapu, čo nie je vhodné pri exporte otázok do Moodle. Toto bolo pozorované pri testoch.

Zdrojový kód programu je slabo zdokumentovaný, kód má pomerne zložitú konštrukciu bez komentárov. Program má veľkú pamäťovú zložitú, ktorá rýchlo rastie so zväčšujúcim vstupným dokumentom. Na odstránenie je v systéme Windows potrebné umožniť príkazovému interpretu pristupovať k väčšej časti pamäte. Zrejme tieto vlastnosti spôsobuje samotný Perl, pretože vstup načítava ako jeden reťazec, nad ktorým sú vykonávané zložité operácie.

4.1.7 H^EV^EA

H^EV^EA je prekladač L^AT_EXu do HTML4.0 napísaný v jazyku OCAML. Vzhľadom k tomu, že priamo rozumie L^AT_EXu nemá závislosti na žiadnych externých progra-

moch. Vďaka tomu rozhodne patrí medzi najrýchlejšie konvertory.

$H^E V^E A$ priamo podporuje úpravy výstupu pomocou \LaTeX makier. V niektorých prípadoch je nutné upraviť makrá používané v \LaTeX texte, ktorý sa prekladá, aby ich mohla $H^E V^E A$ spracovať. $H^E V^E A$ dokáže fungovať aj ako filter, teda vstupný \LaTeX text dostáva na štandardný vstup a na štandardnom výstupe vytvorí HTML preklad. Veľkou výhodou je možnosť vytvárať čiastkové výstupy, teda nie celú HTML stránku. Zaujímavosťou je tiež podpora balíka listings, teda zvýrazňovanie syntaxe programovacích jazykov. Pri testoch kvality výstupov boli výsledky $H^E V^E A$ najlepšie z testovaných prekladačov.

Podobne ako v prípade $T_T H$ by bolo vhodné $H^E V^E A$ použiť nezmenený, alebo vôbec. Je to spôsobené najmä implementáciou v OCAML, súboroch, ktorá nie je uspokojená na ďalšie úpravy. Čiastočne sa táto nevýhoda dá obísť pomocou už spomínaných \LaTeX makier, ktoré sú štandardným spôsobom pre rozširovanie funkcionality $H^E V^E A$.

4.1.8 Aspektovo-orientovaný prístup

Modulárnosť softvérového systému je jedna z dôležitých vlastností, ktorú sa návrhári systému pokúšajú dosiahnuť. Modulárny systém má mnoho predností medzi ktoré patrí jednoduchšia správa, údržba alebo prípadné rozšírenie systému. Jedným z novších prístupov, ktorý napomáha k dosiahnutiu tejto vlastnosti je aspektovo-orientovaný prístup. Táto paradigma poskytuje nové prostriedky pomocou ktorých je možné modulárne vyjadriť zložité vzťahy v systéme.

S využitím tohto prístupu je možné oddeliť kód, ktorý nesúvisí výhradne z primárnym poslaním triedy a vyčleniť ho do samostatnej časti programu. Jednou z možností využitia aspektov v našom systéme je možnosť spravovať perzistenciu dát medzi databázou a aplikačnou vrstvou v takomto module.

Ďalšia možnosť využitia aspektovo-orientovaného prístupu sa črtá pri potrebe využitia niektorého z návrhových vzorov. Objektovo-orientované vzory boli reimplementované pomocou aspektovo-orientovaného prístupu a v mnohých prípadoch sa zlepšila modularita takéhoto systému [HK02]. Využitie môžu byť aj niektoré aspektovo-orientované návrhové vzory.

Aplikácia aspektovo-orientovaného prístupu pritom len minimálne ovplyvní návrh systému. Rozhodnutie využiť tento prístup by znamenalo len zmenu kompilátora kódu. Nový kompilátor pritom produkuje výstup, ktorý je takisto spustiteľný na Java Virtual Machine.

