FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

UNIVERZITA KOMENSKÉHO, BRATISLAVA

PORTÁL VÝUKOVEJ ROBOTIKY PRE PROJEKT CENTROBOT

2010

Ján Rajníček

FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

UNIVERZITA KOMENSKÉHO, BRATISLAVA

PORTÁL VÝUKOVEJ ROBOTIKY PRE PROJEKT CENTROBOT

(Bakalárska práca)

Študijný program : aplikovaná informatika

Študijný odbor: 9.2.9. aplikovaná informatika

Školiace pracovisko: Katedra aplikovanej informatiky

Školiteľ: Mgr. Pavel Petrovič, PhD.

Bratislava 2010

Autor: Ján Rajníček

Zadanie

Preskúmať technológie webového návrhu, navrhnúť a implementovať prototyp pre portál výukovej robotiky pre projekt CentroBot.

Čestne prehlasujem, že som túto bakalársku prácu vypracoval samostatne a použitím citovaných zdrojov.

.....

Ďakujem týmto všetkým, ktorí ma podporovali a pomáhali mi vytvoriť túto prácu, konkrétne Mgr. Pavlovi Petrovičovi, PhD. za jeho odborné vedenie, pripomienky a za čas a trpezlivosť, ktorú mi pri vypracovávaní tejto práce venoval.

Ján Rajníček

Abstrakt

Rajníček Ján: Portál výukovej robotiky pre projekt CentroBot. Bakalárska práca, Univerzita Komenského v Bratislave. Fakulta Matematiky, Fyziky a Informatiky. Vedúci záverečnej práce: Mgr. Pavel Petrovič PhD., Bratislava 2010.

Bakalárska práca sa zaoberá analýzou existujúcich trendov pri tvorbe webových aplikácií, návrhom a implementáciou prototypu pre portál výukovej robotiky pre projekt CentroBot. Teoretická časť sa rozoberá technológiami použitými pri realizácií prototypu, ich výhodami, nevýhodami a opisuje princípy ich fungovania. Ďalšia časť vysvetľuje požiadavky na systém, návrhom systému, rozoberá funkčnosť jednotlivých modulov a jeho realizáciou prototypu.

Kľúčové slová:

Java Servlet, Hibernate, prototyp pre portál výukovej robotiky, projekt CentroBot

Abstract

Rajníček Ján: Robotic educational portal for project CentroBot. Bachelor work, Comenius University in Bratislava. Faculty of Mathematics, Physics and Informatics. Supervisor: Mgr. Pavel Petrovič PhD., Bratislava 2010.

Bachelor thesis deals with analyze of existing trends used in web application deployment, design and implementation of prototype for robotic educational portal for project CentroBot. Theoretical part deals with technologies used during the process of deployment, their advantages, disadvantages and describes principles of system functioning. Next part explain requests on system, design of system, also explain functionality of specific system modules a realization of prototype.

Key words:

Java Servlet, Hibernate, prototype of robotic educational portal, project CentroBot

Obsah

1 Úvod

1.1 Motivácia

Rozvoj technológií a nástup informačnej spoločnosti nám priniesol nový pohľad a samozrejme i nové možnosti v dnešnom modernom vzdelávaní. Nastupujúci trend zavádzania elektronickej podpory vzdelávania (e-learning) umožňuje výrazne zvýšiť efektívnosť vzdelávania tým, že vzdelávací proces sa stáva nezávislým od miesta a času, podporuje študentov v hľadaní rôznych iných materiálov a to najmä prostredníctvom Internetu a nabáda študentov k samostatnej a tvorivej práci.

Jednou z foriem e-learningu je i On-line learning, ktorý ponúka možnosť výučby vo virtuálnom prostredí prostredníctvom Internetu. Toto prostredie ponúka:

- Štúdium interaktívnych a multimediálne zameraných materiálov
- Elektronická komunikáciu s učiteľmi, žiakmi
- Aktuálnejší obsah
- Riadenie procesu výučby
- Do výučby sa môžu aktívne zapojiť i ďalší tútori
- a mnohé ďalšie...

Tieto výhody zaručujú tomuto typu výučby rýchly rast a čoraz väčšiu obľúbenosť ako u učiteľov tak i u žiakov. Svoju bakalársku prácu chcem koncipovať tak, aby sa čo najviac priblížila k tejto modernej filozofii vzdelávania a to takým spôsobom, aby mnou vytvorená aplikácia niesla v sebe niektoré z najdôležitejších spomínaných výhod.

Vzhľadom k výraznému nedostatku a veľmi zlej dostupnosti materiálov zaoberajúcich sa edukáciou v oblasti robotiky, je nevyhnutné vytvoriť virtuálny priestor, ktorý by sa vyznačoval jednoduchosťou používania, ale zároveň by bol dostatočne robustný, aby poskytoval všetky potrebné možnosti k efektívnej výučbe. Dúfam, že touto bakalárskou prácou prispejem k vyplneniu medzery, ktorá existuje v tejto oblasti.

1.2 Cieľ práce

Cieľom bakalárskej práce je preskúmať technológie webového návrhu, navrhnúť a implementovať prototyp pre portál výukovej robotiky pre projekt Centrobot . Tento cieľ som rozdelil na nasledujúce podciele:

1. Analýza tvorby moderných webových aplikácií
2. Vytvoriť návrh portálu výukovej robotiky
3. Realizácia portálu výukovej robotiky ako RIA aplikácie

Prvá časť, analýza tvorby moderných webových aplikácií sa zaoberá problematikou vzniku a potreby vývoja tohto typu aplikácií ako dôležitým prvkom pre výmenu informácií a dát. Obsahuje odpoveď na otázku čo je webová aplikácia a rozoberá pojem RIA ako nový moderný, pokrokový prístup vývoja.

Druhá časť návrh, obsahuje popis používateľských skupín, základnú architektúru a rozdelenie aplikácie na samostatné moduly. Jednotlivé moduly sú ďalej popísané z pohľadu funkcionality a obsahujú schémy a diagramy v modelovacom jazyku UML.

V tretej časti sa nachádzajú ukážky funkčnosti aplikácie a stručný popis použitých riešení s rozborom ich kladných a záporných stránok.

1.3 Projekt CentroBot

Centrobot je medzinárodná iniciatíva pre výskumné a vzdelávacie inštitúcie v regióne Viedeň-Bratislava. Cieľom projektu je vytvoriť v regióne medzinárodné kompetenčné centrum v perspektívnej oblasti - robotike. Projekt zahŕňa medzinárodné robotické súťaže v regióne Viedeň-Bratislava. Záujemcom ponúka jednoduchý prístup k robotike vo viedenských výskumných a vývojových laboratóriách. Výskumná a vývojová infraštruktúra sa rozšíri o virtuálne internetové robotické laboratórium v Bratislave. Projekt chce podporovať inovatívne myšlienky. Vedecké výsledky sprístupňuje verejnosti na vedeckých konferenciách, alebo v odborných časopisoch. Okrem súťaží sa projekt Centrobot zameriava i na popularizáciu

robotiky, podporu využitia robotiky vo vzdelávaní. Združenia zapojené do projektu organizujú semináre, výmenné prednášky, letné školy, podporujú výuku na všetkých stupňoch škôl. Pre zvýšenie účinnosti týchto aktivít je potrebné zvýšiť dostupnosť informácií. Aj preto si riešitelia projektu stanovili za cieľ vybudovať portál, ktorý bude združovať informácie o výukovej robotike. V rámci pracovného balíka 2 aktuálneho projektu Centrobot navrhli technický a didaktický koncept. Cieľom mojej práce je realizovať prototyp podľa tohto rámcového návrhu, ktorý je možné nájsť na adresách:[6]

- http://intern.centrobot.eu/w/WP2.2_Technical_Concept
- http://intern.centrobot.eu/w/WP2.3_Didactic_Concept.

2 Analýza súčasných webových aplikácií

Internet, ako najväčšie komunikačné médium na svete hrá dnes obrovskú úlohu v našich životoch aj keď si to možno nie vždy uvedomujeme. Internet je pravdepodobne najsilnejším motorom globalizácie, čo ho stavia do veľmi silnej pozície v rámci celosvetových potrieb na šírenie a zdieľanie informácií. Jedným z príkladov môže byť globálna ekonomika, ktorej základy stoja práve na Internete a výmene dát prostredníctvom neho. Tieto potreby dali základ pre vývoj webových aplikácií, ktoré by naše potreby dokázali uspokojiť. S príchodom nových technológií a štandardov sa webové aplikácie zmenili do tej miery, že pomaly dokážu konkurovať aplikáciám desktopovým a predpokladá sa že budúcnosť leží práve vo vývoji tohto typu softwaru. Nástup WEB 2.0 znamená výraznú zmenu pohľadu na používanie Internetu. Z média, ktorého hlavný cieľ bolo podať informáciu sa zrazu stáva miesto pre ľudí, pre ich aktivity, komunikáciu a zdieľanie informácií. Podľa WEB 2.0 sú to práve obyčajní používatelia, ktorí budú tvoriť dominantnú časť obsahu Internetu a to je možné práve vďaka novým nástrojom, ktoré ľudia dostali do používania a tými nástrojmi sú práve webové aplikácie. [10]

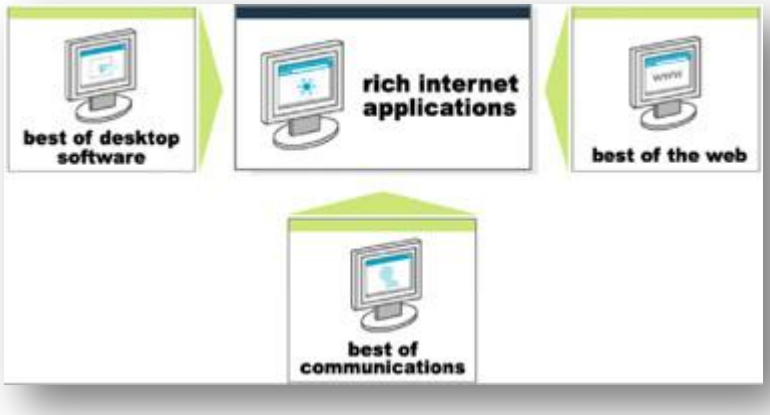
2.1 Čo je webová aplikácia ?

