evropský sociální fond v ČR
EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Platforma průmyslové spolupráce

CZ.1.07/2.4.00/17.0041

Název

Podpůrný nástroj pro výuku značkovacích jazyků

Popis a využití

- nástroj pro studenty pro zadávání úkolů k práci
- výuka: značkovací jazyky

Jazyk textu

- český

Autor (autoři)

- Jiří Krejčí

Oficiální stránka projektu:

- http://lasaris.fi.muni.cz/pps

Dostupnost výukových materiálů a nástrojů online:

- http://lasaris.fi.muni.cz/pps/study-materials-and-tools

Obsah

1. Úvod

V současné době je kladen velký důraz na snadnou výměnu informací mezi programy. XML[1] je jednou z technologií, která tento požadavek splňuje. Značkovací jazyky, jak se zmíněné XML označuje, dávají textu takovou podobu, aby jej bylo možné automatizovaně zpracovávat. Z pohledu základní interpretace můžeme značkovací jazyky rozdělit na procedurální a deklarativní. Procedurální značky definují pro prvek konkrétní akci, zatímco deklarativní značky definují prvek jako logickou část značkovaného textu [1].

Předmět Moderní značkovací jazyky a jejich aplikace vyučovaný na Fakultě informatiky Masarykovy univerzity seznamuje studenty se základními standardy, principy práce a technologiemi moderních značkovacích jazyků na bázi XML a jejich aplikací [2].

V současné době nemají studenti možnost automatizovaně procvičovat probíraná témata v rozsahu zmíněného předmětu. Existují on-line nástroje na procvičení jednotlivých témat, ale nejsou vytvořeny přímo pro podporu výuky. Studenti nemají k dispozici jednotný nástroj, kterým by mohli procvičovat všechny probírané okruhy.

Cílem této bakalářské práce je navrhnout a implementovat webovou aplikaci, která podpoří výuku značkovacích jazyků v rámci předmětu PB138 Moderní značkovací jazyky a jejich aplikace a umožnit tak studentům přístup k databance cvičení vytvořených přímo pro ně. Mezi hlavní probíranou látku předmětu patří validace DTD[2], XSL[3] transformace, práce s XSD[4] schématy, xQuery a xPathy výrazy. Vyvíjená aplikace by měla umožnit procvičování všech těchto témat.

Webovou aplikaci bude nutné implementovat tak, aby procvičování probíraných oblastí bylo snadné, intuitivní, přívětivé, automatizované a bylo ji možné použít při výuce formou on-line cvičení. V práci budou identifikovány uživatelské role, tak aby aplikace umožňovala pracovat jako přihlášený uživatel. Vyvíjený nástroj budou

[1] eXtended Markup Language

[2] Document Type Definition

[3] eXtensible Stylesheet Language

[4] XML Schema Definition

využívat nejen studenti, ale také vyučující, dalším cílem práce je umožnit vyučujícím snadno vytvářet nová cvičení a ty zpřístupňovat studentům, kteří by měli mít možnost vytvořená cvičení prohlížet a plnit.

Bakalářská práce je strukturována do sedmi kapitol. První kapitola se zabývá volbou metodiky návrhu a implementace. Je v ní popsáno použití SCRUM jako metodiky vývoje aplikace. Druhá kapitola popisuje nejdůležitější požadavky na aplikaci, které byly identifikovány a podrobněji se zabývá těmi nejdůležitějšími. Třetí kapitola - Použité nástroje seznamuje s nástroji, který byly využity při vývoji aplikace. Jde zejména o webové aplikační rozhraní Apache Wicket v kombinaci s frameworkem[5] Hibernate pro objektově relační mapování. Dále pak služba OpenShift a GitHub díky kterým bylo možné aplikaci nasadit na server a zpřístupnit ji na internetu. Čtvrtá kapitola podrobně popisuje samotnou aplikaci, její součásti a fungování. V páté kapitole je navrženo možné použití aplikace ve výuce. Popisuje, jak zadávat nová cvičení, zpřístupňovat jejich vzorová řešení, hodnotit řešení vytvořená studenty, vysvětluje jak umožnit studentům přístup do aplikace a prohlížet protokoly o absolvování cvičení. Závěrečná šestá kapitola zhodnocuje konečnou implementaci a navrhuje možnosti dalšího vývoje.

[5] Softwarová struktura, sloužící k podpoře programování a vývoji aplikace [3]

2. Metodika

Při implementaci aplikace byla využit agilní metodika SCRUM. Jedná se o metodiku řízení vývoje aplikace, která zahrnuje několik níže popsaných kroků. Cílem metodiky je stanovení požadavků na aplikaci, pánování, kontrola a evaluace procesu implementace. Předpokládá se práce v týmu s pravidelnými evaluačními schůzkami.

Prvním krokem SCRUM metodiky je definování požadavků na produkt tzv. user stories. Každý takový požadavek musí odpovídat na otázky: Kdo? Co? a Proč? požaduje. Soubor všech požadavků se označuje jako Seznam požadavků (Product Backlog) [4]. Ve druhém korku metodiky dochází k identifikaci důležitých požadavků a eliminaci těch nadbytečných, vzniká tzv. Release Backlog. Třetím a zároveň posledním krokem SCRUM metodiky před implementací je uspořádání a seskupení požadavků do skupin tak jak budou v průběhu času implementovány. Tyto skupiny požadavků se označují jako iterace nebo sprint. Následující kroky metodiky závisí na délce jednotlivých sprintů [5].

Přednášející a cvičící předmětu PB138 Moderní značkovací jazyky a jejich aplikace byly požádáni o sepsání požadavků na aplikaci tak aby na ně bylo možné aplikovat SCRUM metodiku. Výsledkem je seznam všech žádaných požadavků na aplikaci.

Dalším krokem podle metodiky je výběr důležitých požadavků, který probíhal formou diskuse na společné schůzce s přednášejícími a cvičícími.

Všechny stanovené požadavky byly rozděleny do skupin tak jak budou zpracovávány. Následná implementace probíhala bez zásahů cvičících a přednášejících a byla rozděleno do následujících čtyř iterací do kterým byly přiřazeny požadavky podle jejich důležitosti:

1. Spuštění všech potřebných nástrojů a technologií jako Apache Wicket, Hibernate, GitHub, Openshift, atd… (viz kapitola Použité nástroje) a vytvoření první funkční stránky.
2. Vytvoření pěti lekcí a cvičení týkající se validace DTD, XSL transformace, xPath výrazy, XSD schéma a xQuery výrazy.

3. Použití validačních nástrojů Xerces, možnost přidávání nových cvičení učitelem a zpětnou vazbu pro studenty při provádění XSL transformací.
4. Umožnit učiteli přidávat značky, filtrovat studenty, zobrazovat vzorové řešení cvičení a omezit registraci studentů pouze na určité období.

Každá iterace byla konzultována s vedoucím práce a průběžně zveřejňována prostřednictvím sužby OpenShift. Po čtvrté iteraci byla aplikace a její funkcionality představeny přednášejícím a cvičícím předmětu PB138 Moderní značkovací jazyky a jejich aplikace. Na základě tohoto představení vzniklo několik nových požadavků, který byla následně doimplementovány.

3. Požadavky

Tato kapitola se zabývá požadavky kladenými na aplikaci. Jejich kompletní seznam je uveden v příloze této práce. Hlavním požadavkem bylo automatizované procvičování témat uvedených v sylabu předmětu PB138 Moderní značkovací jazyky a jejich aplikace.

Identifikované požadavky se dají rozdělit do dvou skupin. První skupinu tvoří požadavky kladené na automatizované procvičování příkladů. Do druhé skupiny patří požadavky, které umožňují práci s aplikací nebo ji usnadňují. Patří sem zejména autorizace a autentizace uživatelů, obsluha práce s cvičeními a vytváření protokolů o absolvování cvičení.