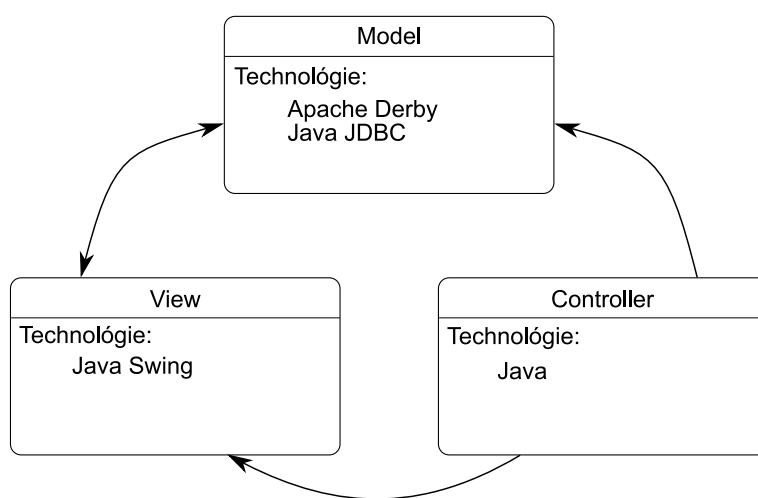
4.2 Architektúra systému Genex

V prvej fáze návrhu architektúry sa uvažovalo nad dvoma možnými riešeniami. Prvou možnosťou bolo viacpoužívateľské prostredie kde by mohlo pracovať viacero používateľov s jednou vzorkou otázok a testov. Druhým riešením bolo navrhnúť a implementovať jednoduchší jednopoužívateľský systém.

Vzhľadom k pôvodným požiadavkám bolo napokon zvolené modifikované riešenie typu jednopoužívateľský systém s možnosťou rozšírenia na viacpoužívateľský systém v budúcnosti. Tento spôsob riešenia samozrejme kladie väčšie nároky na návrh databázy, no využitie Apache Derby ako databázového systému (viď 4.1.2) umožňuje aj takéto zmeny prístupu bez väčších problémov.

Vzhľadom k požadovanej funkcionalite a previazanosti väčšiny funkcií systému s dátami uloženými v databáze bola zvolená architektúra MVC (Model View Controller). Okrem iného sa takto v implementačnej fáze zjednoduší využívanie návrhových vzorov. Ako príklad môže slúžiť využitie návrhového vzoru Observer (pozri [GHJV95]) medzi časťou View a Model. Celkovo MVC architektúra sprehladní implementáciu, ktorá bude navyše flexibilnejšia vo viacerých ohľadoch.

V rozdielnych častiach architektúry budú použité rôzne technológie ako vidno na obr. 4.1. Model architektúry bude v pozadí zabezpečovať už spomínaný databázový systém Apache Derby. Samotný logický a fyzický model databázy je opísaný v častiach 4.3 resp. 4.4. Grafické používateľské rozhranie bude vytvárané v knižnici Swing, ktorá je priamo založená na modele MVC a podporuje ho.