Webová aplikácia je aplikácia, ktorú webový prehliadač využíva v roli klienta. Môže ňou byť napr. návštevná kniha, chat, tabuľkový procesor... Medzi nesporne najväčšie výhody patrí oslobodenie vývojárov od závislosti na operačnom systéme, nezáleží či má klient na svojej strane MAC, Linux..., dôležitá je len prítomnosť webového prehliadača.

Dnešné webové aplikácie využívajú kombináciu technológií na strane servera (JSP, ASP, PHP, …) a na strane klienta (HTML, JavaScript, …). Klientska strana sa zúčastňuje hlavne na zobrazovaní informácií a interakcii s užívateľom, kým serverová strana má za úlohu

informácie poskytovať a to najmä z databázy a zároveň vykonávať úlohy, ktoré nie je možné alebo príliš nebezpečné robiť na strane klienta.

2.2 Rich Internet Applications (RIA)



Obrázok 1. Trend vývoja internetových aplikácií

Obrázok znázorňuje základný cieľ tvorby RIA aplikácií a tým je skĺbiť to najlepšie z oblasti desktop softwaru, web a komunikácie. Klasické črty desktopových aplikácií ako drag and drop, okamžitá validácia a formátovanie, rýchly reakčný čas na požiadavky a možnosť pracovať online aj offline je dnes už možné nájsť v mnohých kvalitných RIA aplikáciách. To najlepšie z webu ako platformová nezávislosť, okamžitý vývoj a získavanie dát a obsahu sú tiež významné črty, ktoré hovoria v prospech RIA aplikácií. Tým najlepším z oblasti komunikácie je myslená možnosť integrácie hlavne multimediálneho obsahu, videa a audia. Obrovskou zmenou oproti bežným webovým aplikáciam je skutočnosť, že pri RIA odpadá nutnosť obnovy stránok, čo znamená, že celá aplikácia pracuje a reaguje v reálnom čase. [11] Vďaka tomuto dynamicky generovateľnému obsahu môže celú aplikáciu tvoriť prakticky jedna web stránka a užívateľ sa nemusí preklikávať cez rôzne úrovne obsahu. Medzi techniky používané v tomto type aplikácií sa radia napríklad Ajax, Adobe Flash, Flex, Silverlight a ďalšie.

3 Technológie

Táto kapitola sa zaoberá a popisuje použité technológie v rámci portálu výukovej robotiky pre projekt CentroBot. V rámci každej podkapitoly je vysvetlený stručný princíp fungovania a výhody, prípadne nevýhody tej ktorej technológie.

3.1 JAVA

3.1.1 JSP

JSP je technológia, ktorá dovoľuje naplno použiť výhody jazyka JAVA na strane serveru. Dovoľuje tvorbu dynamicky generovateľného webového obsahu na strane servera a jeho následné odposlanie klientovi. Na rozdiel od iných serverových technológií ako PHP poskytuje možnosť využiť obrovskú kolekciu rozhrania API(Application Programming Interface) pre programovací jazyk JAVA. Výhodou JSP je skutočnosť, že sa nejedná o skriptovací jazyk, to znamená, že každá požiadavka nemusí byť interpretovaná od začiatku.

Princíp funkčnosti JSP:

1. Užívateľ vyzve prehliadač k zobrazeniu stránky vytvorenej ako JSP. Pri tomto procese sa tiež vytvorí request a odpošle sa na server.
2. Na serveri sa daný JSP súbor, čiže požadovaná stránka pošle do JSP Servlet Engine.
3. Ak sa súbor vytvára po prvý raz je skontrolovaný. V prípade, že súbor už existuje pokračuje sa bodom 6.
4. V tomto bode dochádza k generovaniu servletu podľa JSP súboru.
5. Vytvorený servlet je skompilovaný a vytvorí sa *.class súbor.
6. Servlet sa inštancuje, čo vyvolá metódy init() a service().
7. Servlet vygeneruje výstupný HTML kód, ktorý sa odpošle klientovi.

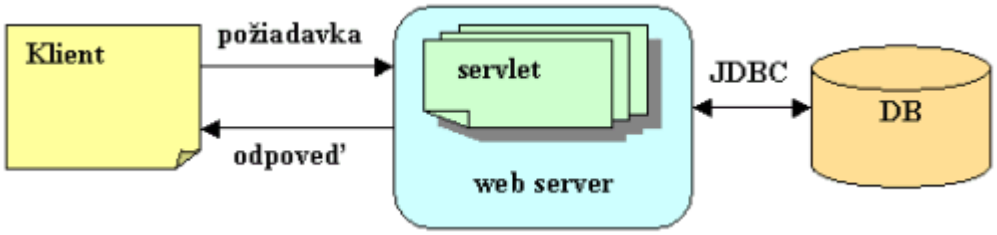
Výhodou JSP je platformová prenositeľnosť, to znamená, že môžu fungovať na linuxovom serveri a aj na Windows serveroch. [2]

3.1.2 JAVA Servlety

3.1.2.1 Predstavenie

Servlety sú dôležitou časťou platformy J2EE od SUN, ktoré v systémoch vystupujú ako komponenty pracujúce na strane servera. Na svoj beh potrebujú server s podporou JAVA a nainštalovanou JVM (Java Virtual Machine). JAVA servlety sú predchodcami technológie JSP, ktorá v skutočnosti poskytuje len určité zjednodušenie tvorby JAVA servletov. Špecifikácia hovorí o servletoch ako nevizuálnych komponentoch, teda nepoužívajú GUI. V mnohých prípadoch ale servlety generujú používateľské rozhranie na základe HTML kódu, ktorý sa v servlete zostaví a pošle sa klientovi. Využitie servletov je mnohostranné. Je možné pomocou nich posielať emaily, pracovať s databázovou , parsovať formuláre, používať cookies... [5]

Servlety sú založené na princípe požiadavka a odpoveď. Java Servlet API určuje rozhranie pre spracovanie týchto požiadaviek a odpovedí. Obrázok znázorňuje fungovanie a komunikáciu servletu.



Obrázok 2. Princíp fungovania JAVA servlet technológie

1. Klient pošle požiadavky
2. Servlet zistí novú požiadavku a pošle ju príslušnému servletu

3. Servlet spracuje požiadavku, vykoná naimplementovaná funkcionalitu a vráti odpoveď (napr. spomínaný html kód vo forme textového reťazca, ktorý sa prevedie na údajový stream) a pošle ju klientovi

3.1.2.2 Výhody servletov

- Portabilita a platformová nezávislosť – tieto vlastnosti dedia od jazyka JAVA
- Perzistencia a výkonnosť – servlet je potrebné skompilovať a inštancovať len prvýkrát. Neskôr sa volá práve táto už vytvorená inštancia. To nesporne tak nezaťažuje serverové prostriedky.
- Vychádzajú z JAVY – táto skutočnosť umožňuje servletom využívať výhody ako sú objektovosť, bezpečnosť, modularita …

3.1.2.3 Java Servlet API

JAVA Servlet API je skupina tried definujúca rozhranie medzi klientom a servletom. Nie je súčasťou JAVA framework, ale dostputný ako samostatný balíček, prípadne sa nachádza v J2EE. Skladá sa z javax.servlet a javax.servlet.http. Servlety podporujú ľubovoľné protokoly záleží na samotnej implementácií. Http balík pridáva konkrétne podporu HTTP protokolu. Z balíka javax.servlet je najdôležitejšie rozhranie definujúce množinu metód, najmä metódu service(), slúžiacu pre obsluhu klientskych požiadaviek. Na prácu s HTTP protokolom sa využíva trieda HttpServlet poskytujúca metódy Get a Post pre obsluhu požiadaviek podľa príslušnej metódy ako bola požiadavka odoslaná od klienta. [5]

3.1.2.4 Klient-servlet

Komunikácia klient so servletom nie je priama. Klient komunikuje so serverom, ktorý preposiela požiadavku na konkrétny servlet vďaka JAVA Servlet API. Server dokáže servlet spúšťať, inicializovať, ukončovať, uvoľňovať z pamäti. Za normálnych okolností sa na serveri

nachádza len jedna inštancia daného servletu, ktorý v nej zostáva aj pri nečinnosti. Štart procesu uvoľňovania z pamäti závisí na nastaveniach servera. Pri komunikácií servletu s viacerými klientmi, sa pre každého klienta vytvorí nové vlákno danej inštancie, tzn. servlet je multithreadový. Nové vlákna vytvára server. [5]

3.2 JavaScript

JavaScript je skriptovací programovací jazyk založený voľne na jazyku JAVA. Využíva podobnú syntax a metódy programovania, avšak je určený na spracovanie vo webových prehliadačoch. JavaScript je teda interpretovaný na strane klienta. Toto bolo aj dôvod pre vznik tohto jazyka pretože v dobe pomalého pripojenia bolo nutné minimalizovať komunikáciu so serverom a robiť čo možno najviac úkonov u klienta. Vývoj započala spoločnosť Netscape a po vydaní prehliadača Internet Explorer 3 sa do jeho vlastnej implementácie pustila i spoločnosť Microsoft, ktorá mala názov JScript. Neskôr sa tento jazyk podarilo štandardizovať do podoby ECMA-262 dokumentu. To ,čo robí tento jazyk výnimočným v oblasti webového programovania je to, že poskytuje veľkú flexibilitu hlavne v spojení s DOM (Document Object Model). Vďaka tomuto spojeniu technológií je možné zabudovať do stránok grafické efekty, dynamickú navigáciu a rôzne iné prvky, ktoré výrazne zvyšujú interaktivitu aplikácie s užívateľom. [1]

3.2.1 AJAX

Skratka AJAX (Asynchrónny JavaScript + XML) označuje techniku programovania webových aplikácií prostredníctvom XMLHttpRequest (XHR) objektu.