Do první skupiny požadavků patří možnost řešení úkolů typu „Je zadáno XML a otázka je „Dopište DTD“ (Doplnění stávajícího částečného řešení)“. Tento typ cvičení se zaměřuje na problematiku vytváření DTD, jako deklarace typu dokumentu [6]. Cvičení je složeno ze zadaného XML dokumentu, ke kterému má student za úkol vytvořit validní DTD, případně doplnit částečné zadané DTD. Absolvováním tohoto typu cvičení student prokazuje znalost z oblasti tvorby DTD dokumentů. Dalším typem cvičení jsou XSLT transformace, tedy transformace XML dokumentů na XML výstup [7]. Ve cvičení student musí projevit schopnost vytvořit transformaci, která ze zadaného XML dokumentu vytvoří podle určených specifikací nový dokument na výstup. Třetím typem cvičení jsou XML schémata. Student má za úkol vytvořit k zadanému XML dokumentu schéma, které jej popisuje. K jednomu XML dokumentu je možné vytvořit několik různých schémat, proto je vhodné řešení částečně zadat, tak aby student pouze doplnil část řešení. Čtvrtým a pátým typem cvičení jsou výrazy xPath a xQuery. Student má ve cvičení za úkol vytvořit takový výraz, který bude splňovat požadavky uvedené v zadání příkladu.

Do druhé skupiny požadavků patří ty, které jsou potřebné k obsluze aplikace. Patří sem umožnění registrace a přihlašování uživatelů, rozdělení oprávnění uživatelů na studenty a učitele. Dále pak označení uživatelů specifickou značkou, možnost jejich filtrování, prohlížení protokolů o absolvování cvičení a jejich komentování.

4. Použité nástroje

Při vývoji a implementaci aplikace bylo použito několik nástrojů K těm nejdůležitějším patří aplikační rozhraní Apache Wicket, které pracuje nad bez stavovým protokolem HTTP, framework Hinernate, který umožňuje provádět objektové relační mapování a databázové dotazy s použitím SQL [8].

Pro nasazení aplikace byla použita cloudová služba Open Shift a systém pro správu verzi Git.

4.1. Apache Wicket

Apache wicket je jedním z nejznámějších webových aplikačních rozhraní, které zjednodušuje vývoj webové aplikace [9].

Wicket dovoluje vyvíjet webovou aplikaci objektově orientovaným programováním v jazyce. Snaží se řešit problémy bez stavového HTTP protokolu a stavového serverově orientovaného programování. Nabízí programový model, který skrývá skutečnost, že programátor pracuje nad bez stavovým protokolem. Vytvářet aplikace pomocí Wicketu z velké části vypadá jako programování v jazyce Java [10].

Obecně v programování má objekt nějaký stav a vlastnosti. Wicket obsluhuje stavy objektů, tak aby bylo možné se soustředit zejména na řešení problému aplikace namísto psaní zdlouhavých kódů [11].

Wcket striktně odděluje prezentační a aplikační vrstvu. Jde tedy o programování čistě v Javě nebo čistě v HTML. Nesnaží se poskytnout rozhraní, které by snížilo nebo dokonce eliminovalo samotné programování, naopak se snaží umožnit programování v Javě v co nejširší škále. Tím umožňuje také použití oblíbeného vývojového prostředí a Javy se všemi jeho přednostmi a výhodami.

Aplikační vrstva představuje pouze kód v Javě. Prezentační vrstva je tvořena statickými HTML šablonami, které nedovolují programátorovi směšování těchto dvou vrstev. HTML šablony jsou čistě statickým kódem doplněným o speciální Wicketové značky a identifikátory [9]. Striktní oddělení aplikační a prezentační vrstvy by mělo vést k zlepšení čitelnosti kódu, týmové spolupráci, využití

nástrojů vývojového prostředí jako refaktorování (refactoring), navigace v kódu, zvýraznění syntaxe, automatické doplnění kódu, apod.

4.1.1. Ukázka použití Wicketu

V ukázce je představena jednoduchá stránka zobrazující text „Ahoj světe!“. Každá stránka se ve Wicketu skládá ze dvou souborů. První s HTML kódem a druhý s Java třídou. Oba soubory musejí být ve stejném balíku, mají stejný název, ale rozdílnou koncovku:

```
cz/muni/fi/wicket/PrvniStranka.html
cz/muni/fi/wicket/PrvniStranka.java
```

Ukázka kódu č. 1 Obsah příkladového balíku.

Soubor PrvniStranka.java obsahuje velmi jednoduchý kód. Pro zobrazení textu použijeme komponentu Label, která má v konstruktoru identifikátor a text, který se má zobrazit. Identifikátor slouží k propojení prezentační a aplikační vrstvy.

```
package cz.muni.fi.wicket;
import org.apache.wicket.markup.html.WebPage;
import org.apache.wicket.markup.html.basic.Label;

public class PrvniStranka extends WebPage {

     public PrvniStranka () {
       add(new Label("message",
                     " Ahoj světe!"));
     }
}
```

Ukázka kódu č. 2 Aplikační část stránky zobrazující komponentu Label.

Soubor PrvniStranka.html obsahuje HTML značky. Značka `<h1>` je identifikátorem spojena s komponentou Label. Obsah značky bude nahrazen textem z komponenty.

```
<html>
<body>
   <h1 wicket:id="message">[zobrazovaný text...]</h1>
</body>
</html>
```

Ukázka kódu č. 3 Prezentační část stránky zobrazující komponentu Label.

Výsledný kód vytvořený pomocí Wicketu bude vypadat následovně:

```
<html>
<body>
    <h1>Ahoj světe!</h1>
</body>
</html>
```

Ukázka kódu č. 4 Výsledký vygenerovaný HTML kód.

4.2. Hibernate

Hibernate je framework pro jazyk Java, který nabízí objektově relační mapování. Umožňuje nahlížet na relační databáze jako na objektově orientovaný model. Hibernate mapuje objekty v Javě na entity v relační databázi. Patří k oblíbeným frameworkům v Javě [11].

Jeho použití v aplikaci šetří čas při vytváření objektů z výsledků databázových dotazů a přenášení vztahů mezi objekty do relačního modelu.

Hibernate je licencován pod GNU licencí a je uživatelům k dispozici zdarma.

4.3. OpenShift

OpenShift patří mezí cloudovou službu. Dalšími známými službami jsou Google App Engine od Googlu, AWS od Amazonu, MS Azure od Microsoftu [12]. OpenShift je platformou[6] pro cloudový

[6] Pracovní prostředí z hardwarové i softwarové stránky

hosting poskytovaný společností Rad Hat a patří do kategorie služeb PaaS (Platform as a Service), platforma jako služba. Podporuje celou škálu serverových technologií jako JavaScript, Ruby, Python, PHP, Perl, Java, databáze MySQL, PostgreSQL, MonogoDB a několik rámců. Z technologií Javy nabízí aplikační server Tomcat ve verzi 6 a 7, JBoss aplkační server ve verzi 7, JBoss aplikační platformu verze 6 [13]. Pro aplikaci byl zvolen pro svoji jednoduchost, cenovou dostupnost a při dodržení omezené prostorové kapacity užívání nemá omezení.

Pro potřebu práce je OpenShift vhodné místo kam lze aplikaci nahrát a dále bezplatně provozovat. Uživateli umožňuje pod jedním účtem nahrát a provozovat až 3 aplikace současně.

Aplikace využívá JBoss Application Server 7, MySQL 5.1 a pro správu databáze phpMyAdmin 4.0. Aplikaci lze nahrát na OpenShift za pomoci systému pro správu verzí Git.