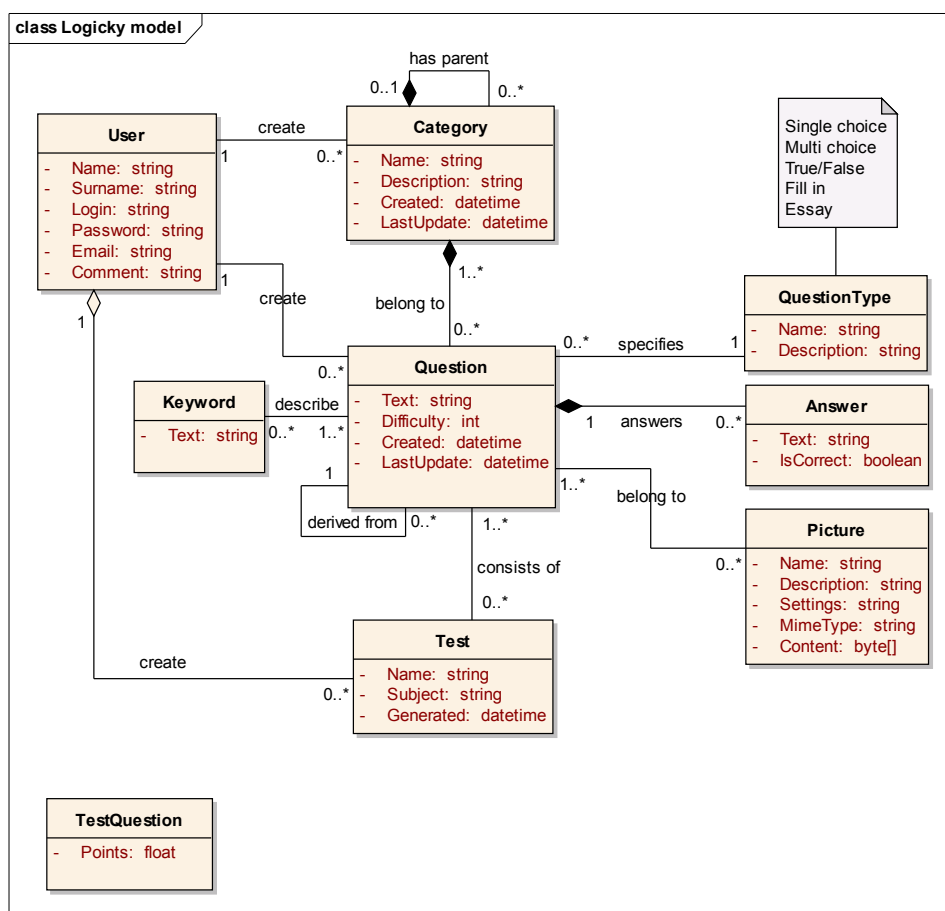


Obrázok 4.1: Architektúra systému a použité technológie

4.3 Logický model

Pre tvorbu logického modelu sme si ako základ zobrali logický model systému Test-Gen. Hlavným dôvodom je fakt, že tento systém bol vytváraný v rovnakom prostredí ako náš systém a mali až na pár rozdielov rovnakú špecifikáciu. Samozrejme sme urobili niekoľko úprav, ktoré vyplynuli z rozdielov v špecifikácii. Logický model je zobrazený na obr. 4.2.

Základnou entitou v systéme je entita *Question*, ktorá reprezentuje otázku. Táto entita je vo vzťahu s každou inou entitou v systéme. Každá otázka je buď originálom alebo je odvodená od nejakej inej otázky, čo je naznačené vzťahom entity so sebou samou. Odvodená otázka vznikne zmenou parametrov existujúcej otázky, alebo skopírovaním otázky. Každá otázka patrí do aspoň jednej kategórie. To znamená, že jedna otázka sa môže nachádzať aj vo viacerých kategóriách. Pre takéto riešenie sme sa rozhodli, aby sme sa vyhli obmedzeniu systému, že ak bude používateľ chcieť mať nejakú otázku v dvoch kategóriách, nemusí vytvoriť dvoch identických otázok, ale jednoducho umiestni otázku do oboch kategórií. Atribút *Difficulty* označuje náročnosť otázky. Táto náročnosť sa použije pri generovaní



Obrázok 4.2: Logický model

testu na nastavenie bodového hodnotenia jednotlivých otázok. Toto hodnotenie sa potom v prípade potreby upraví používateľom.

Ďalšou dôležitou entitou v systéme je entita *Category*, ktorá je podobne ako entita *Question* vo vzťahu so sebou samou. Teda každá kategória môže obsahovať ľubovoľný počet podkategórií do ľubovoľnej hĺbky. Takto si môže používateľ definovať vlastnú štruktúru triedenia otázok do tématických celkov, ktorá v spojení s možnosťou opakovania otázok vo viacerých kategóriách poskytuje vysokú mieru flexibility systému. Kategória a otázka sú vo vzťahu $m..n$, čo znamená, že kategória obsahuje ľubovoľný počet otázok a teda jedna otázka môže byť vo viacerých kategóriách.

Pre každú otázku je možné definovať množinu kľúčových slov, ktoré ju opisujú. Na to slúži entita *Keyword*, ktorá je vo vzťahu s otázkou $m..n$, čiže jedno kľúčové slovo môže zase opisovať viacero otázok, čo je logické.