Asynchrónny znamená, že prehliadač nemusí pozastaviť operácie s aplikáciou počas komunikácie so serverom. To teda znamená, že prenos dát sa uskutočňuje na pozadí a beh aplikácie nie je zastavený.

JavaScript je tým elementom, ktorý riadi úkony v prehliadači. Dovoľuje aplikácií pripojenie na server, odosielanie a príjem dát. Pri použití s DOM technológiou dovoľuje získané dáta následne zobraziť.

XML sa stal kľúčovým prvkom pri prenášaní dát v textovej podobe prostredníctvom Internetu. Keďže internet bol navrhnutý na prenos textových dokumentov je práve textová podoba XML ideálna na prenos. Z toho dôvodu je pri použití AJAX techniky odpoveď zo serveru často práve vo formáte XML.

Technológia sa využíva v širšej miere od roku 2006 a okamžite zaznamenala obrovský nárast popularity, hlavne kvôli skutočnosti, že aplikácie využívajúce AJAX sa chovajú podobne ako aplikácie desktopové.[3]

3.2.1.1 XMLHttpRequest (XHR)

Tento objekt vyvinula spoločnosť Microsoft a prvýkrát bol použitý v prehliadači Internet Explorer 5, no neskôr bol implentovaný i do ostatných prehliadačov. XHR vystupuje v IE prehliadači ako objekt ActiveX, ktorý je súčasťou knižnice MSXML. Od verzia IE 7 a v prehliadačoch Mozzila, Opera, Chrome, Safari je natívne podporovaný ako objekt XMLHttpRequest. Napriek týmto odlišnostiam je objekt XHR v každom prehliadači kompatibilný s pôvodnou verziou v IE. Pred nástupom tohto objektu asynchrónna komunikácia prebiehala prostredníctvom skrytých elementov frame a iframe. Vďaka tomuto objektu je možné asynchrónne načítať údaje odoslané zo servera a následne ich vložiť do stránky pomocou modelu DOM, bez obnovenia stránky.[1]

3.2.2 jQuery

jQuery je knižnica s otvoreným zdrojovým kódom, ktorá ponúka robustnú funkcionalitu. Jej jadro je postavené pomocou selektorov jazyka CSS pracujúcich s elementmi modelu DOM. Kód využívajúci túto knižnicu nevyzerá príliš ako kód JavaScriptu, ale je skôr deklaratívny,

čiže popisuje to čo sa má stať. To zlepšuje jeho čitateľnosť a umožňuje rýchlejšie zvládnutie tejto technológie aj pre programátorov so základnou znalosťou JavaScriptu. Autorom tejto knižnice je John Resig, ktorý ju dodnes aj spravuje.[1]

3.2.3 JSON

JSON (JavaScript Object Notation) je dátový formát, ktorého cieľom je zjednodušiť prístup k dátam prichádzajúcim zo servera. Oproti formátu XML, ktorý vyžaduje pre získanie dát veľké množstvo kódu a v mnohých prehliadačoch sa jednotlivé postupy extrahovania dát líšia, formát JSON ponúka jednoduchý prevod z textovej podoby informácie do objektovej podoby. Tento prevod sa uskutočňuje vďaka funkcii eval, ktorá na základe dodaného reťazca s príslušnou syntaxou dokáže vrátiť výsledný JavaScriptový objekt s príslušnými vlastnosťami. Internet Explorer 8 a Mozilla 3.1 už obsahuje natívnu verziu analyzátora na spracovanie JSON reťazcov , čo svedčí o rastúcej popularite tohto formátu. [1]

Príklad:

```
var str = [ {  "name": "Jan", "age": 22 }, {  "name": "Martin", "age": 30 }]  //JSON reťazec
var people = eval(str);
v people je dvojprvkové pole objektov zložených z atribútov name, age
```

3.2.4 TinyMCE

TinyMCE je platformovo nezávislý JavaScriptový HTML WYSIWYG editor vydávaný pod open-Source licenciou spoločnosťou Moxiecode Systems AB. Editor poskytuje užívateľské rozhranie pre úpravu a formátovanie webového obsahu prostredníctvom značiek HTML jazyka. Výhodou je jeho rozsiahle aplikačné rozhranie a jednoduchá integrácia do stránok alebo do CMS systémov.

3.3 DOM

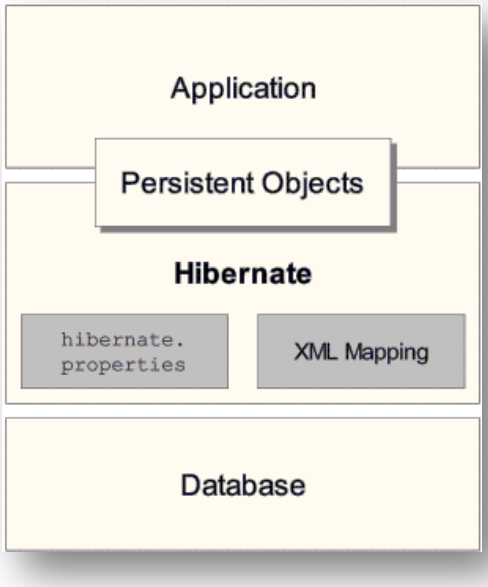
DOM (Document Object Model) je aplikačné rozhranie API pre dokumenty HTML a XML. DOM reprezentuje dokument ako hierarchický strom. Jednotlivé značky alebo informácie v dokumente sú uzly tohto stromu. Vďaka tomu je možné do tohto stromu pridávať, meniť, odstraňovať elementy a tak dynamicky meniť obsah dokumentu. Úroveň DOM Level 1 sa v roku 1998 stala odporučením od organizácie W3C a medzi prehliadačmi je takmer kompletne podporovaná.

3.4 Hibernate

3.4.1 Úvod

Hibernate framework je open-source nástroj pre objektovo relačné mapovanie založené na JAVA POJO (Plain Old Java Object) objektoch, ktoré sú namapované podľa tabuliek v databáze. Framework umožňuje využiť pokročilé črty objektového programovania ako dedičnosť, kompozície, kolekcie. Hibernate takiež poskytuje silný dopytovací mechanizmus prostredníctvom objektového jazyka HQL. Tieto vlastnosti robia z Hiberantu omnoho flexibilnejší nástroj získavania dát z databázy ako pomocou statického JDBC prístupu.[9]

3.4.2 Architektúra



Obrázok 3. Architektúra Hibernate

Databáza – slúži ako úložisko dát. Môže sa jednať o MySQL, Oracle a iné kompatibilné databázové enginy.

XML mapovanie – na mapovanie objektov sa používa schéma definovaná v XML súboroch. Schéma obsahuje definíciu perzistentných vlastností objektu ako i asociácie s inými objektami, takiež podtriedy a komponenty.

Hibarnate konfigurácia – konfigurácia Hibernate pomocou properties súborov. Konfiguráciu je možné vykonať programovo i s využitím XML súborov. Konfigurácia sa použije pri vytvorení session factory, vďaka ktorej môžeme vytvárať konkrétne session pre komunikáciu s databázou.

Perzistentné objekty – sú to normálne JAVA objekty (nie JavaBeans, EntityBeans), ktoré nemôžu implementovať rozhrania nejakého JAVA frameworku. Objekt k svojim atribútom poskytuje povinné get a set metódy a bezparametrický konštruktor.

Aplikácia – program, ktorý pomocou otvorenej session a POJO objektov modifikuje dáta v databáze.[9]

3.4.3 Príklad použitia

Ukazuje praktické použitie vyššie popísaných častí. Atribúty POJO objektu sú namapované do XML súboru. V samostatnom XML súbore sa nachádza Hibernate konfigurácia, ktorá sa použije pri inicializácií session factory. Časť aplikácie ukazuje konkrétne použitie Hibernate pri získavaní objektu User z databázy.

POJO objekt:

```
public class User  implements java.io.Serializable {
        private Integer id;
        private String name;
        private String email;
        public User() { }
        public User(String name,  String email) {
                this.name = name;
                this.email = email;
        }

        //get a set metódy
}
```

XML mapovanie:

```
<hibernate-mapping>
<class name=" User" table="user"
catalog="DBname">
    <id name="id" type="int">
        <column name="id" />
        <generator class="increment" />
    </id>
    <property name="name" type="string">
        <column name="Name" length="50" />
    </property>
    …
</class>
</hibernate-mapping>
```

Hibarnate konfigurácia:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
  <session-factory>
    <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
    <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/centrobot_sk</property>
    <property name="hibernate.connection.username">username</property>
    <property name="hibernate.connection.password">password </property>
    <mapping resource="Hibernate/User.hbm.xml"/>
  </session-factory>
</hibernate-configuration>
```

Aplikácia:

```
SessionFactory factory = new AnnotationConfiguration().configure().buildSessionFactory();
Session session = factory.openSession();
try {
    User myUser = (User) session.get(User.class, id);
} finally { session.close(); }
```

3.4.4 Dopytovací jazyk HQL

HQL je databázovo nezávislý jazyk, ktorý je postavený na preklade do SQL. HQL pracuje s objektmi a ich atribútmi, naproti SQL jazyku, ktorý je postavený na prácu s tabuľkami, riadkami. HQL je teda objektová obdoba SQL.

Príklady HQL:

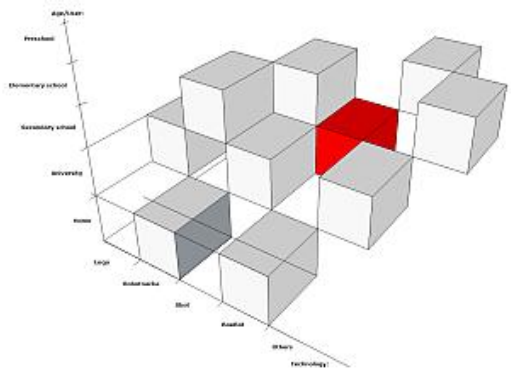
- *from Robtivity* – získa všetky objekty robtivity z databázy
- *from User user where user.name = 'meno'* – získa objekt User alias user, kde atribút name je rovný reťazcu meno.
- *delete from Request where id = '1'* – vymaže objekt Request, ktorého id je 1 z databázy

3.4.5 Porovnanie s JDBC

- Pri Hibernate odpadá nutnosť vytvoriť veľké množstvo dotazov, ktoré sú často veľmi príbuzné.
- Pri JDBC sa získava množina výsledkov, pomocou ktorej sa dáta mapujú na príslušný objekt ručne, čo výrazne zväčšuje dĺžku kódu
- Hibernate ponúka automatické načítanie asociovaných objektov naproti načítavaniu viacerých tabuliek pri JDBC
- Pri JDBC existuje určitá závislosť na použitej databáze, kvôli niektorým typom dotazov. Hibernate zvláda bezproblémovo migráciu medzi databázami. [8]

4 Požiadavky na systém

V tejto kapitole sa nachádza zoznam požiadaviek spolu s ich podrobnejším popisom. Okrem toho vysvetľuje úlohy a možnosti jednotlivých typov používateľov.



Obrázok 4. Uloženie cubes v 3D priestore

4.1 Pojmy

3D priestor virtuálny priestor, v ktorom sa nachádzajú cubes. Priestor je definovaný troma osami úroveň, technológia, doména

Robtivity konkrétny projekt, aktivita, lekcia zaoberajúca sa určitou oblasťou robotickej výučby s pevnou obsahovou štruktúrou

Cube organizačná jednotka združujúca množinu špecifických robtivity, definovaná súradnicami v 3D priestore

4.2 Účel aplikácie

Aplikácia bude primárne určená na zdieľanie edukačných materiálov pre oblasť robotiky. Medzi tieto edukačné materiály patria projekty (robtivity), informácie a iné metadáta, ktoré

môžu použiť učitelia a žiaci pri vzdelávaní. Bude taktiež poskytovať miesto pre diskusie a priestor pre virtuálne triedy (class).

4.2.1 Používatelia

Aplikáciu budú využívať nasledovné používateľské skupiny. Jednotlivé skupiny dedia funkcionalitu od predošlých skupín.

- Neprihlásení používatelia – táto skupina bude mať umožnené prezeranie obsahu jednotlivých robtivities a ich sťahovanie v rámci cubes pre offline prezeranie. Ďalej bude môcť prispievať do fóra a hodnotiť robtivity. V prípade, že užívateľ má záujem stať sa registrovaným používateľom, má možnosť podať žiadosť o registráciu. Ak už má používateľ vytvorené konto môže sa do systému prihlásiť. Ak sa jedná o užívateľa, ktorý má prístupové heslo do triedy môže sa prihlásiť do systému v roli žiaka.
- Prihlásení používatelia v roli žiakov – žiaci majú právo prezerať obsah triedy v ktorej sú prihlásení.
- Prihlásení používatelia v roli učiteľov – tento typ používateľov môže vytvárať nové robtivity a v prípade, že už je autorom niektorej robtivity nachádzajúcej sa v systéme má možnosť úpravy obsahu a práv danej konkrétnej robtivity prostredníctvom EditRobtivity komponentu. Má umožnené taktiež vytvárať, upravovať a mazať práve jednu triedu. Takisto môže schváliť žiadosť o registráciu, ale len v rámci svojej používateľskej skupiny.
- Prihlásení používatelia v roli administrátora – táto rola má prístup k celej funkcionalite systému, vrátane používania Maintenance komponentu pre správu aplikácie.

4.3 Požiadavky

4.3.1 Robtivity:

- Editácia cez WEB – túto požiadavku bude systém spĺňať využitím WYSIWYG editoru TinyMCE. V rámci editácie bude zahrnutá podpora formátovania textu a vkladanie obrázkov.
- Diskusné fórum – každá robtivity bude obsahovať diskusné fórum, ktoré bude otvorené pre všetkých používateľov.
- Upload – bude realizovaný prostredníctvom pluginu pre TinyMCE a umožní upload súborov do špecialného priečinka pre danú robtivity a následné vloženie uploadnutého súboru do robtivity ako odkaz, prípadne obrázok.
- Hodnotenie – všetci užívatelia aplikácie budú mať právo ohodnotiť robtivity.

