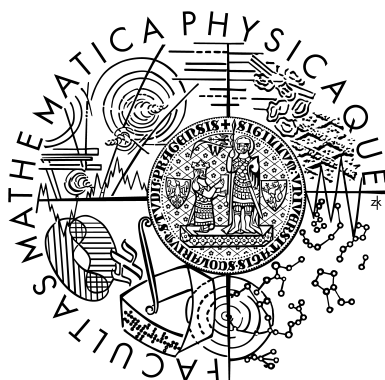


Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Rudolf Rosa

OprDU – Software pro opravu domácích úkolů

Katedra softwarového inženýrství

Vedoucí bakalářské práce: Mgr. Tomáš Knap

Studijní program: Informatika – Programování

2010

Na tomto místě bych rád poděkoval vedoucímu práce za přátelský přístup a dobré rady a doporučení. Dále chci poděkovat všem svým kolegům učitelům a svým studentům, bez jejichž podpory a rad by tato práce nejspíš nebyla vůbec vznikla.

Prohlašuji, že jsem svou bakalářskou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce.

V Praze dne 5.8.2010

Rudolf Rosa

Obsah

1 Úvod.....	6
2 Případové scénáře.....	8
2.1 Scénář 1: Státní škola.....	8
2.2 Scénář 2: Soukromý učitel.....	8
2.3 Scénář 3: Korektor textů.....	9
3 Architektura systému.....	10
3.1 Přehled komponent OprDU.....	10
3.2 Korektury v datovém formátu ODU.....	11
Vyznačování oprav – korektorské elementy.....	12
Bílé znaky.....	12
Zobrazování a export dokumentů.....	13
Příklad ODU dokumentu.....	13
3.3 Stavy dokumentu.....	16
3.4 Server.....	17
Databáze.....	17
Webové služby.....	18
3.5 OprDU Klient.....	19
Práce s dokumenty.....	19
Export dokumentů.....	19
Kontrola pravopisu.....	19
Makra pro časté opravy.....	20
Komunikace s OprDU Serverem.....	20
3.6 OprDU Student.....	20
4 Návrh řešení případových scénářů.....	22
4.1 Scénář 1: Státní škola.....	22
4.2 Scénář 2: Soukromý učitel.....	23
4.3 Scénář 3: Korektor textů.....	24
5 Srovnání s existujícími nástroji.....	25
5.1 Papír a tužka.....	25
Výhody.....	25
Nevýhody.....	25
Hodnocení.....	26
5.2 Textové procesory.....	26
Výhody.....	26
Nevýhody.....	27
Hodnocení.....	28
5.3 Specializovaný software.....	28
TurnItIn.....	28
Webkorr.....	29
Software pro studenty.....	29
5.4 OprDU.....	29
Výhody.....	30
Nevýhody.....	30

Hodnocení.....	31
6 Závěr.....	32
7 Seznam použité literatury.....	33
8 Příloha A: Obsah přiloženého disku.....	35
9 Příloha B: Uživatelské příručky.....	36
9.1 Příručka pro učitele.....	36
Před prvním spuštěním.....	36
Před opravou dokumentu.....	36
Oprava dokumentu.....	37
Export dokumentu.....	40
9.2 Příručka pro studenta (OprDU Student).....	40
Použití internetové aplikace OprDU Student.....	41
Použití Java aplikace OprDU Student.....	41
9.3 Příručka pro administrátora serveru.....	42
Potřebná konfigurace.....	42
Instalace.....	42
Nastavení.....	44
Adresy webových služeb.....	44
Odstávka serveru.....	44
Přizpůsobení funkcí.....	44
10 Příloha C: Datová příloha XML.....	45
10.1 ODU dokumenty.....	45
XML Schema pro ODU.....	45
XSLT skript odu2rtfTagged – transformace ODU dokumentu pro zobrazení v klientských programech.....	48
10.2 OprDU Server.....	55
WSDL pro OprDU server pro učitele.....	55
WSDL pro OprDU server pro studenty.....	59

Název práce: OprDU - Software pro opravu domácích úkolů

Autor: Rudolf Rosa

Katedra (ústav): Katedra softwarového inženýrství

Vedoucí bakalářské práce: Mgr. Tomáš Knap

e-mail vedoucího: Tomas.Knap@mff.cuni.cz

Abstrakt: Rešerší bylo zjištěno, že neexistuje vhodný nástroj pro učitele, který by jim usnadnil opravu domácích úkolů. Cílem práce je proto vytvoření komplexního a flexibilního softwarového řešení pro studenty a učitele pro práci s písemnými domácími úkoly. Řešení kombinuje výhody současných přístupů k problému a odstraňuje jejich nevýhody. Svými součástmi pokrývá všechny fáze práce na dokumentu, tj. jeho vytvoření studentem, odeslání učiteli, opravu učitelem, odeslání opraveného dokumentu studentovi a prohlédnutí oprav studentem.