QuestionType špecifikuje o aký typ otázky ide. Každá otázka je nejakého typu. Ak to bude typ otázky umožňovať, bude možné pridať k nej možnosti odpovedí,

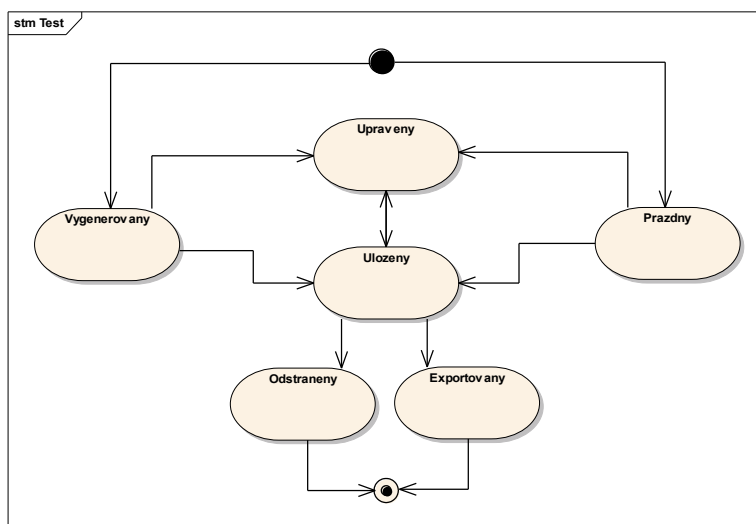
príčom každá odpoveď bude možnosťou práve na jednu otázku, čo je vyjadrené vzťahom otázky k odpovedi $1..n$. Pri každej odpovedi bude treba určiť, či ide o správnu alebo nesprávnu odpoveď.

Ku každej otázke je možné pripojiť viacero obrázkov (entita *Picture*), pričom každý obrázok môže byť použitý vo viacerých otázkach, čo vyjadruje vzťah entity *Picture* k entite *Question* $m..n$. Samotný obrázok bez otázky ale v systéme byť nemôže (viď obr. 4.4). Ak počet prepojení obrázka s otázkami klesne na nulu, obrázok sa z databázy vymaže. K obrázka je možné zadať aj jeho opis, ktorý sa bude zobrazovať v testoch.

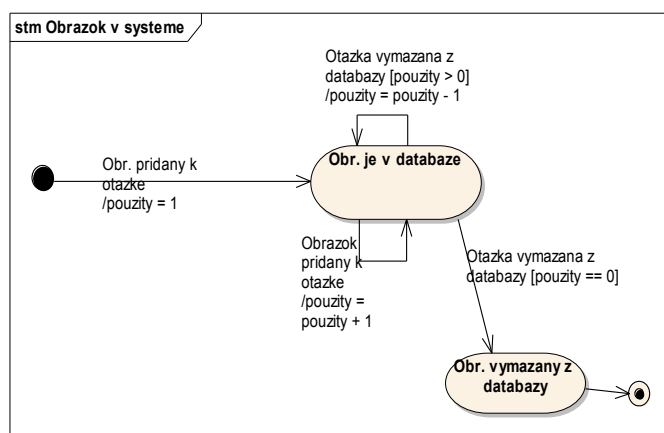
Entita *User* reprezentuje používateľov systému. Je vo vzťahu s entitami *Category*, *Question* a *Test*. Kardinalita týchto vzťahov je $1..n$. To znamená, že jednotlivé kategórie, otázky a testy sú vždy spojené s nejakým používateľom.

Na reprezentáciu testov slúži entita *Test*. S otázkou je vo vzťahu $m..n$, čo znamená, že sa testy obsahujú n otázok, ale naopak, že otázka sa môže nachádzať vo viacerých testoch. Ak sa nejaká otázka zmení (UC-7), tieto zmeny sa prejavia v každom teste, v ktorom sa daná otázka nachádza. Tieto zmeny sú totiž chápané ako opravy chýb a tie sa samozrejme musia prejaviť vo všetkých testoch. Ak nepôjde o opravu chyby, ale o zmenu otázky, bude potrebné vytvoriť novú verziu danej otázky (UC-12). Stavový diagram entity *Test* v systéme je na obr. 4.3.