4.3.2 Používateľské skupiny

- Úrovne obsahu – niektoré materiály budú sprístupnené len určitým používateľským skupinám.
- Registrácia – proces registrácie pozostáva z vytvorenia požiadavky nového užívateľa a následným potvrdením tejto požiadavky už existujúcim používate.

4.3.3 Class: (Učebná skupina

- Editácia cez WEB – rovnako ako pri robtivity.
- Prístupnosť – obsah classy bude prístupný len užívateľom, ktorý sa prihlásili do danej classy.

4.3.4 Ďalšia funkcionalita

- Off-line prezeranie cube – systém bude podporovať možnosť stiahnutia balíka pre off-line prezeranie obsahu cube, tzn. všetkých robtivity v danej cube, vrátane všetkých obrázkov.

- Export robtivity do pdf – každú robtivity bude možné stiahnuť na klientsky počítač ako pdf dokument, vhodný napr. pre tlač.
- Link manažment – v rámci správy aplikácie, užívateľ s administrátorským účtom bude mať umožnené zapnúť kontrolu všetkých odchádzajúcich linkov z portálu.
- Automatická záloha – systém bude mať schopnosť automatickej zálohy, pri ktorej vytvorí zozipovaný súbor s celou aplikáciou.

4.3.5 Štruktúra robtivity[7]

- Základné údaje
    - Cube súradnice

      Technológia

      Oblasť

      Úroveň

      Verzia

      Autor
    - Didaktické informácie

      Obsah

      Potrebné znalosti

      Časová náročnosť

      Cesta - ktorá cube nasleduje?

- Praktická časť
    - Príprava na kurz

      Prostredie

Vybavenie

Prezentácie

Poznámky

- Postup riešenia

  Popis ako lekcia prebieha

- Ukážkové riešenia
- Multimediálny obsah (Audio, Video,Obrázky)

- Technická časť

  - Konštrukčný manuál

    Ako postaviť robota

  - Popis komponentov

    Prehľad používateľských komponentov

    Manuál pre každú komponentu

  - Podpora
  - Fórum pre diskusiu

    Obsah

    Špecifikácia

  - Kontaktné údaje pre individuálnu pomoc

    Obsah

    Špecifikácia

  - FAQ

    Obsah

    Špecifikácia

  - Ohodnotenie a spätná väzba

Kritéria

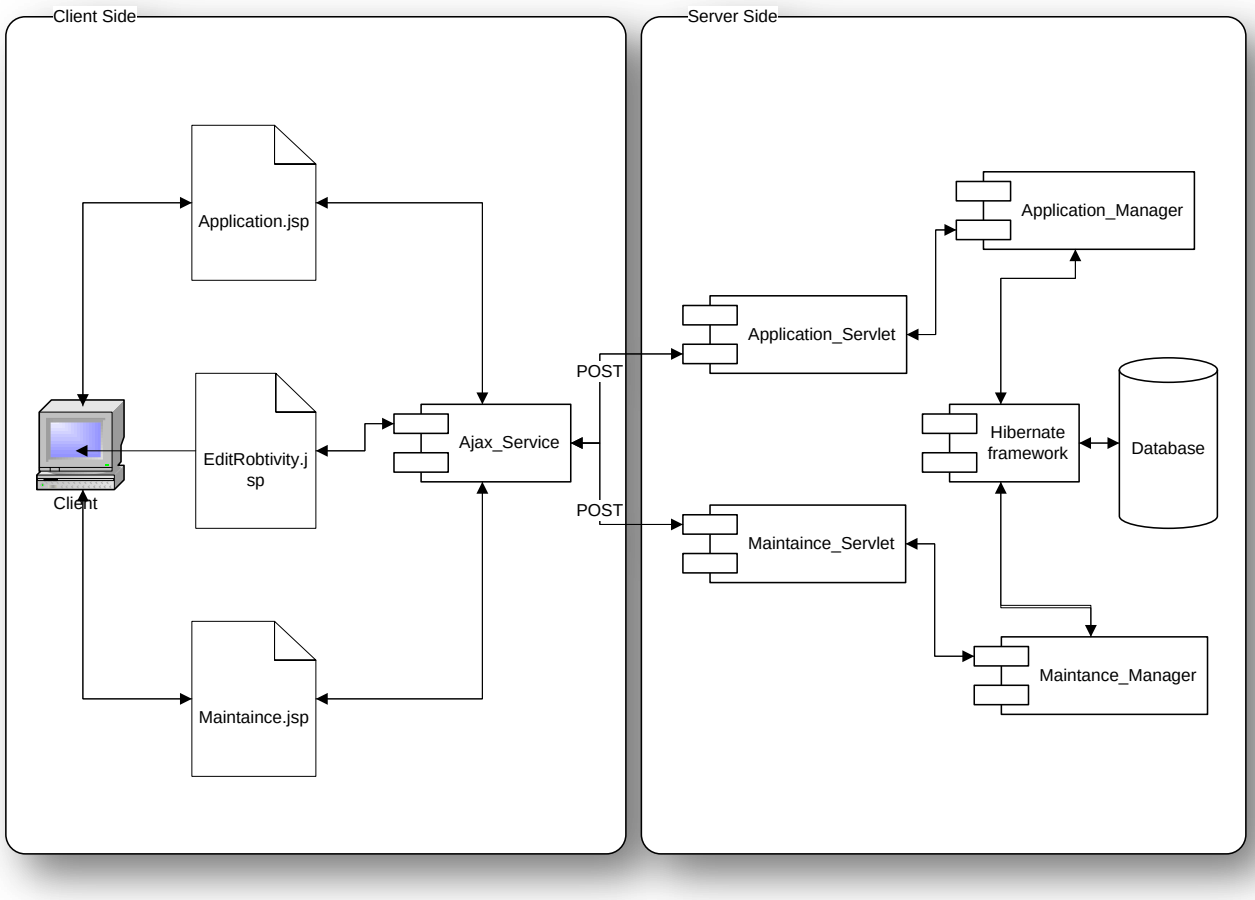
Špecifikácia

- Literatúra
    - Publikácie
    - Teória
    - Linky a zoznam literatúry
    - Autori

5 Návrh systému

5.1 Architektúra

Nasledujúci obrázok znázorňuje architektúru systému a rozdelenie jednotlivých častí



Obrázok 5. Architektúra systému

5.2 Moduly systému

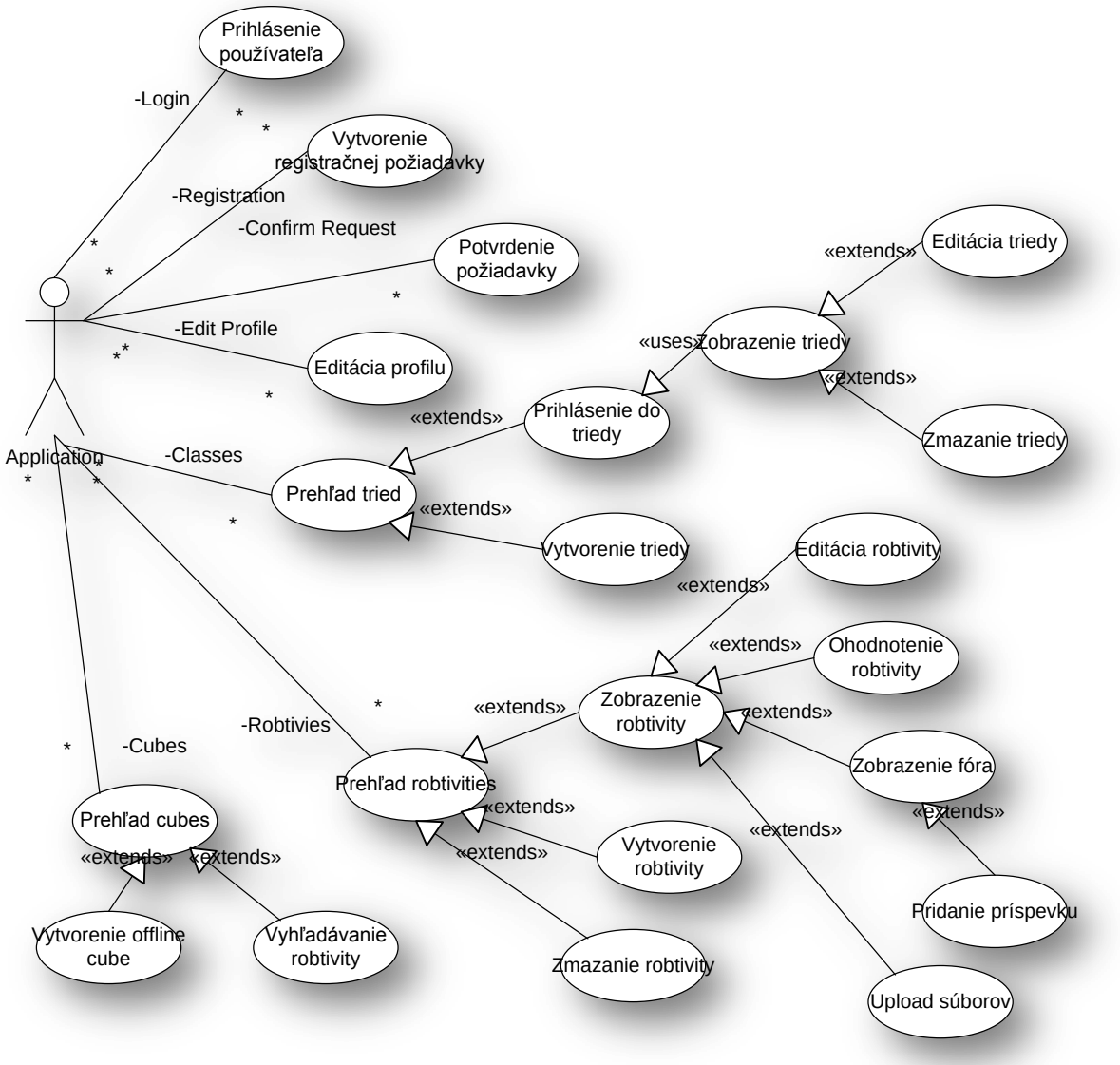
V tejto kapitole je špecifikovaný funkčný model jednotlivých komponentov aplikácie. Pre prehľadné zobrazenie funkčnosti a organizácie komponentov bol zvolený modelovací jazyk UML.

5.2.1 Application

Application modul slúži ako základné GUI pre používateľa webovej aplikácie. Modul je implementovaný ako JSP stránka. S pomocou AJAX technológie umožňuje dynamicky zobraziť dáta posielané prostredníctvom servletov a zároveň reagovať na vstupy od užívateľa bez nutnosti znovu načítania stránky. Pre grafické efekty využíva JavaScriptovskú knižnicu jQuery.

Funkcionalita modulu z pohľadu užívateľa:

- vytvorenie registračnej požiadavky
- prihlásenie používateľa
- potvrdenie požiadavky užívateľa
- editácia profilu užívateľa
- prehľad robtivities
- vytvorenie robtivity
- zobrazenie robtivity
- ohodnotenie robtivity
- vyhľadávanie robtivity podľa nastavených kritérií
- zobrazenie fóra pre každú robtivity
- editácia robtivity (obsah, práva)
- zmazanie robtivity
- prehľad cubes
- vytvorenie súboru pre offline prezeranie cube
- upload súborov pre danú robtivity
- vytvorenie triedy
- prihlásenie do triedy
- editácia triedy
- zobrazenie triedy
- zmazanie triedy
- logovanie aktivít



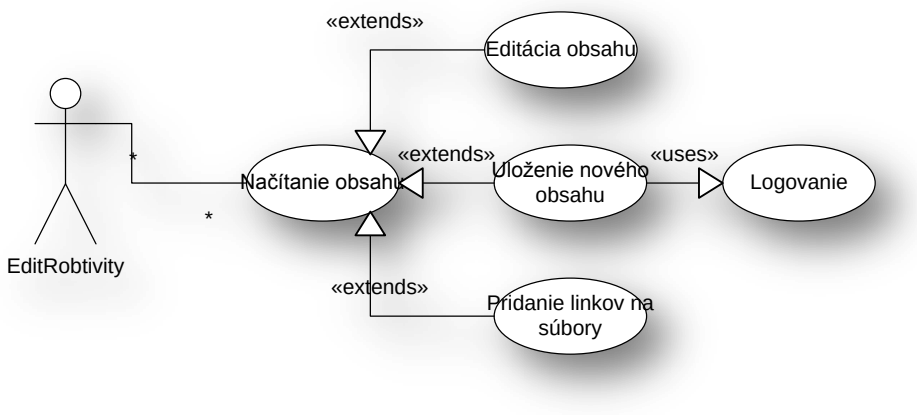
Obrázok 6. Jednotlivé udalosti vznikajúce pri interakcii s modulom Application

5.2.2 EditRobtivity

EditRobtivity modul umožňuje editovať obsah jednej sekcie v rámci danej robtivity. Editácia je možná prostredníctvom WYSIWYG editoru TinyMCE v grafickom režime alebo v HTML režime. Modul je implementovaný ako JSP stránka. Tento modul je prístupný užívateľovi len v prípade, že je prihlásený do systému a zároveň je autorom danej robtivity, ktorej sekciu má v úmysle editovať.

Funkcionalita modulu:

- editácia obsahu sekcie
- načítanie aktuálneho stavu sekcie
- uloženie nového stavu sekcie
- logovanie aktivít
- pridanie odkazu na súbor nachádzajúci sa v úložisku pre danú robtivity