4.4. Git a GitHub

Git je jedním ze systém pro správu verzí, který umožňuje uchovávat historie provedených změn ve zdrojovém kódu aplikace [14]. Repozitář v gitu je možné získat vložením již existujícího projektu, nebo klonováním jiného projektu. Klonování repozitáře je využívánou službou OpenShift.

GitHub patří mezi nejznámější servery pro hosting projektů verzovaných v Gitu. GitHub nabízí jak placené repozitáře, tak repozitáře neplacené pro volně rozšiřitelné aplikace.

5. Aplikace

Aplikace je rozdělena do šesti balíků, které obsahují jednotlivé programové třídy. V prvním balíku jsou obsaženy třídy reprezentující stránky s jednotlivými cvičeními a třídy obsluhující cvičení a protokoly o absolvování cvičení. Lekce a jejich správa je umístěna v druhém balíku. Třetí balík se stará o komunikaci s databází a objektově relační mapování. Validační procesy spouštěné při vyhodnocování cvičení jsou umístěny spolu s obsluhou chyb ve čtvrtém balíku. Registrace uživatelů, protokoly o absolvování cvičení a značky jsou umístěny v balíku pro obsluhu uživatelů. Poslední balík obsahuje základní stránku, panely pro učitele a studenty, panel pro přihlášení a menu panel.

5.1. Základní stránka

Aplikace umožňuje autorizaci a autentizaci uživatelů, proto musí rozšiřovat třídu `AuthenticatedWebApplication`, která podporuje autentizaci založenou na rolích a rozšiřuje třídu `WebApplication`, která je podtřídou `Application` asociovanou s instancí WicketServletu pro obsluhu stránek přes HTTP protokol.

Vlastní aplikace je tvořena třídou `HomePage` rozšiřující abstraktní třídu `BasePage`. Třída `BasePage` je rozšířením třídy `Page` z balíku org.apache.wicket.markup.html.

Nejen hlavní stránka – HomePage, ale také všechny ostatní stránky aplikace rozšiřují třídu BasePage.

Třídě BasePage jsou v konstruktoru přiřazeny následující komponenty:

- HeaderPanel – hlavička stránky
- MenuPanel – základní menu
- TeacherPanel – menu pro uživatele s rolí učitel
- StudentPanel – menu pro uživatele s rolí student
- SignPanel – panel pro přihlašování a odhlašování uživatelů
- FooterPanel – patička stránky

5.1.1. HeaderPanel

HeaderPanel je rozšířením komponenty Panel a reprezentuje hlavičku stránky, obsahující pouze její titulek, který funguje jako navigační prvek a odkazuje návštěvníky na titulní stránku aplikace.

5.1.2. MenuPanel

MenuPanel je obsáhlejším rozšířením komponenty Panel, které obsahuje základní menu celé aplikace. Menu je rozděleno do dvou kategorií:

- menu
- lekce

Kategorie menu obsahuje odkaz na stránku se stručným popisem aplikace. Kategorie lekce odkazuje již na konkrétní lekce a jejich cvičení.

MenuPanel je přístupný jak přihlášeným, tak nepřihlášeným uživatelům a umožňuje tak seznámit se s problematickou značkovacích jazyků každému kdo má k aplikaci přístup.

5.1.3. TeacherPanel

TeacherPanel je komponenta, která je určena výhradně pro uživatele s oprávněním učitel. Učitel ze své role má možnost přidávat nová cvičení, registrační období, nahlížet do seznamu studentů a protokolů o splnění cvičení. Panel slouží jako rozcestník s odkazy na stránky určené učiteli.

Obr. č. 1 Ukázka komponenty TeacherPanel.

Komponenta je uživateli poskytnuta v závislosti na jeho oprávněním. Uživatelům s oprávněním učitel je komponenta

zobrazena, ostatní uživatelé se o její existenci nedozví. Viditelnost komponenty je zajištěna překrytím metody `boolean isVisible()`, která získá relaci přihlášeného uživatele a ověří, zda se jedná o učitele.

5.1.4. StudentPanel

Komponenta StudentPanel je svojí funkcí podobná komponentě TeacherPanel, ale je určena pro uživatele s rolí student. Obsahuje odkaz vedoucí k vlastním protokolům o splnění cvičení.

Stejně jako u komponenty TeacherPanel je uživateli poskytnuta v závislosti na jeho oprávněním. Překrytím metody `boolean isVisible()`, je zajištěna viditelnost komponenty pouze pro uživatele s oprávněním student.

5.1.5. SignPanel

SignPanel je komponenta, která souží k přihlašování a odhlašování uživatelů. Je rozdělena na dvě části. Nepřihlášenému uživateli nabízí přihlašovací formulář, případně vytvoření nové registrace. Přihlášený uživatel vidí své základní informace a odkaz pro odhlášení.

Viditelnost jednotlivých částí je opět zajištěna překrytím metody `boolean isVisible()`, která ověří stav přihlášení uživatele metodou `boolean isSignedIn()`. Metoda vrací `true` pro přihlášeného uživatele, `false` v opačném případě.

Obr. č. 2 Ukázka SignPanelu před přihlášením (vlevo) a po přihlášení (vpravo)

5.1.6. FooterPanel

FooterPanel je jednoduché rozšíření třídy Panel, které reprezentuje patičku stránky.

5.2. Autorizace a autentizace

Autorizace uživatelů, jako proces získání souhlasu s provedením nějaké operace [15], probíhá v aplikaci na dvou úrovních:

1. autorizace aplikací a
2. autorizace učitelem.

Autentizace probíhá pomocí standardního přihlašovacího formuláře.

5.2.1. Autorizace aplikací

Podpora autorizace na straně aplikace je ve Wicketu řešena konceptem autorizační strategie pomocí rozhraní `Iauthorization Strategy` (v balíku org.apache.wicket.authorization). Toto rozhraní definuje dvě metody, které ověřují zda je uživatel oprávněn vytvářet instance dané komponenty a zda je oprávněn na dané instanci komponenty provádět operace [16].

Standardní implementace autorizačního rozhraní dovoluje každému uživateli vytvářet instance z každé komponenty a provádět na ni všechny dostupné operace. Autorizační strategie může být změněna v průběhu inicializace aplikace.

V aplikaci je použita autorizace na základě rolí a k ní příslušná strategie `RoleAuthorizationStrategy` z balíku org.apache.wicket. authroles.authorization.strategies.role. Při inicializaci aplikace je vytvořena nová instance autorizační strategie s vlastní třídou pro získání role z relace (třída `UserRolesAuthorizer`).

Rolově orientovaná strategie se opírá o dvě vestavěné anotace. Těmi jsou `AuthorizeInstantiation` a `AuthorizeAction`. Anotace `AuthorizeInstantiation` specifikuje, která role smí vytvářet instance, zatímco anotace `AuthorizeAction` specifikuje která role smí používat danou metodu komponenty [16].

Pro autorizaci uživatelů se v aplikace používá anotace `AuthorizeInstantiation`. Pro autorizaci uživatelů s rolí učitel jde o anotaci `@AuthorizeInstantiation("TEACHER")`.

5.2.2. Autorizace učitelem

Přístup do některých částí aplikace je podmíněn přihlášením. Každý uživatel, který chce získat přístup do aplikace se musí nejprve registrovat. Při registraci musí projevit znalost registračního hesla, které je vytvářeno učitelem a má omezenou platnost. Poskytnutím hesla učitel opravňuje uživatele k registraci do aplikace.

5.2.3. Autentizace

Autentizace jako proces ověření proklamované identity uživatele je v aplikaci implementován pomocí přihlašovacího formuláře, kde uživatel zadává své uživatelské jméno a heslo.

Pro jednoduchou autentizaci uživatelů pomocí formuláře nabízí Wicket komponentu `SignInPanel` z balíku org.apache.wicket. authroles.authentication.panel. Komponenta je složena z přihlašovacího formuláře a informačního panelu pro chybovou nebo informační zprávu – `FeedbackPanel`.