Hlavní částí řešení je aplikace pro učitele, která umožňuje provádění oprav. Obsahuje pokročilé funkce, jako je kontrola pravopisu nebo export dokumentů. Důraz je zde kladen na maximální efektivitu a uživatelskou přívětivost.

Pro reprezentaci dokumentů byl vytvořen XML formát, který podporuje jednoduché vyznačování oprav, ale i vkládání dalších informací. Pro dokonalejší správu dokumentů a komunikaci mezi studentem a učitelem byl vytvořen server komunikující pomocí webových služeb. Součástí je také jednoduchá studentská aplikace pro práci s dokumenty.

Klíčová slova: oprava domácích úkolů, aplikační software, webové služby, školství

Title: OprDU - Software for Correcting Homework

Author: Rudolf Rosa

Department: Department of Software Engineering

Supervisor: Mgr. Tomáš Knap

Supervisor's e-mail address: Tomas.Knap@mff.cuni.cz

Abstract: It has been discovered that there is no suitable tool for teachers which would make correcting of homework easier for them. Therefore the goal of this work is to create a complex yet flexible software solution for students and teachers for dealing with written homework. The solution combines advantages of current approaches to the issue and eliminates their disadvantages. By its components it covers all phases of work on the document, i.e. its creation by the student, submission to the teacher, correction by the teacher, sending the corrected homework to the student and displaying the corrections to the student.

The main part of the solution is an application for the teacher which enables him to make the corrections. It contains advanced functions, e.g. spell-checking or document export. There is an emphasis on maximum effectiveness and user-friendliness.

An XML format has been created to represent the documents, which supports not only easy marking of corrections but insertion of other information as well. A server communicating via Web services has been created for a more sophisticated management of the documents and for communication between teachers and students. The last component is a simple student application for documents handling.

Keywords: homework correction, application software, Web services, education

Kapitola 1

Úvod

Kromě studia na Matematicko-fyzikální fakultě pracuji jako učitel cizích jazyků, díky čemuž jsem si uvědomil jeden zásadní nedostatek v softwarových nástrojích určených pro školství. V dnešní době již mají školy dostatek počítačů a další moderní technické vybavení. Co ale často chybí, je odpovídající software, který by umožnil využít výhod moderních technologií, a zároveň by byl přizpůsoben specifikům školního prostředí, jako je požadavek na jednoduché ovládání, nízkou cenu při velkém počtu uživatelů nebo pevné a ostré rozdělení uživatelů na dvě skupiny (učitelé a žáci).

Jde o obecný problém vývoje software – nejdokonalejší nástroje jsou ty, které jsou používány přímo programátory a podobnými profesemi. Naopak učitelé spíše mají s počítači problémy – učitel a programátor v jedné osobě je spíše výjimkou. Specializované softwarové nástroje pro učitele jsou proto vzácné a nástroje, které jsou v současné době dostupné a používané, jsou často pozadu za možnostmi, které máme.

V této práci jsem se zaměřil na jeden takový chybějící nástroj, a to software pro opravu domácích úkolů (odtud název *OprDU*). Stávající nástroje jsem v praktickém použití shledal být nedostatečnými, rozhodl jsem se proto vytvořit vlastní softwarové řešení, jehož zaměřením je obecně provádění korektur textu, se specializací na korektury domácích úkolů.

Učitelé (a nejen oni, ale i většina počítačových uživatelů) nebývají počítačovými odborníky, mnozí odmítají trávit s počítačem více času, než je nezbytně nutné, a mají téměř odpor k učení se pracovat s novými programy. Celý systém proto musí být co nejvíce uživatelsky přívětivý a snadný na pochopení a ovládání, ale zároveň natolik flexibilní, aby dokázal pokrýt většinu požadavků svých uživatelů. V neposlední řadě by také měl být dostatečně robustní a spolehlivý, aby v něj učitelé měli důvěru.