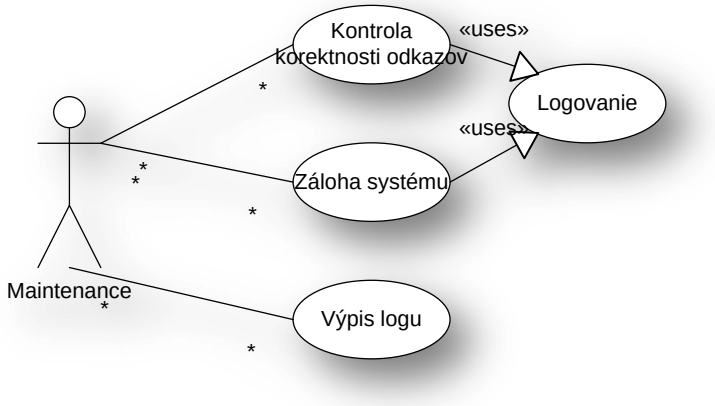
Obrázok 7. Schéma znázorňuje udalosti vznikajúce pri interakcii s modulom EditRobtivity

5.2.3 Maintenance

Maintenance je modul určený na správu a údržbu aplikácie. Tento modul je prístupný len pre prihlásených užívateľov s právami administrátora.

Funkcionalita modulu:

- kontrola korektnosti odkazov
- záloha systému
- logovanie aktivít
- výpis logu



Obrázok 8. Udalosti vznikajúce pri interakcii s modulom Maintenance

5.2.4 Ajax_Service

Hlavnou úlohou tohto komponentu je sprostredkovať asynchrónnu komunikáciu medzi klientskou a serverovou stranou, čo je docielené prostredníctvom XMLHttpRequest objektu. Komunikácia bude prebiehať prostredníctvom HTTP protokolu metódou POST. Komponent bude realizovaný ako HTML stránka, do ktorej sa bude vkladať JavaScriptový kód z nasledujúcich externých súborov:

5.2.4.1 Ajax_Framework.js

V tomto súbore budú naimplementované všetky JavaScript funkcie pre asynchrónnu komunikáciu, ktoré sa budú využívať v Application, EditRobtivity a Maintenance komponentoch.

5.2.4.2 Ajax_Viewer.js

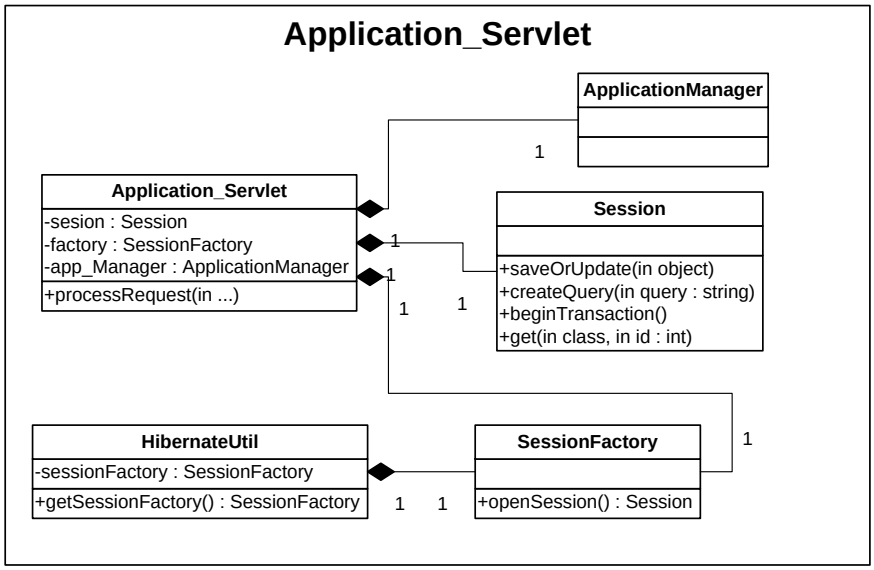
Tento súbor bude obsahovať JavaScript funkcionalitu, pre dynamické vytvorenie HTML štruktúry určenej na zobrazenie obdržaných údajov zo servera. Tvorba štruktúr bude prebiehať s využitím DOM technológie úrovne 1.

5.2.4.3 GUI_Functions.js

Súbor bude zahŕňať hlavne funkcionalitu zameranú na generovanie dynamického obsahu aplikácie, vytváranie tabuliek a ostatných prvkov určených na zobrazovanie údajov získaných z databázy.

5.2.5 Application_Servlet

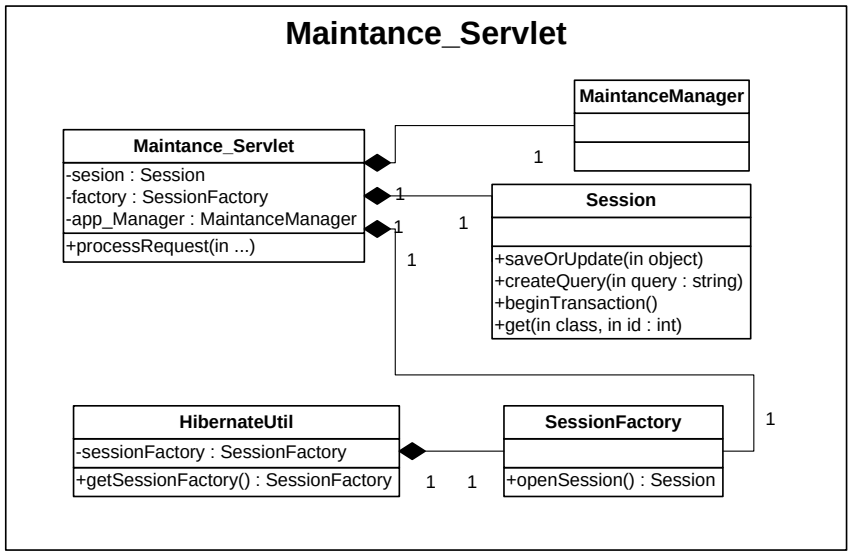
Application_Servlet má za úlohu spracovať požiadavku od Ajax_Service, konkrétne od objektu XHR. Pri každej obsluhe požiadavky sa skontroluje existencia objektu session pre komunikáciu s databázou prostredníctvom hibernate frameworku. Parametre pre spracovanie požiadavky sa budú posielať metódou POST a budú spracované v rámci metódy processRequest, kde sa získa meno akcie, ktorá sa má vykonať a ďalšie nutné parametre. Implementovaný bude ako JAVA servlet s nasledujúcimi triedami a triedami, ktoré sú vo vzťahu .



Obrázok 9. Triedny diagram Application_Servlet

5.2.6 Maintenance_Servlet

Maintenance_Servlet je JAVA servlet, ktorý bude fungovať na rovnakom princípe ako hore uvedený Application_Servlet, ale s rozdielom, že bude obsluhovať požiadavky od Maintenance modulu týkajúce sa údržby a správy aplikácie. Obrázok ukazuje rozdelenie do tried a ich vzájomné relácie.



Obrázok 10. Triedny diagram Maintenance_Servlet

5.2.7 Application_Manager

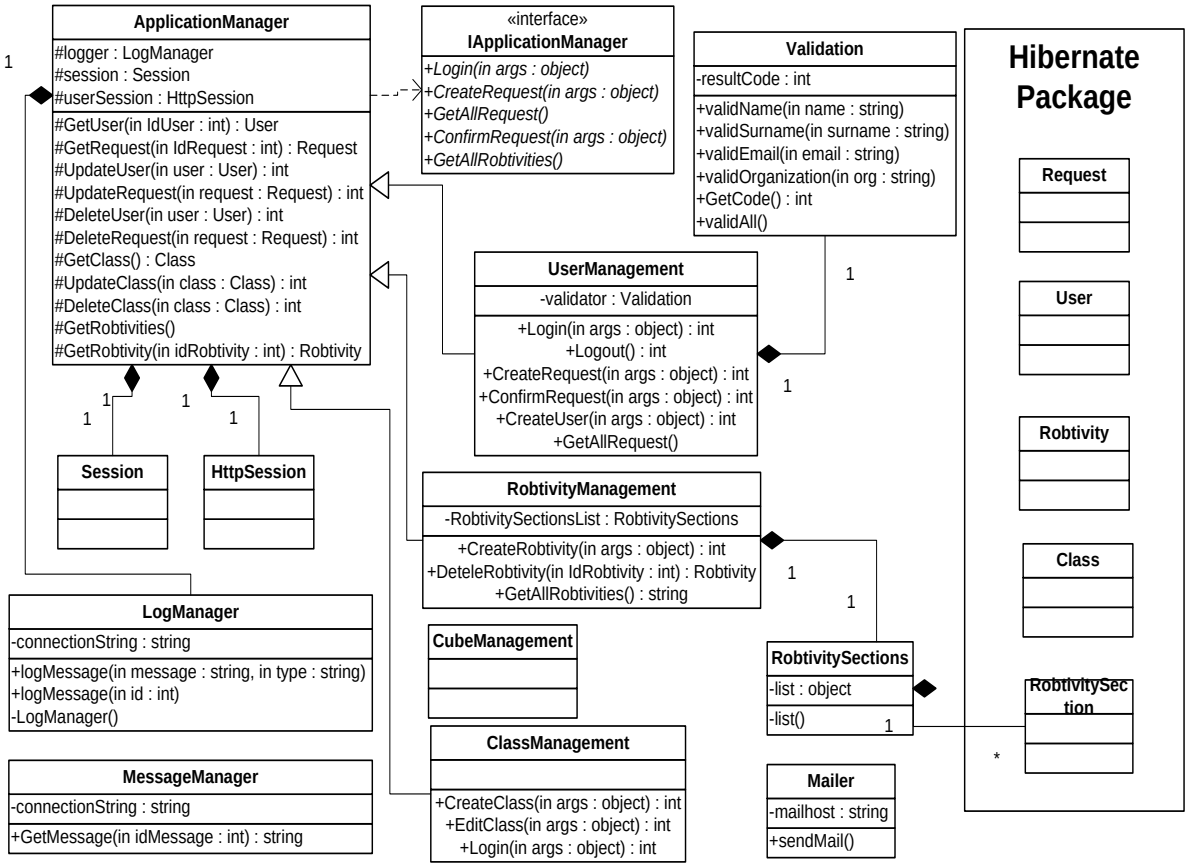
V tomto module bude naimplementované jadro systému, ktoré v sebe zahŕňa užívateľskú správu, správu jednotlivých robtivities , správu cubes a takiež funkcionalitu okolo class. Medzi ďalšie funkcie, ktoré bude táto časť systému obsahovať, taktiež patrí logovanie, posielanie emailov a získanie správ pre užívateľský výstup. Modul bude mať podobu balíčka tried, ktoré budú obsluhovať jednotlivé funkcionality.

Zoznam tried pre tento balíček:

IApplicationManager	Rozhranie pre komunikáciu s Application_servlet modulom. Poskytuje metódy pre: • Prihlasovanie uživateľa • Vytvorenie registračnej požiadavky • Získanie všetkých požiadaviek • Získanie všetkých robtivities • Potvrdenie registračných žiadostí
ApplicationManager	Obsahuje metódy k vyberaniu, editovaniu a mazaniu dát z databázy
UserManagement	Trieda je zdedená od ApplicationManager a implementuje niektoré z metód IApplicationManager rozhrania určené pre správu používateľov
CubeManagement	Trieda dedí od ApplicationManager a zároveň implementuje metódy IApplicationManager rozhrania zamerané na prácu s cube.
RobtivityManagement	Zdedená trieda od ApplicationManager, ktorá obsahuje metódy potrebné pre manažment robtivies v systéme.
ClassManagement	Trieda je zdedená od ApplicationManager a vykonáva operácie nad triedami (priestor, v ktorom učitelia editujú

	obsah pre žiakov)
Validation	Trieda používaná ku kontrole vstupným údajov získaných z formulárov pre tvorbu novej registračnej požiadavky a editácií profilu.

.



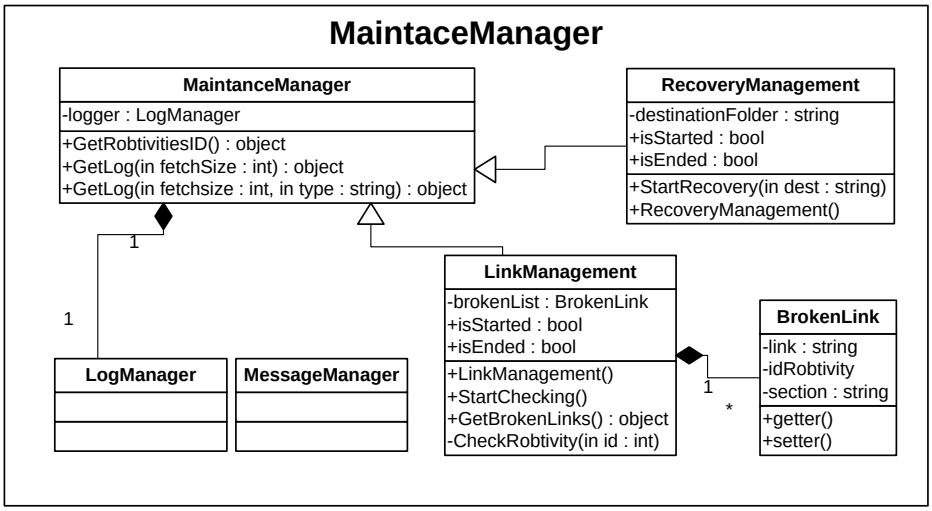
Obrázok 11. Triedny diagram ApplicationManager