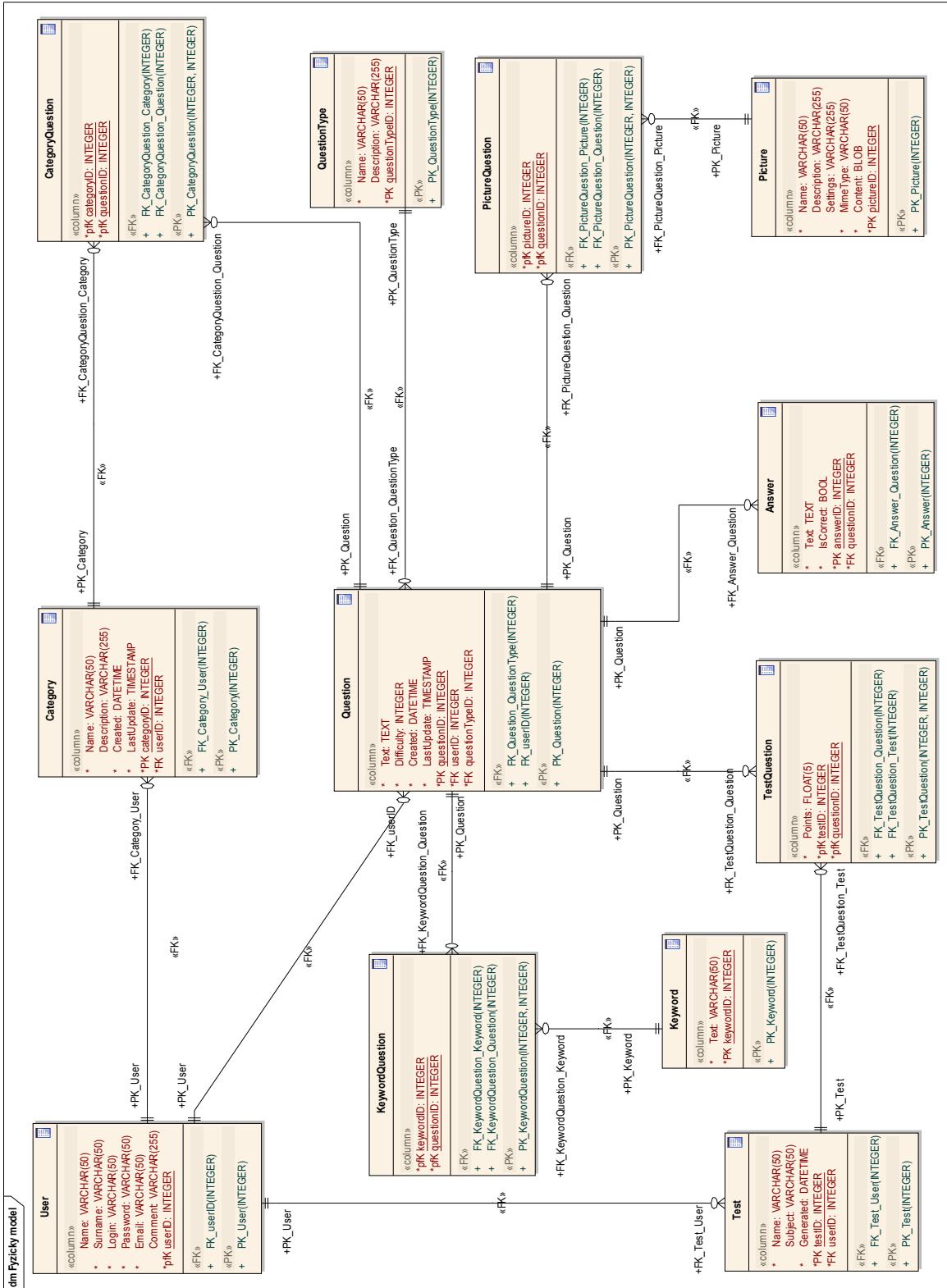
Logický model obsahuje aj 4 asociačné entity, ktoré slúžia na “rozbitie” vzťahov $m..n$. Entita *TestQuestion* obsahuje navyše atribút *Points*, ktorý reprezentuje bodové ohodnotenie danej otázky v konkrétnom teste.