Hlavní součástí řešení je aplikace *OprDU Klient*, která je určena pro učitele a umožňuje provádět korekturu textu. Záběr práce je ale širší a pokrývá celý proces práce na domácím úkolu – od jeho vytvoření a odeslání na straně studenta přes opravu na straně učitele, včetně opatření komentáři, až po předání a prezentaci finálního dokumentu studentovi.

Cílem je, aby nebylo nutné používat pro různé fáze tohoto procesu různé vzájemně nekompatibilní programy. Proto jsou všechny navržené programy součástí jednoho komplexního systému s jasně definovanými univerzálními rozhraními – nyní mám na mysli zejména vlastní XML formát *ODU XML* a webové služby poskytované aplikací *OprDU Server*. Celý systém je navíc otevřený, takže libovolná jeho součást může být nahrazena jinou aplikací, ovšem za předpokladu dodržení výše zmíněných rozhraní. Již samotná tato práce nabízí dvě jednoduché implementace aplikace *OprDU Student* (aplikace pro odevzdávání úkolů a přijímání opravených dokumentů) – jednu postavenou na platformě Java, druhou v podobě webové aplikace.

Vývoj systému *OprDU* je vývoj motivovaný praxí. Software je průběžně testován v praxi, což umožňuje ověření, zda se práce na něm ubírájí správným směrem, stejně jako stanovení dalších cílů a rozlišení jejich důležitosti.

Aplikace obsahuje základní nástroje podporující rychlost provádění oprav, jako je kontrola podle slovníku nebo makra pro časté opravy. Nekladu si ale za cíl, alespoň v této verzi, žádné inteligentní automatické opravování chyb – to je ponecháno na učiteli.

Kapitola 2

Případové scénáře

Podívejme se na několik běžných situací, které bychom chtěli systémem OprDU řešit. Případové scénáře jsou jakési modelové případy, volené tak, aby se pokryly co nejvíce reálných případů.

2.1 Scénář 1: Státní škola

Větší škola, kde lze předpokládat větší objem domácích úkolů, ale také dostatečné technické zázemí – zejména alespoň několik počítačů přístupných učitelům i studentům a vlastní veřejně dostupný server.

Na škole jsou až stovky žáků a desítky učitelů. Někteří učitelé zadávají písemné domácí úkoly, jejichž vypracování studenti zasílají e-mailem v elektronické podobě.

Učitel typicky nemá na opravy příliš mnoho času, potřebuje nárazově opravit např. 30 prací bez výrazných vedlejších časových nákladů. Nechce se proto zdržovat neustálým otevíráním e-mailů, zkoumáním příloh v rozličných formátech, ukládáním souborů, posíláním e-mailů apod. Komunikace e-mailem má také svá úskalí – některé úkoly padají do spam-koše, někteří studenti mají takovou adresu a nastavené jméno odesílatele, že nelze poznat, o koho se jedná, apod.

Učiteli by vyhovovalo řešení, kde jednoduše jedním kliknutím otevře domácí úkol, opraví jej, a dalším kliknutím opravený pošle zpátky. Bylo by tedy nejlepší, kdyby kdyby studenti posílali úkoly v unifikovaném formátu, jinou cestou než e-mailem a každý úkol by se uložil na nějakém dobře dostupném místě – nejlépe tak, aby jej bylo možné otevřít ve stejném programu, ve kterém se pak provede jeho oprava.

Takové řešení ale učitel nenašel, proto si vždy všechny dokumenty vytiskne, opraví je v ruce a v pondělí je nechá rozdat ve třídě.

2.2 Scénář 2: Soukromý učitel

Učitel má několik stálých studentů a průběžně několik spíše jednorázových (například chystajících se na blížící se zkoušku). Pokud od studenta dostane e-mailem text k opravě, potřebuje jej opravit pokud možno rychle a obratem jej studentovi poslat opravený. Chce, aby jeho opravy vypadaly alespoň trochu profesionálně, proto používá textový procesor – není s ním ale plně spokojen (viz Kapitola 5.2). Chtěl by nějaký program specializovaný na opravy textů, který mu je umožní provádět kvalitně a efektivně.