5.2.8 Maintenance_Manager

Maintenance_Manager bude modul určený pre vykonávanie úloh zameraných na správu a údržbu aplikácie ako je záloha, kontrola adries a prezeranie logu. Modul bude vo forme balíčka JAVA tried. Triedny graf zobrazuje rozloženie tried a ich vzťahov.

Zoznam tried pre tento balíček:

Triedy pre správu a údržbu aplikácie	
MaintenanceManager	Trieda implementujúca funkcionalitu pre nevyhnutnú prácu s databázou a logovanie v Maintenance_Managery
LinkManagement	Trieda zabezpečuje proces kontroly korektnosti linkov, ukladá neplatné linky do databázy, prípadne ich získava z databázy
BrokenLink	Trieda reprezentujúca link smerujúci na neplatnú adresu. Ku každému linku je zaznamenaná informácia o jeho umiestnení
RecoveryManager	Trieda poskytuje možnosť vykonania zálohy celej aplikácie, vrátane všetkých uploadnutých súborov



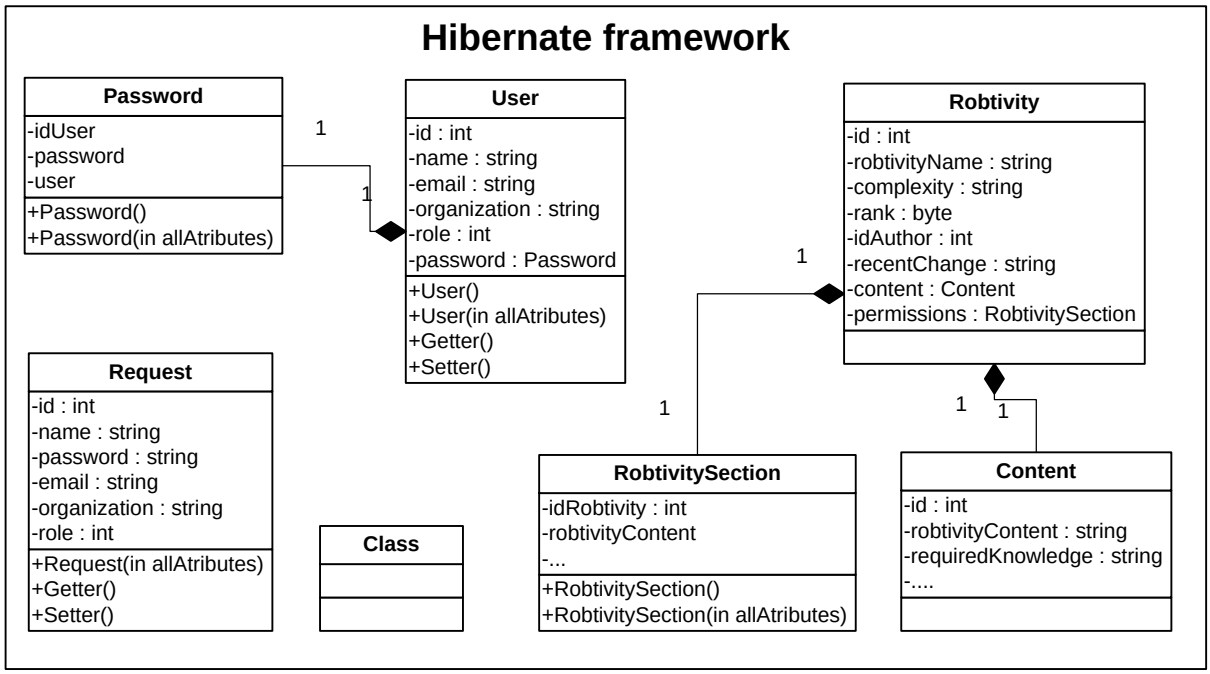
Obrázok 11. Triedny diagram MaintaceManager

5.2.9 Hibernate framework

Obsah modulu bude zameraný pre potreby komunikácie medzi Application_Manager a Maintenance_Manager modulmi s databázou. Tento balíček bude zahrňovať JAVA POJO triedy vrátane ich XML mapovaní a tiež konfiguračný súbor pre vytvorenie session factory.

Zoznam tried pre tento balíček:

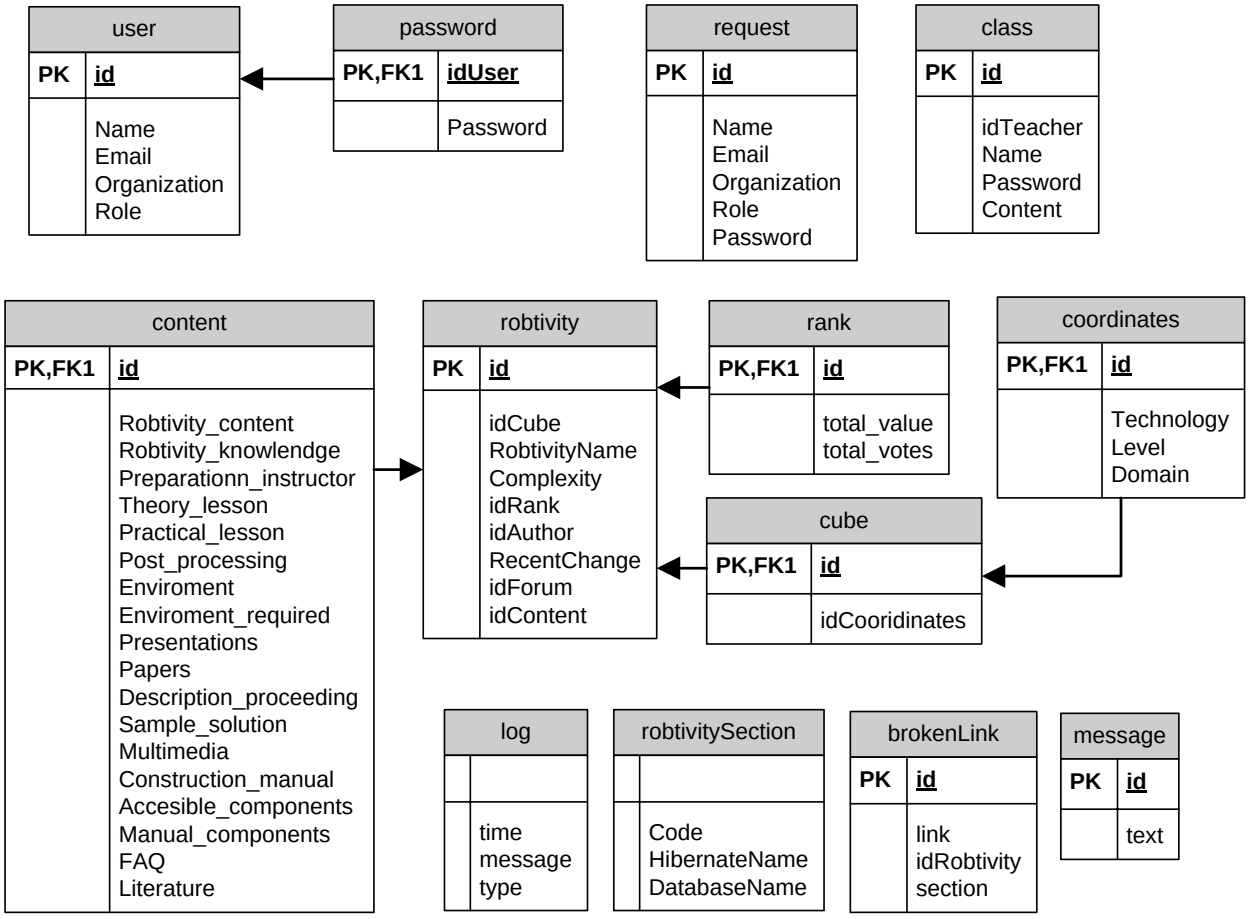
JAVA POJO triedy	
Request	trieda pre žiadosti o registráciu
User	trieda pre používateľa
Password	trieda pre heslá
Robtivity	trieda pre robtivity
Content	trieda pre obsah robtivity sekcií
Class	trieda pre triedy
RobtivitySection	pre robtivity sekcie s ich právami



Obrázok 12. Triedny diagram Hibernate framework

5.2.10 Databázový model

Schéma ukazuje rozdelenie údajov do tabuliek a relácie medzi nimi.



Obrázok 12. Databázový model

6 Realizácia

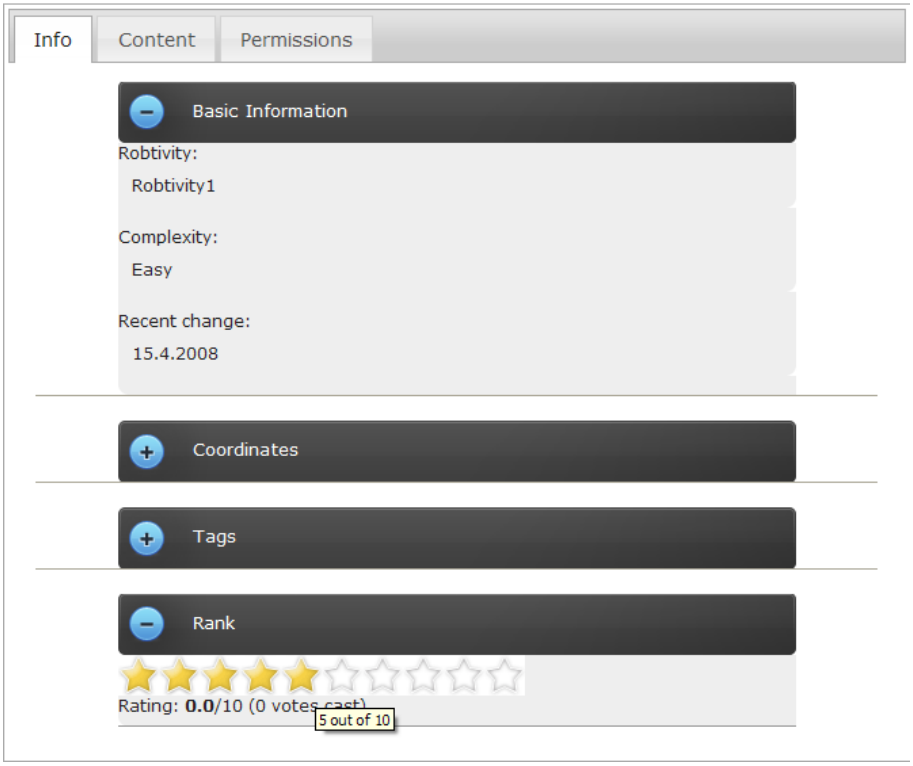
6.1 Popis výslednej aplikácie

Aplikácia je optimalizovaná pre spúšťanie na internetových prehliadačoch Internet Explorer 8.0 a Mozilla Firefox 3.6.3, pri obrazovom rozlíšení 1280x800 pixlov. Aplikácia bude umiestnená na serveri virtuallab.kar.elf.stuba.sk, kde sa nachádza nainštalovaný Apache Tomcat. Ako databázový stroj pre túto aplikáciu bol zvolený MySQL. Hlavné dôvody pre túto voľbu sú jednoduchosť, rýchlosť spracovania údajov a jedná sa o bezplatne poskytovaný nástroj. Pre správnu funkčnosť musí mať klient aktivované spúšťanie JavaScriptu v opačnom prípade aplikácia oznámi skutočnosť o jej nesprávnom fungovaní.

6.2 Príklady praktického použitia aplikácie

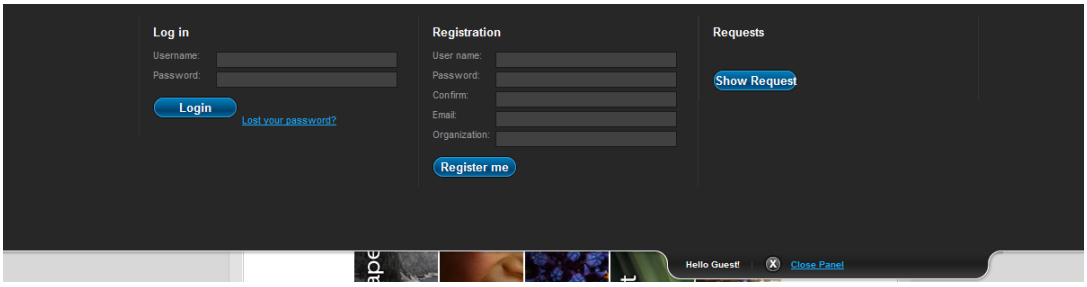
V aplikácií sú použité mnohé efekty, ktoré som vytvoril na základe návodov z Internetu, alebo som integroval už hotové voľne šíriteľné efekty.

6.3 Zobrazenie a práce s robtivity



Obrázok 14. Grafické rozhranie pre prácu s robtivity

6.4 Funcionalita pre správu používateľov



Obrázok 15. Grafické rozhranie pre prácu s používateľskými kontami

6.5 Riešenie problémov pri realizácii

- Neustála potreba posielať požiadavky na server prostredníctvom XHR objektu, ale s rôznymi parametrami a s rozdielnym spôsobom spracovania získaných údajov.

Riešenie : Vytvorenie vlastného jednoduchého AJAX frameworku pre posielania požiadaviek a spracovanie odpovedí, ktorý obsahuje nasledovné funkcie:

Zoznam funkcií v AJAX framework	
CreateHttpRequest()	Inštancovanie objektu pre potreby asynchrónnej komunikácie
GetTextCallback(url, method, param, element, callback)	Funkcia zostaví a odošle požiadavku na server a čaká na odpoveď. Po jej obdržaní zavolá funkciu v parametri callback a výsledok callback funkcie vráti do elementu
GetText(url, method, param, element)	Pracuje podobne ako funkcia GetTextCallback s rozdielom, že odpoveď sa rovno vráti do elementu
GetCallback(url, method, param, callback)	Obdoba funkcie GetTextCallback, kde sa ale výsledok spracovania nevracia do elementu, len sa vykoná volanie funkcie callback s potrebnými parametrami

- Realizácia kontroly odchádzajúcich linkov

  Riešenie : Táto funkčnosť pozostáva z parsovania všetkých sekcií vo všetkých robtivity uložených v systéme a extrahovaní odchádzajúcich linkov. Následne sa na každý jeden link pošle požiadavka podľa HTTP protokolu. Po obdržaní návratového kódu od serveru sa daný kód analyzuje.