`FeedbackPanel` podporuje filtry pro zprávy, aby bylo možné separovat zprávy pouze z určitého formuláře. K tomu lze použít `ContainerFeedbackMessageFilter`.

Již zmiňovaná komponenta `SignInPanel` není implementována tak, aby bylo možné zadat filtr. Proto bylo nutné implementovat vlastní přihlašovací panel z důvodu kolize více FeedbackPanelů dalších formulářů. Aplikace na jedné stránce zobrazuje více různých formulářů, které mají vlastní FeedbackPanel.

Samotná autentizace probíhá metodou `boolean authenticate(String username, String password)` třídy SignInSession, která obsluhuje přihlašovací relaci. Aplikace používá vlastní třídu - SignInSession, která se stará o relaci přihlášení a přepisuje metody `Roles getRoles()`, `void signOut()` a `boolean authenticate(…)`.

V aplikaci není možné najít přímé volání metody authenticate. Ta je volána Wicketem při vytváření nové relace přihlášení.

5.3. Role a oprávnění

V aplikaci jsou použity dvě úrovně oprávnění: student a učitel. Role uživatelů jsou stejnojmenné podle úrovně oprávnění. Způsob získání oprávnění a jeho kontrola jsou popsány v kapitole 5.2 Autorizace a autentizace. V této kapitole je popsáno, jaké může uživatel vykonávat akce s příslušnou rolí.

Nezávisle na roli je dovoleno všem uživatelům:

- prohlížet všechny typy lekcí a jejich obsah,
- prohlížet jednotlivá cvičení (ne je absolvovat),
- zaregistrovat se do aplikace.

5.3.1. Role student

Uživatel, kterému je přiřazena role student může provádět tyto akce:

- přihlásit a odhlásit se z aplikace,
- absolvovat všechny typy cvičení,
- nahlížet do svých protokolů o absolvování svých cvičení a
- nahlížet do komentářů, které do protokolu o cvičení zadal učitel.

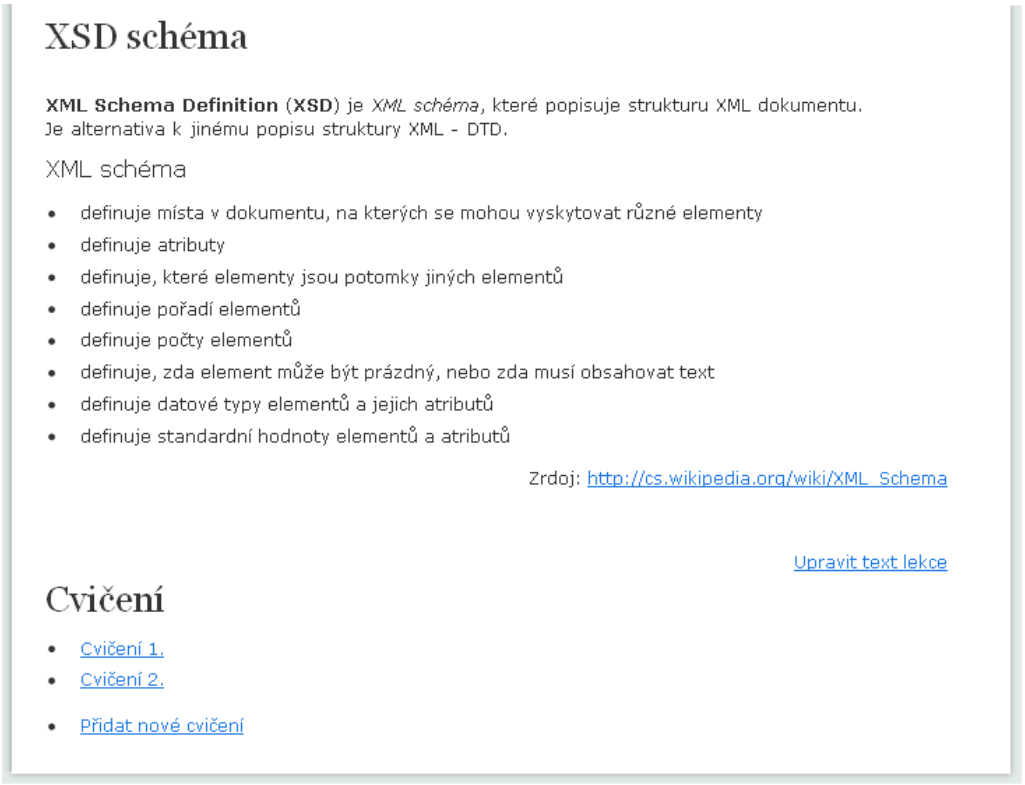
5.3.2. Role učitel

Role učitel má dovoleno provádět více akcí. V roli učitele může uživatel provádět všechny následující akce:

- přihlásit a odhlásit se z aplikace,
- absolvovat všechny typy cvičení,
- nahlížet do všech protokolů o absolvování cvičení všech uživatelů,
- přidat a měnit komentáře do protokolu o absolvování cvičení,
- přidat nové cvičení a zobrazit nebo skrýt jeho vzorové řešení,
- nahlížet do existujících registračních období a
- přidat nové registrační období.

5.4. Lekce a cvičení

Hlavním záměrem aplikace je umožnit uživatelům procvičovat problematiku předmětu PB138 Moderní značkovací jazyky a jejich aplikace v rozsahu jeho sylabu. Základní jednotkou je lekce, do které je možné přidat nebo upravovat text a sdružovat cvičení daného tématu. Ukázková lekce je zobrazena na Obr. č. 3.



Obr. č. 3 Ukázka lekce se vzorovým textem, dvěmi přidanými cvičeními a odkazy pro přidání nového cvičení a úpravu textu lekce.

Cvičení dovoluje automatizovaně procvičovat danou problematiku. Aplikace umí automatizovaně procvičovat:

- ověření dokumentu podle DTD,
- XSLT transformace,
- XSD schéma,
- xQuery výrazy a
- xPath výrazy.

Každé cvičení je složeno ze zadání, částečného řešení a vzorového řešení. Proto bylo možné navrhnout společnou třídu

`Exercise` z balíku `cz.muni.fi.exercise`, která obsahuje všechny společné prvky. Cvičení jsou zaměřena na různou problematiku a není možné je vyhodnocovat stejně, proto bylo zapotřebí implementovat vlastní vyhodnocovací mechanizmy jednotlivých cvičení. U vyhodnocovacích mechanizmů se podařilo identifikovat společné prvky a ty částečně implementovat ve třídě `AProcessor`. Jde o abstraktní třídu, která obsahuje několik abstraktních metod, které jednotlivé procesory implementují.

Obsah cvičení je rozdělen na veřejnou a neveřejnou část. Do veřejné části, kterou vidí všichni uživatelé patří zadání cvičení a jeho vzorové řešení (pokud je učitel povolil zobrazit). Po přihlášení do aplikace je uživateli přístupný navíc formulář, který vyhodnotí jeho řešení cvičení a každé řešení je uloženo v podobě protokolu o absolvování cvičení. Nepřihlášený uživatel má možnost si cvičení prohlédnout, ale nemůže vyhodnotit své řešení, z důvodu aby mohl učitel nebo student nahlížet do všech řešení.

5.4.1. Ověření dokumentu podle DTD

Záměrem cvičení na ověření dokumentu podle DTD je umožnit studentům procvičovat vytváření DTD k zadanému XML. Cvičení má zadané XML ke kterému studenti mají za úkol vytvořit vlastní DTD nebo doplnit částečně zadané řešení.