Obrázok 4.3: Stavový diagram entity *Test* v systéme

Obrázok 4.4: Stavový diagram entity *Picture* v systéme

4.4 Fyzický model



Obrázok 4.5: Fyzický model systému

4.4.1 Opis dátových tabuliek

Tabuľka 4.1: Entita Test

Test				
Kľúč	Názov stĺpca	Typ	Povinný	Opis
	Name	VARCHAR(50)	áno	Názov testu (pomenovanie)
	Subject	VARCHAR(50)	nie	Názov vyučovacieho predmetu
	Generated	DATETIME	áno	Dátum vytvorenia testu
PK	TestID	INTEGER	áno	Primárny kľúč
FK	UserID	INTEGER	áno	Cudzí kľúč používateľa

Tabuľka 4.2: Entita Question

Question				
Kľúč	Názov stĺpca	Typ	Povinný	Opis
	Text	TEXT	áno	Text otázky
	Difficulty	INTEGER	áno	Náročnosť otázky
	Created	DATETIME	áno	Dátum vytvorenia otázky
	LastUpdate	TIMESTAMP	áno	Automaticky generovaná položka označujúca poslednú zmenu otázky
PK	QuestionID	INTEGER	áno	Primárny kľúč
FK	UserID	INTEGER	áno	Cudzí kľúč používateľa
FK	QuestionTypeID	INTEGER	áno	Cudzí kľúč otázky

Tabuľka 4.3: Entita QuestionType

QuestionType				
Kľúč	Názov stĺpca	Typ	Povinný	Opis
	Name	VARCHAR(50)	áno	Názov kategórie zobrazený používateľovi
	Description	VARCHAR(255)	nie	Opis typu otázky
PK	questionTypeID	INTEGER	áno	Primárny kľúč

Tabuľka 4.4: Entita Answer

Answer				
Kľúč	Názov stĺpca	Typ	Povinný	Opis
	Text	TEXT	áno	Text odpovede zobrazený v teste
	IsCorrect	BOOL	áno	Označenie správnej odpovede
PK	answerID	INTEGER	áno	Primárny kľúč
FK	questionID	INTEGER	áno	Cudzí kľúč otázky

Tabuľka 4.5: Entita Category

Category				
Kľúč	Názov stĺpca	Typ	Povinný	Opis
	Name	VARCHAR(50)	áno	Názov kategórie zobrazený používateľovi
	Description	VARCHAR(255)	nie	Opis kategórie
	Created	DATETIME	áno	Čas vytvorenia kategórie
	LastUpdate	TIMESTAMP	áno	Automaticky generovaná položka označujúca poslednú zmenu kategórie
PK	categoryID	INTEGER	áno	Primárny kľúč
FK	userID	INTEGER	áno	Cudzí kľúč používateľa

Tabuľka 4.6: Entita Keyword

Keyword				
Kľúč	Názov stĺpca	Typ	Povinný	Opis
	Text	VARCHAR(50)	áno	Text kľúčového slova
PK	KeywordID	INTEGER	áno	Primárny kľúč

Tabuľka 4.7: Entita Picture

Picture				
Kľúč	Názov stĺpca	Typ	Povinný	Opis
	Name	VARCHAR(50)	áno	Názov obrázka zobrazený používateľovi
	Description	VARCHAR(255)	nie	Opis obrázku zobrazený v teste
	Settings	VARCHAR(255)	nie	Doplnkové nastavenia zobrazenia obrázka v Latex formáte
	MimeType	VARCHAR(50)	áno	Typ obrázka
	Content	BLOB	áno	Údaje obrázka
PK	pictureID	INTEGER	áno	Primárny kľúč

Tabuľka 4.8: Entita User

User				
Kľúč	Názov stĺpca	Typ	Povinný	Opis
	Name	VARCHAR(50)	nie	Meno používateľa
	Surname	VARCHAR(50)	áno	Priezvisko používateľa
	Login	VARCHAR(50)	áno	Prihlasovacie meno používateľa
	Password	VARCHAR(50)	áno	Prihlasovacie heslo používateľa
	Email	VARCHAR(50)	áno	Adresa elektronickej pošty používateľa
	Comment	VARCHAR(250)	nie	Poznámka
PK	UserID	INTEGER	áno	Primárny kľúč

Tabuľka 4.9: Entita CategoryQuestion

CategoryQuestion				
Kľúč	Názov stĺpca	Typ	Povinný	Opis
PK/FK	categoryID	INTEGER	áno	Cudzí kľúč kategórie
PK/FK	questionID	INTEGER	áno	Cudzí kľúč otázky