```
URLConnection luf = linkURL.openConnection();
        if (linkURL.getProtocol().equals("http")) {
            HttpURLConnection huf = (HttpURLConnection) luf;
            String s = "";
            if (huf.getResponseCode() == -1) {  s = "wrong http response";  }
            if (huf.getResponseCode() == 200) {  s = "OK";  }
            return s;
        }
  return "bad link";
```

7 Záver

Dnešná doba sa vyznačuje obrovským množstvom informácií, ktoré treba tvoriť, spracovávať, vyhodnocovať. Toto množstvo je tak značne veľké, že nie je v silách jedného, či dokonca aj viacerých ľudí robiť všetky tieto činnosti.

Toto platí aj v oblasti robotickej výučby. Preto som sa snažil v tejto bakalárskej práci navrhnúť a implementovať taký systém, ktorý by umožnil tvoriť obsah výučby a iných edukačných materiálov všetkým používateľom so záujmom o robotiku. Zároveň by systém poskytoval úvod do oblasti robotiky a práci s robotmi. Mal by slúžiť ako oporný bod pre všetkých, ktorí prejavili záujem o vzdelávanie v tejto určite perspektívnej oblasti, či už sú to učitelia základných, stredných škôl, univerzít ale aj žiakov a študentov samotných.

Prototyp, ktorý sa podarilo navrhnúť a vytvoriť v rámci tejto práce by mohol slúžiť ako dobrý základ pre budúci vývoj ešte dokonalejšieho portálu. Využitím moderných technológií sa mi podarilo docieliť toho, že aj bežný človek, bez znalostí webových technológií, je schopný aktívne vytvárať obsah a zapájať sa tak do vytvárania virtuálneho priestoru určeného k robotickej výučbe.

7.1 Budúci vývoj

- Viacjazyčná podpora

Kedže organizácia CentroBot pôsobí medzinárodne je viacjazyčná podpora jednou z najdôležitejších budúcich vylepšení aplikácie. V rámci tejto podpory by systém implementoval aj preklad textu pomocou prekladača, napr. Google Translator, ktorý poskytuje vhodné API pre potreby našej aplikácie.

- Online testovanie

V rámci tried by bolo možné vytvárať a vyhodnocovať testy vytvorené učiteľmi, prípadne testy prevziať z databázy testov. Do určitej miery by testovací modul mohol podporovať aj tvorbu testov na základe generovania testovacích otázok.

- Inteligentné vyhľadávanie

Nutná súčasť modernej webovej aplikácie. V rámci tejto aplikácie by inteligentné vyhľadávanie mohlo fungovať ako fulltextové vyhľadávanie s podporou automatického dopĺňania výrazov vo vyhľadávacom formulári.

8 Literatúra

[1] JavaScript pro webové vývojáře, Nicholas Z. Zakas, Computer Press 2009

[2] JSP: JavaServer Pages, Barry Burd, Computer Press 2003

[3] Mistrovství v AJAXu, Steven Holzner, Computer Press 2003

[4] Java, kuchařka programátora, Ian F. Darwin, Computer Press 2006

[5] Java Servlets – predstavenie technológie

http://interval.cz/clanky/java-servlets-predstavenie-technologie

[6] http://www.centrobot.eu/sk/home/

[7] http://intern.centrobot.eu/images/f/fd/AP2_Didactic_concept.odt

[8] Pracujeme s Hibernate

http://ics.upjs.sk/~novotnyr/java/hibernate-tutorial/hibernate-tutorial.pdf

[9] Hiberante :: how know

http://neuron.tuke.sk/~kostelni/tutorialy/hibernate/index.html

[10] Web 2.0 Tutorials

http://www.roseindia.net/Technology-revolution/web2.0/index.shtml

[11] WHAT Are Rich Internet Applications?

http://www.perryinc.com/solutions/rich_internet_applications_what.aspx