K ověření dokumentu XML podle DTD bylo vyžadováno použití validačního nástroje Xerces. Pro Javu existuje knihovna s implementací funkcionalit Xercesu. Do aplikace byla přidána jako závislost (dependency). Knihovna je použita ve verzi 2.11.0 z balíku org.apache.xerces.

V průběhu validace je zadané XML v podobě textového řetězce převedeno instancí třídy `TransformerFactory` na `StreamResult`. Transformátoru je také nastaven `DOCTYPE_SYSTEM` v podobě DTD, které student vytvořil. XML dokument s nastaveným DTD v podobě StreamResult je analyzován instancí třídy DOMParser, z knihovny Xerces a metodou `parse(InputSource is)` je ověřena validita DTD dokumentu, který student vytvořil. Při analýze a zpracování DTD

dokumentu mohou vzniknout tři druhy chyb, které jsou obslouženy a vráceny v podobě zprávy pro uživatele.

V případě, že při analýze nevznikla žádná chyba je DTD vyhodnoceno jako správné.

Pro obsluhu chybových hlášení byla implementována třída `ErrorHandler` rozšiřující standardní `DefaultHandler` z balíku org.xml.sax.helpers. Programovány byly metody pro obsluhu varování (warnings), chyb (errors) a závažných chyb (fatal errors). Ani v případě závažné chyby nedojde k pádu aplikace, ale student dostane hlášení o závažnosti.

5.4.2. XSLT transformace

Cvičení na XSLT transformace umožňuje studentům vytvářet vlastní nebo doplňovat částečná řešení XSLT transformací. Zadání se skládá z XML dokumentu a slovního popisu toho, co má transformace provést. Po vyhodnocení cvičení má student přístup k výstupnímu XML dokumentu. Vidí tedy jak dopadla transformace, kterou navrhl.

Transformace je provedena pomocí instance třídy `TransformerFactory` z balíku `javax.xml.transform`, který představuje rozhraní transformační procesy. Použitá metoda `transform(Source s, Result r)` převede zdroj `s` na výsledek `r`. Transformátoru je nastaven vlastní `ErrorListener` implementující rozhraní `ErrorListener` ze stejného balíku.

Transformátoru je možné nastavit specifické vlastnosti, jako například odsazování elementů pro lepší orientaci v XML dokumentu, čehož je při transformaci využito.

Během transformace mohou vzniknout chyby. Třída `ErrorListener` zajišťuje sdělení chyby uživateli, tak aby mohl transformaci opravit. Mohou vzniknout tři typy chyb: varování (warning), chyba (error) nebo závažná chyba (fatal error). O každé z chyb je uřivatel informován v podobě zprávy.

5.4.3. XSD Schéma

Ve cvičeních zaměřených na XSD schéma má student za úkol napsat XSD schéma k zadanému XML.

K vyhodnocení uživatelem vytvořeného XSD schématu je použita instance třídy `SchemaFactory` z balíku javax.xml.validation. Při vytváření instance třídy `SchemaFactory` je nutné zadat jmenný prostr popisující schéma. Vybrán byl jmenný prostor uveřejněný na webu mezinárodního konsorcia World Wide Web Consortium (W3C). Jmenný prostor je dostupný online na adrese: http://www.w3.org/2001/XMLSchema.

Z textové podoby XSD Schématu byla vytvořena instance třídy `Schema`. Z této instance byl metodou `newValidator()` vytvořen validátor, který vyhodnotí správnost XSD schématu. Pro obsluhu chybových hlášení byla implementována třída `ErrorHandler`, která je použita také k obsluze chyb při validaci proti DTD.

5.4.4. xQuery výrazy

XQuery výrazy je možné procvičovat v dalším typu cvičení. Student má zadané XML a jeho úkolem je vytvořit xQuery výraz, který bude odpovídat zadání.

Ke zpracování a vyhodnocení vytvořeného xQuery výrazu byla použit knihovna Saxon ve verzi 7.8 a převážně třídy z balíku net.sf.saxon.query.

Prvním krokem k vyhodnocení xQuery výrazu je vytvoření konfiguračního objektu `net.sf.saxon.Configuration`, který uchovává nastavení systému. Objektu není potřeba dodatečně nastavovat žádné vlastnosti, základní nastavení je dostačující. Dalším krokem je zpracování vstupního XML třídou `StaticQueryContext`, která zkompiluje XML a vytvoří z něj výraz `XqueryExpression`. Z xQuery výrazu bylo nutné vytvořit instanci třídy DocumentInfo a nastavit ji jako kontextový uzel třídy `DynamicQueryContext`. Spuštění xQuery výrazu nad XML se provádí metodou `run(DynamicQueryContext dqc, Result result, Properties p)`. Případné chyby při vykonávání metody jsou vráceny prostřednictvím obsluhy chyb – třídou `ErrorListener`.

5.4.5. xPath výrazy

Poslední typ cvičení studentům umožňuje procvičovat vytváření xPath výrazů. Student má za úkol napsat takový xPath výraz, aby odpovídal zadání úlohy. Součástí zadání je také XML, které se vztahuje k xPath výrazu. Student vidí XML před i po zpracování. Nedostává pouze zprávu o tom, zda je xPath výraz správně nebo ne, ale má možnost vidět jak XML dopadlo po aplikaci jeho výrazu.

Zpracování xPath výrazu se děje za pomoci rozhraní `XPath` balíku `javax.xml.xpath` a instance třídy `TransformerFactory` a transformátoru `Transformer`, který metodou `transform(Source s, Result r)` převede zdroj `s` na výsledek `r`. Chyby vzniklé při transformaci jsou vráceny prostřednictvím třídy `ErrorListener`.

Zdrojem transformace je objekt typu `DOMSource`. Vstupní XML v textové podobě je převedeno na objekt typu DOMSource z knihovny Xerces.

Samotné zpracování xPath výrazu probíhá nad instancí třídy `XpathFactory` z balíku javax.xml.xpath. Metoda `evauate(String expression, InputSource source, Qname returnType)` vyhodnotí xPath výraz `expression` vzhledem ke zdroji `source` a vrátí výsledek jako specifický typ `returnType`.

5.5. Protokol o absolvování cvičení

Protokol o absolvování cvičení je navržen tak, aby umožnil prohlížet řešení absolvovaných cvičení a dovoluje učiteli dát zpětnou vazbu studentovi prostřednictvím komentáře k výsledku.

Entita `UserExerciseProtocol` uchovává informace o výsledku, uživateli který cvičení absolvoval, cvičení, které bylo absolvováno, čas absolvování a komentář učitele. Z důvodu různých typů cvičení je výsledek uložen pouze v textové podobě. Obr. č. 4 ilustruje podobu protokolu prohlíženou učitelem.

Učitel může komentovat výsledek studenta prostřednictvím komponenty `AjaxEditableLabel`, která umožňuj upravit obsah komentáře přímo na stránce s protokolem bez nutnosti znovunačtení stránky. Komentář je dovoleno vkládat pouze uživatelům s oprávněním učitel. Ostatní uživatelé mají právo do komentáře pouze nahlížet.