Tabuľka 4.10: Entita KeywordQuestion

KeywordQuestion				
Kľúč	Názov stĺpca	Typ	Povinný	Opis
PK/FK	KeywordID	INTEGER	áno	Cudzí kľúč kľúčového slova
PK/FK	QuestionID	INTEGER	áno	Cudzí kľúč otázky

Tabuľka 4.11: Entita PictureQuestion

PictureQuestion				
Kľúč	Názov stĺpca	Typ	Povinný	Opis
PK/FK	pictureID	INTEGER	áno	Cudzí kľúč obrázka
PK/FK	questionID	INTEGER	áno	Cudzí kľúč otázky

Tabuľka 4.12: Entita TestQuestion

TestQuestion				
Kľúč	Názov stĺpca	Typ	Povinný	Opis
	Points	FLOAT(5)	áno	Bodové ohodnotenie otázky
PK/FK	TestID	INTEGER	áno	Cudzí kľúč testu
PK/FK	QuestionID	INTEGER	áno	Cudzí kľúč otázky

4.5 Akceptačné testy

Táto časť dokumentu obsahuje hlavnú časť navrhnutých akceptačných testov pre používanie systému.

Tabuľka 4.13: Pridanie novej kategórie

ID	1	Názov	Vývor kategóriu	Úroveň splnenia testu	Musi - Mal by - Mohol by
Pripad použitia		UC-1		Aplikačné rozhranie pre správu otázok a kategórií	
Rozhranie				Otestovanie funkcionality vytvorenia novej kategórie/podkategórie	
Účel				Musi existovať cieľová kategória	
Vstupné podmienky					
Výstupné podmienky					
Krok		Akcia		Očakávaná reakcia	Skutočná reakcia
1		Používateľ si vyberie cieľovú kategóriu		Je vybraná cieľová kategória	
2		Výber akcie vytvorenia novej kategórie		Zobrazená výzva na zadanie mena kategórie	
3		Zadanie názvu kategórie		Zobrazený zadany názov kategórie	
4		Potvrdenie vytvorenia novej kategórie		Pridaná nová kategória do databázy	

Tabuľka 4.14: Pridanie novej otázky

ID	2	Názov	Vytvor otázku	Úroveň splnenia testu	Musi - Mať-by - Meľet-by
Prípad použitia		UC-6			
Rozhranie				Úroveň splnenia testu	
Účel				Aplikačné rozhranie pre správu otázok a kategórií	
Vstupné podmienky				Otestovanie funkcionality pridania otázky do databázy otázok	
Výstupné podmienky				Musi existovať cieľová kategória	
Krok		Akcia		Očakávaná reakcia	Skutočná reakcia
1		Používateľ si vyberie cieľovú kategóriu		Je vybraná cieľová kategória	
2		Výber akcie pridanie otázky		Je zobrazený formulár na pridanie otázky	
3		Zadanie parametrov otázky		Všetky povinné položky formulára sú vyplnené	
4		Potvrdenie pridania otázky do databázy		Nová otázka je pridaná do databázy	

Tabuľka 4.15: Pridanie novej odpovede

ID	3	Názov	Vytvor odpoveď	Úroveň splnenia testu	Musí - Mal-by - Mohel-by
Prípád použitia		UC-13		Aplikačné rozhranie pre správu otázok a kategórií	
Rozhranie				Otestovanie funkcionality pridania novej odpovede otázke	
Účel				Must existovať daná otázka	
Vstupné podmienky					
Výstupné podmienky					
Krok		Akcia		Očakávaná reakcia	Skutočná reakcia
1		Používateľ si vyberie otázku, ktorej chce pridať odpoveď		Je vybraná otázka	
2		Výber akcie pridanie odpovede		Zobrazený formulár na úpravu otázky	
3		Zadanie parametrov odpovede		Vyplnené položky formulára na pridanie odpovede	
4		Potvrdenie pridania odpovede		Pridaná nová odpoveď na otázku do databázy	