Student

Jiří Krejčí (jirka)

Datum a čas uložnení výsledku

2013-12-21 17:25:37

Cvičení

Pro níže uvedené XML dopište správné DTD:

```
<?xml version="1.0" encoding="UTF-8" ?>

<bookshell>
  <books year="2000">
    <book>
      <author>J. K. Rowling</author>
      <title>Harry Potter</title>
      <pages>1000</pages>
    </book>
  </books>
  <books year="1980">
    <book>
      <author>O. G. Wells</author>
      <title>War of the worlds</title>
      <pages>876</pages>
    </book>
  </books>
</bookshell>
```

Výsledek

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT bookshell (books)+>

<!ELEMENT books (book)+>
<!ATTLIST books year CDATA #REQUIRED>

<!ELEMENT book (author+,title,pages)>

<!ELEMENT author (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT pages (#PCDATA)>
```

Komentář učitele

Obr. č. 4 Ukázka protokolu o absolvování cvičení prohlížená učitelem.

5.6. Zvýraznění syntaxe XML

Součástí každého cvičení je vždy zadané XML a vzorové řešení v podobě XML. Aby bylo XML lépe čitelnější, je v aplikaci využito zvýraznění syntaxe, které barevně odděluje logické části, např. název atributu a jeho hodnotu nebo název značky a název atributu.

Jak ilustruje Obr. č. 5, syntaxe není zvýrazněna pouze barevně, ale také prostorově. Dochází k odsazování vnořených značek, tak aby vznikal na první pohled strukturovaný XML dokument.

Pro zvýraznění syntaxe XML doumentů je v aplikaci je využito externí JavaScriptové knihovny jQuery a její rozšíření jQuery Syntax Highlighter ve verzi 1.1. Rozšíření umožňuje programátorovi jednoduše zvýraznit syntaxi XML dokumentů. Práce se zvýrazněným dokumentem je pak pro uživatele přívětivější a může pomoci k lepší orientaci v dokumentu.

Jak knihovna jQuery tak její rozšíření jsou licencovány pod licencí MIT.

Knihovna i rozšíření byly vybrány z důvodu jejich snadné aplikace a volně rozšiřitelné licenci.

```
<?xml version="1.0" encoding="utf-8" ?>
<xsl:stylesheet version="1.0"
     xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <myFavoriteBooks>
      <xsl:for-each select="bookshell/book">
        <book>
          <author><xsl:value-of select="author"/></author>
          <title><xsl:value-of select="title"/></title>
        </book>
      </xsl:for-each>
    </myFavoriteBooks>
  </xsl:template>
  <xsl:output method="xml" encoding="utf-8" />
</xsl:stylesheet>
```

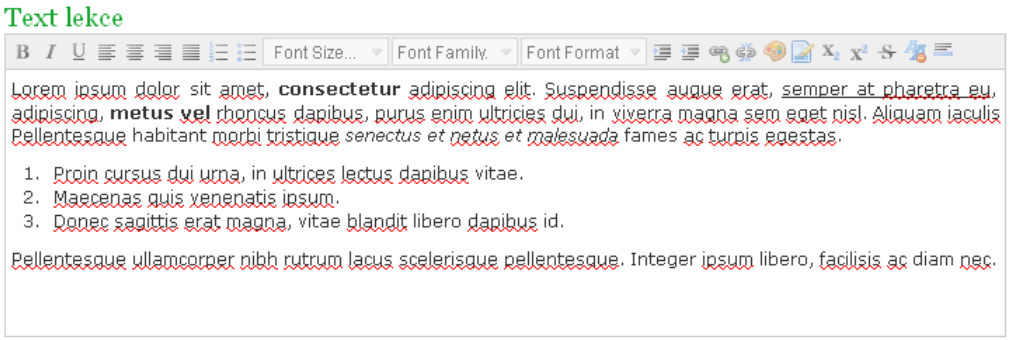
Obr. č. 5 Ukázka zvýraznění syntaxe.

5.7. WYSIWYG editor

Aplikace dovoluje učiteli editovat text lekce a přidávat nová cvičení. Aby bylo možné jednoduše formátovat přidávaný text přímo v prohlížeči, je v aplikaci použit WYSIWYG editor NicEdit ve verzi 0.9, který je licencován pod licencí MIT.

Editor byl vybrán z důvodu snadné implementace, možnosti zvolit, jaké formátovací prvky budou uživateli poskytnuty a které budou zakázány. Ze základní nabídky formátování, které editor nabízí je v aplikaci odebráno pouze tlačítko pro vkládání obrázků. Editor uživateli nabízí možnost zarovnání textu, změnu velikosti textu, změnu fontu písma, odsazení, vytvoření číslovaných nebo odrážkových seznamů, apod. Obr. č. 6 demonstruje použití editoru nad vzorový textem.

Strukturovaný a formátovaný text lekce vytvořený pomocí WYSIWYG editoru NicEdit je pro studenty lepé čitelný, umožňuje snazší orientaci a dovoluje také odkazovat na externí zdroje.



Obr. č. 6 Ukázka použití editoru na úpravu textu lekce se vzorovým textem.

6. Návrh použití aplikace ve výuce

Aplikace je navržena tak, aby se mohla používat jako podpora výuky předmětu PB138 Moderní značkovací jazyky a jejich aplikace. Při skutečném využití získají vyučující předmětu soubor cvičení k jednotlivým tématům, které mohou jednoduše rozšiřovat nebo doplňovat. Studenti tím mají možnost procvičit danou problematiku větším množstvím cvičení různé obtížnosti a také získat zpětnou vazbu od cvičícího.

Po spuštění aplikace musí cvičící nejprve vytvořit teoretický obsah lekce a přidat k němu cvičení. Bez těchto dvou základních kroků je zbytečné udělovat přístup do aplikace studentům.

Teoretický obsah lekce může upravovat pouze uživatel s oprávněním učitel na stránce s danou lekcí kliknutí na tlačítko „Upravit text lekce“. WYSIWYG editor umožňuje text lekce strukturovat a formátovat tak aby byl co nejpřehlednější. Součástí každé lekce je soubor číslovaných cvičení uvedený pod textem lekce. Uživatel s oprávněním učitel může také přidávat nová cvičení. Každé cvičení se skládá z typu, textového zadání a XML, které tvoří nedílnou součást zadání. Dále je možné zadat částečné řešení cvičení. Důležitou součástí cvičení je očekávaný XML výstup. Nejde o vzorové řešení, ale o výstupní XML po transformaci nebo aplikaci dotazu. Cvičení také obsahuje vzorové řešení, jak v textové tak v XML podobě, které lze zobrazit nebo skrýt. Na všechny XML bude aplikováno zvýraznění syntaxe pro lepší přehlednost. Uložením cvičení vznikne nová položka v seznamu cvičení dané lekce a je možné je ihned absolvovat.

Nástroje vyhodnocující řešení cvičení jsou běžně využívané ve výuce. Jejich použití není závislé pouze na aplikaci a student je tak může aplikovat sám např. prostřednictvím příkazové řádky.

Přístup uživatelů je podmíněn registrací. Registrace je povolena pouze v registračním období a je podmíněna heslem. Registrační období má právo přidat pouze uživatel s oprávněním učitel.

Takový uživatel musí zadat registrační období (odkaz „Registrační období“ v menu) pro povolení registrace. Při zadávání nového období je zapotřebí stanovit kdy začíná, kdy končí a zvolit

vhodné registrační heslo. Registrační heslo slouží k označení uživatelů zaregistrovaných v daném období a slouží také jako filtr v seznamu uživatelů. Vhodným heslem je např. 05jaro2013 pro studenty páté seminární skupiny v semestru jaro 2013.

Při zadávání nového období je důležité, aby cvičící zadal oprávnění „Student“. Po zadání registračního období může sdělit studenům adresu, na které se zaregistrují. Pokud chce uživatel umožnit registraci učiteli, je potřeba zadat oprávnění „Učitel“ a období registrace co nejvíce zkrátit, aby nemohlo dojít k neoprávněné registraci.

Studenti se vyplněním registračních údajů zaregistrují a získávají okamžitý přístup ke všem lekcím a jejich cvičení. Zadání cvičení je možné prohlížet i jako nepřihlášený uživatel, absolvování cvičení je ale přihlášením podmíněné. Cvičení jsou dostupná v každé lekci pod jejím teoretickým obsahem.

Po zadání vlastního řešení a jeho vyhodnocením kliknutím na tlačítko „Vyhodnotit“ vidí uživatel okamžitě svůj výsledek. Počet vyhodnocení každého cvičení není omezen. Při každém vyhodnocení dochází k uložení protokolu o absolvování cvičení, který obsahuje informace o tom, jaké cvičení který uživatel absolvoval a jaké bylo jeho řešení. Protokol lze zpětně prohlížet (odkaz „Protokoly o cvičení“ při oprávnění Student, „Seznam uživatelů“ při oprávnění Učitel.

Protokoly jsou zde řazeny podle typu lekce do které patří. Jakmile studenti absolvují cvičení, může si cvičící dohledat protokol o absolvování cvičení v Seznamu uživatelů a přidat svůj komentář k řešení. K zadávání komentáře je využito AJAX komponenty, cvičící jej tedy zadává přímo do protokolu. Dvojklikem na položku komentář (v případě neexistujícího komentáře se zobrazuje text „…“) se zobrazí editační textové pole pro zadání obsahu.

Odkaz „Seznam uživatelů“ učiteli dává přehled o všech zaregistrovaných uživatelích a umožňuje mu tento seznam filtrovat podle značky a oprávnění, pokud chce zobrazit pouze danou skupinou uživatelů. U každého uživatele je možné prohlížet jeho protokoly a absolvování cvičení. Zeleně zbarvené odkazy na protokoly s nápisem „OK“ značí, že zadané řešení vyhovuj zadání, oproti tomu červené

zbarvené odkazy s nápisem „NOK“ značí nevyhovující nebo žádné řešení cvičení. Každý protokol může učitel prohlížet a komentovat.

7. Závěr

Cílem této práce bylo navrhnout a implementovat aplikaci, která umožní automatizovaně procvičovat problematiku značkovacích jazyků v rozsahu předmětu PB138 Moderní značkovací jazyky a jejich aplikace.

Na základě zvolené SCRUM metodiky byly stanoveny požadavky, které se podařilo všechny implementovat. Vznikl tak nástroj, který umožňuje vytvářet a rozšiřovat databanku cvičení k podpoře výuky témat jako je validace DTD dokumentů, XML transformací, XSD schémat, xQuery a xPath výrazů.

Aplikace podporuje dvě úrovně oprávnění. Uživatelé s oprávněním student mají možnost pouze plnit cvičení, zatímco uživatelé s oprávněním učitel mohou nová cvičení přidávat, komentovat protokoly a absolvování cvičení nebo povolovat registraci uživatelům novým. Volba úrovní oprávnění je tak vhodná k použití ve výuce.

Aplikace byla nasazena na službu OpenShift a je volně přístupná.

8. Literatura

[1] VOCHOZKA JOSEF. *Značkovací jazyky a XML* [online]. 2000 [cit. 2013-11-29]. Dostupné z: <http://www.ics.muni.cz/zpravodaj/articles/201.html>.

[2] *PB138 Moderní značkovací jazyky a jejich aplikace* [online]. 2012 [cit. 2013-11-25]. Dostupné z: <https://is.muni.cz/auth/predmet/fi/jaro2012/ PB138>.