Tabuľka 4.16: Pridanie obrázka k otázke

ID	4	Názov	Pridanie obrázka k otázke
Pripad použitia	UC 16	Úroveň splnenia testu	Musí - Mal-by - Mehet-by
Rozhranie		Apikačné rozhranie pre úpravu otázok	
Účel		Možnosť pripojiť k otázke obrázky	
Vstupné podmienky		V systéme musí existovať aspoň jedna otázka, používateľ musí mať k dispozícii obrázok	
Výstupné podmienky		Otázka bude mať priradený daný obrázok	
Krok	Akcia	Očakávaná reakcia	Skutočná reakcia
1	Používateľ otvorí otázku pre úpravy	Zobrazenie editora otázok	
2	Výber akcie pridanie obrázka	Zobrazí sa výber obrázkov, ktoré môže používateľ pridať k otázke	
3	Výber obrázka ktorý sa má pridať k otázke	V editore otázok je naznačené, že otázka má priradený daný obrázok	
4	Potvrdenie zmien v editore otázok	Zatvorenie zmien v editore otázok a vytvorenie prepojenia v databáze medzi otázkou a obrázkom	

Tabuľka 4.17: Generovanie testu

ID	5	Názov	Generovanie testu	Úroveň splnenia testu	Musí - Mal by - Mohol by
Prípady použitia	UC-24	Aplicačné rozhranie pre vytváranie a úpravu testov			
Rozhranie	Automatizované vytváranie testu				
Účel	V systéme existuje dostatočný počet otázok, aby z nich bolo možné vygenerovať test				
Vstupné podmienky	V systéme existuje dostatočný počet otázok, aby z nich bolo možné vygenerovať test				
Výstupné podmienky	V systéme existuje dostatočný počet otázok, aby z nich bolo možné vygenerovať test				
Krok	Akcia	Očakávaná reakcia	Skutočná reakcia		
1	Používateľ aktivuje modul pre vytváranie testov.	Zobrazí sa rozhranie pre tvorbu testov	Zobrazí sa formulár pre zadávanie parametrov generovania		
2	Zadanie parametrov pre vytvorenie nového testu a označenie možnosti generovania testu	Vygenerovanie a automatické uloženie testu			
3	Zadanie parametrov generovania do príslušného formulára				

Tabuľka 4.18: Export do L^AT_EXu

ID	6	Názov	Export do Latexu
Prípady použitia		UC 31	Úroveň splnenia testu Musí - Mat-by - Mehel-by
Rozhranie		Aplikačné rozhranie pre export, databázová vrstva	
Účel		Možnosť vygenerovania testu do formátu Latex	
Vstupné podmienky		V systéme existuje vygenerovaný test	
Výstupné podmienky			
Krok	Akcia	Očakávaná reakcia	Skutočná reakcia
1	Používateľ zvolí v systéme možnosť generovania výstupu do Latexu	Otvorí sa dialóg výstupných nastavení	
2	Používateľ vyberie požadovanú šablónu	Systém zobrazí dialógy na zadanie požadovaných údajov.	
3	Používateľ potvrdí vstupné údaje a inicializuje tak proces generovania výstupu	Systém zobrazí správu o výsledkoch procesu generovania výstupu.	

Literatúra

- [Alb03] Stephen T. Albin. *The Art of Software Architecture*. John Wiley & Sons, 2003.
- [Dra] Nikos Drakos. latex2html project. <http://saftsack.fs.uni-bayreuth.de/~latex2ht> [15.11.2007].
- [Fou] Apache Foundation. Apache derby. <http://db.apache.org/derby/> [15.11.2007].
- [GHJV95] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design patterns: elements of reusable object-oriented software*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995.
- [HK02] Jan Hannemann and Gregor Kiczales. Design pattern implementation in Java and AspectJ. In *OOPSLA*, pages 161–173, 2002.
- [Hut] Ian Hutchinson. Tth project. <http://hutchinson.belmont.ma.us/tth/> [15.11.2007].
- [Mar] Luc Maranget. Hevea project. <http://hevea.inria.fr/> [15.11.2007].
- [proa] Glassfish project. <https://glassfish.dev.java.net/> [15.11.2007].
- [prob] Groovy project. <http://groovy.codehaus.org/> [15.11.2007].
- [proc] Moodle project. <http://moodle.org> [15.11.2007].