[3] Wicket: A simplified framework for building and testing dynamic Web pages. *IBM developerWorks* [online]. 2008 [cit. 2013-08-12]. Dostupné z: <http://www.ibm.com/developerworks/ web/library/wa-aj-wicket/?S_TACT=105AGY82&S_CMP=GENSITE>.

[4] SCHWABER KEN. *Agile Software Development with SCRUM*. Pearson 2008, 158 s. ISBN 978-0-1320-7489-6.

[5] SUTHERLAND JEFF, SCHWABER KEN. *The Scrum Guid ™* [online]. 2013 [cit. 2013-08-20] Dostupné z: <https://www.scrum.org/Portals/0/Documents/Scrum%20Guides/2013/Scrum-Guide.pdf>.

[6] PITNER TOMÁŠ. *Základní standardy a rozhraní rodiny XML* [online]. 2. 3. 2010. [cit. 2013-09-05]. Dostupné z: <https://is.muni.cz/auth/el/1433/jaro2012/PB138/um/apis/foil4.html>.

[7] PITNER TOMÁŠ. *Transformace XML dat* [online]. 22. 3. 2010. [cit. 2013-09-05]. Dostupné z: <https://is.muni.cz/auth/el/1433/jaro2012/PB138/um/xslt/foil1.html>.

[8] *Hibernate ORM* [online]. [cit. 2013-10-12]. Dostupné z: <http://hibernate.org/orm/>.

[9] VAYNBERG IGOR. *Apache Wicket Cookbook*. 1. vydání. Packt Publishing, 2011, 312 s. ISBN 978-1-849511-60-5.

[10] MARTIJN DASHORST, EELCO HILLENIUS. *Wicket in Action*. Greenwich: Mannig Publications Co., 2009, 392 s. ISBN 1-932394-98-2.

[11] BAUER CHRISTIAN, KING GAVIN. *Hibernate in Action*. Greenwich: Mannig Publications Co., 2005, 430 s. ISBN 1932394-15-X.

[12] Nová cloudová platforma OpenShift. *Živě.cz* [online]. 2011 [cit. 213-10-11]. Dostupné z: <http://linuxzblizka.blog.zive.cz/2011/05/nova-cloudova-platforma-openshift/>.

[13] Java Application Hosting. *OpenShift* [online]. 2013 [cit. 2013-10-23]. Dostupné z: <https://www.openshift.com/developers/java>.

[14] CHACON SCOTT. *Pro Git*. Praha: CZ.NIC, z. s. p. o., 2009, 263 s. ISBN: 978-80-904248-1-4.

[15] MATYÁŠ VÁCLAV, KRHOVJÁK JAN. *Autorizace elektronických transakcí a autentizace dat i uživatelů*. 1. vyd. Brno: Masarykova univerzita, 2008. 128 s. 1. ISBN 978-80-210-4556-9.

[16] 20 Security with Wicket - Reference Documentation. *Apache Wicket* [online]. 2012 [cit. 2013-12-05]. Dostupné z: < https://wicket.apache.org/guide/guide/chapter19.html#chapter19_2>.

9. Přílohy

Autor	Požadavky
Mgr. Filip Nguyen	Student může komentovat jednotlivé lekce, aby učitel měl zpětnou vazbu o tom, co přidal, že je chyba ve cvičení, apod.
Mgr. Filip Nguyen	Student může měnit obsah lekce. Aby nebylo možné "ničit" obsah, tak jakýkoliv obsah který se do systému dostane nebude nikdy smazán.
Mgr. Filip Nguyen	Měla by být možnost jednoduše přidat studenty určitého ročníku. Registrace na heslo, tj. učitel přijde do učebny, v systému zadá hodinové heslo na registraci. Systém nyní akceptuje registrace po dobu jedné hodiny na základě tohoto hesla.
Mgr. Filip Nguyen	Student má možnost odpovídat na otázky systému a/b/c/d
Mgr. Filip Nguyen	Student může řešit úkol typu: je zadáno XML a otázka je "Dopište DTD" (Doplnění stávajícího částečného řešení)
Mgr. Filip Nguyen	Student má možnost zpětné vazby při řešení cvičení XSLT transformací. (Tj. vidí XML před i po transformaci)
Mgr. Filip Nguyen	Když student validuje pomocí DTD ve cvičení, tak výstup validace bude ve formátu, který zprostředkovává XMLLint. Tzn. že technologicky se na pozadí spouští XMLLint.
Mgr. Filip Nguyen	Učitel může přidávat další cvičení.
Mgr. Filip Nguyen	Učitel může zobrazit statistiku nejčastěji vyhodnocovaných řešení a nejvíce problematických cvičení.

Mgr. Filip Nguyen	V lekci je více stránek. Aby učitel při výkladu mohl studentům říct, že budeme pracovat s další stránkou.
Jiří Krejčí	Učitel může zobrazit nebo skrýt vzorové řešení příkladu.
Jiří Krejčí	Mezi uživatelem a systémem probíhá automatická mailová komunikace (při registraci, změně hesla).
Jiří Krejčí	Učitel může přidávat další lekce.
Jiří Krejčí	Učitel může vytvářet testové otázky a/b/c/d.
Jiří Krejčí	Uživatel si může přidávat vlastní značky (tagy), aby si vytvořil supiny, např. tag '2011', 'skupina01'.
Jiří Krejčí	Učitel může přiřazovat značku (tag) studentovi, aby byl schopen filtrovat studenty.
Jiří Krejčí	Student může řešit úkol typu: je zadáno XML a otázka je "Napište XPath výraz, který vybere elementy s atributem 'id'."
Jiří Krejčí	Student může řešit úkol typu: je zadáno XML a otázka je "Dopište XML schéma" (Doplnění stávajícího částečného řešení)
Jiří Krejčí	Student může řešit úkol typu: je zadáno XML a otázka je "Dopište XSL transformaci, aby výsledný XML splňoval tyto podmínky:..." (Doplnění stávajícího částečného řešení)
Jiří Krejčí	Student může řešit úkol typu: je zadáno XML a otázka je "Napište XQuery výraz, který vybere elementy XY."
Mgr. Marek Grác	Cvičení v lekci jsou uspořádány nelineárně, aby pokročilí studenti mohli rychle přejít na složitější úlohy a slabší studenti si mohli lehčí příklady více osvojit.
Mgr. Marek Grác	Učitel může ke cvičení přidat předpokládaný čas

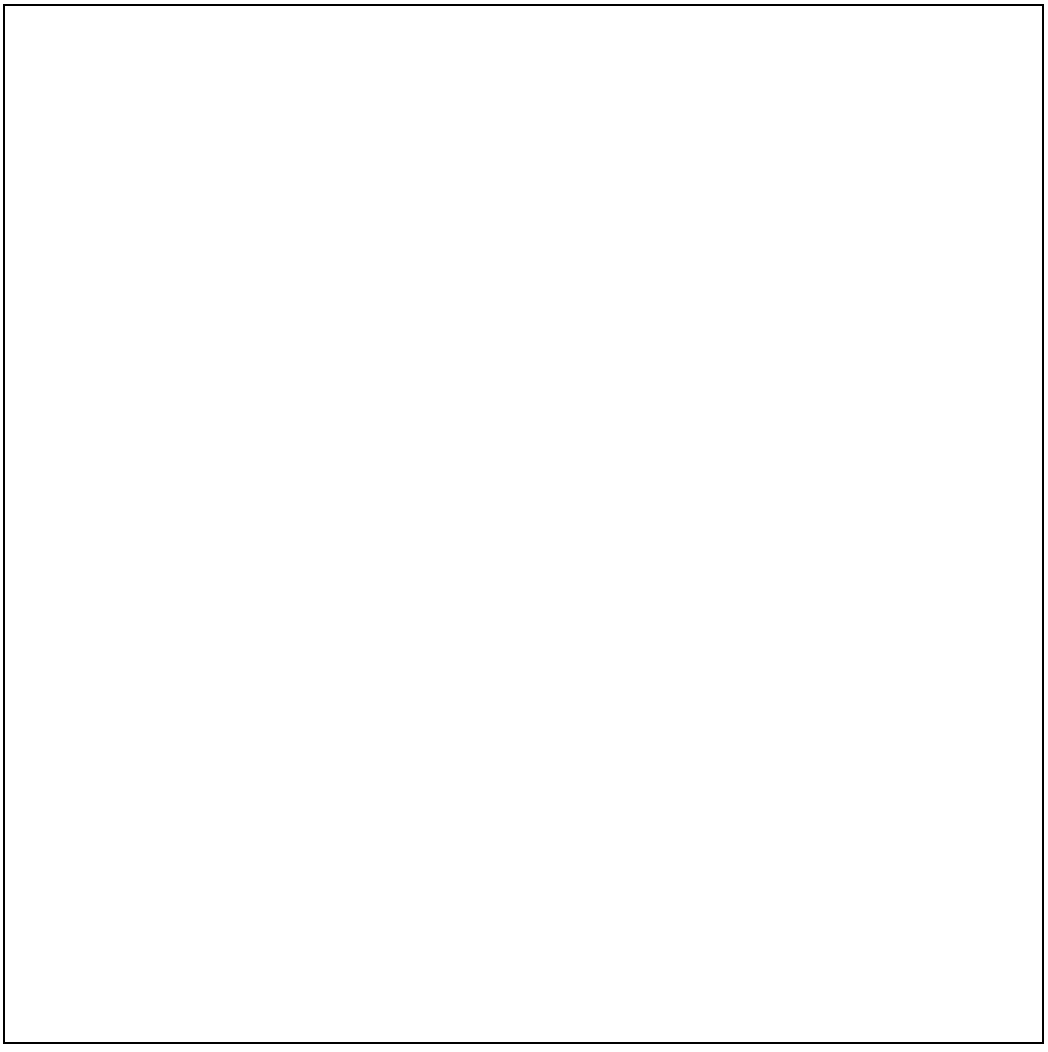
	potřebný k zvládnutí úlohy, aby studenty motivoval úlohu dokončit v nějakém čase.
doc. RNDr. Tomáš Pitner, Ph.D.	Student může řešit úkoly typu: vytvořte dokumentaci v DocBook. Dokumentace bude vložena jako volný text a učitel ji bude mít možnost prohlížet.
doc. RNDr. Tomáš Pitner, Ph.D.	Student může řešit úkoly typu: vytvořte dokument v HTML5 vložte protokol z online validátoru. Protokol bude vložen jako volný text a učitel bude mít možnost jej prohlížet.
doc. RNDr. Tomáš Pitner, Ph.D.	Student může programově spouštět xPath a xQuery doplněním programové spuštění xPath, XQuery, apod. doplňování 1 metody.
Mgr. Luděk Bártek, Ph.D.	Když student validuje pomocí DTD ve cvičení, tak výstup validace bude ve formátu, který zprostředkovává validační nástroj Xerces.

Tab. 1 Seznam požadavků a jejich autoři.

Student může řešit úkol typu: je zadáno XML a otázka je "Dopište DTD" (Doplnění stávajícího částečného řešení).
Student může řešit úkol typu: je zadáno XML a otázka je "Dopište XSL transformaci, aby výsledný XML splňoval tyto podmínky: ..." (Doplnění stávajícího částečného řešení.)
Student může řešit úkol typu: je zadáno XML a otázka je "Nepiště XPath výraz, který vybere elementy s atributem 'id'."
Student může řešit úkol typu: je zadáno XML a otázka je "Dopište XML schéma" (Doplnění stávajícího částečného řešení)
Student může řešit úkol typu: je zadáno XML a otázka je "Nepiště XQuery výraz, který vybere elementy XY."
Student má možnost zpětné vazby při řešení cvičení XSLT transformací. (Tj. vidí XML před i po transformaci)
Učitel může přidávat další cvičení.
Učitel může zobrazit nebo skrýt vzorové řešení příkladu.

Učitel může přiřazovat značku (tag) studentovi, aby byl schopen filtrovat studenty.
Když student validuje pomocí DTD ve cvičení, tak výstup validace bude ve formátu, který zprostředkovává validační nástroj Xerces.
Měla by být možnost jednoduše přidat studenty určitého ročníku. Registrace na heslo, tj. učitel přijde do učebny, v systému zadá hodinové heslo na registraci. Systém nyní akceptuje registrace po dobu jedné hodiny na základě tohoto hesla.

Tab. 2 Seznam vybraných požadavků k implementaci.



CD č. 1 CD se zdrojovými kódy.