Pro provádění oprav vždy používá tentýž počítač s připojením k internetu. Se studenty pravidelně komunikuje e-mailem, zasílání úkolů tímž médiem mu proto vyhovuje, neboť ve svém poštovním klientovi pak může přehledně sledovat veškerou komunikaci se studentem. Nepotřebuje ani žádné centrální úložiště dokumentů – pokud výjimečně potřebuje nějaký starší dokument, najde si ho v e-mailu.

2.3 Scénář 3: Korektor textů

Korektor často dostává texty ke korektuře od zákazníků, typicky e-mailem. Potřebuje kvalitně a srozumitelně provést korekturu textu a zaslat jej zpět.

Nepotřebuje univerzální nástroj, jakými jsou textové procesory. Raději by použil jednoúčelový nástroj, který bude umožňovat rychle, pohodlně a přehledně provést opravy.

Pokud se korektor věnuje opravování textů častěji, ocenil by také funkci centrální správy dokumentů, ve které by zřetelně viděl, které dokumenty má rozpracované, od kterého zákazníka je má a kdy je obdržel ke korektuře; zároveň by ale měl rád zachovanou historii všech zakázek.

Pokud má korektor stále zákazníky nebo pracuje jako firemní korektor, ocenil by také pohodlnější způsob předávání dokumentů, který by byl rychlý a jednoduchý.

Kapitola 3

Architektura systému

Ze Scénářů 1 – 3 v Kapitole 2 lze vypožorovat několik požadavků na navrhovaný systém. Vždy lze rozlišit dvě strany, které spolu komunikují (učitel a student, korektor a zákazník, a podobně). Pro zjednodušení budu dále v textu jednu stranu označovat jako *studenta* (ten, kdo vytváří dokument a požaduje opravu) a stranu druhou jako *učitele* (ten, kdo dokument opravuje).

Těžištěm celého systému je program pro provádění samotných korektur – ten, na rozdíl od ostatních součástí, je potřebný pro řešení každého ze scénářů. Tento program musí umožňovat jednoduše vyznačovat provedené opravy v dokumentu. Musí umožňovat export do souboru, který bude možné otevřít na většině počítačů. Musí podporovat jednoduchou komunikaci s okolím. Měl by také obsahovat funkce zvyšující efektivitu práce, jako je kontrola pravopisu (spell-checker).

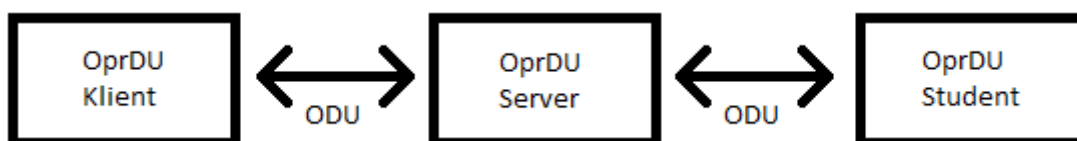
Požadavek na snadnou správu dokumentů a jednoduchou komunikaci vede na implementaci serveru, který bude poskytovat úložiště pro dokumenty a zprostředkovávat komunikaci s uživateli; ve Scénáři 2 tento server nebude využit. Pro učitele jako klientský program poslouží aplikace pro vyznačování korektur – bude tedy možné splnit požadavek na otevření dokumentu jedním kliknutím. Studentovi bude stačit tenký klientský program, který bude zprostředkovávat komunikaci se serverem a minimum dalších služeb – opět ho nevyužijeme pro Scénář 2.

S dokumenty musí být pracováno ve formátu, který bude jednoduchý, aby bylo možné je snadno zpracovávat, ale zároveň umožní vyznačit jednotlivé typy oprav včetně doplňkových informací. Protože žádný vyhovující formát nebyl nalezen, bylo využito XML standardu (dle [26], [31], [6], [7] a [1]) pro vytvoření vlastního formátu dokumentů. Formát XML navíc pomocí XSLT (dle [30] a [32]) umožňuje snadnou transformaci dokumentu do jiných formátů; jako výchozí formát pro zobrazování a export bylo zvoleno RTF (dle [14]).

3.1 Přehled komponent OprDU

OprDU (software pro Opravu Domácích Úkolů) se skládá z následujících komponent (způsob jejich vzájemné komunikace ukazuje Obrázek 1):

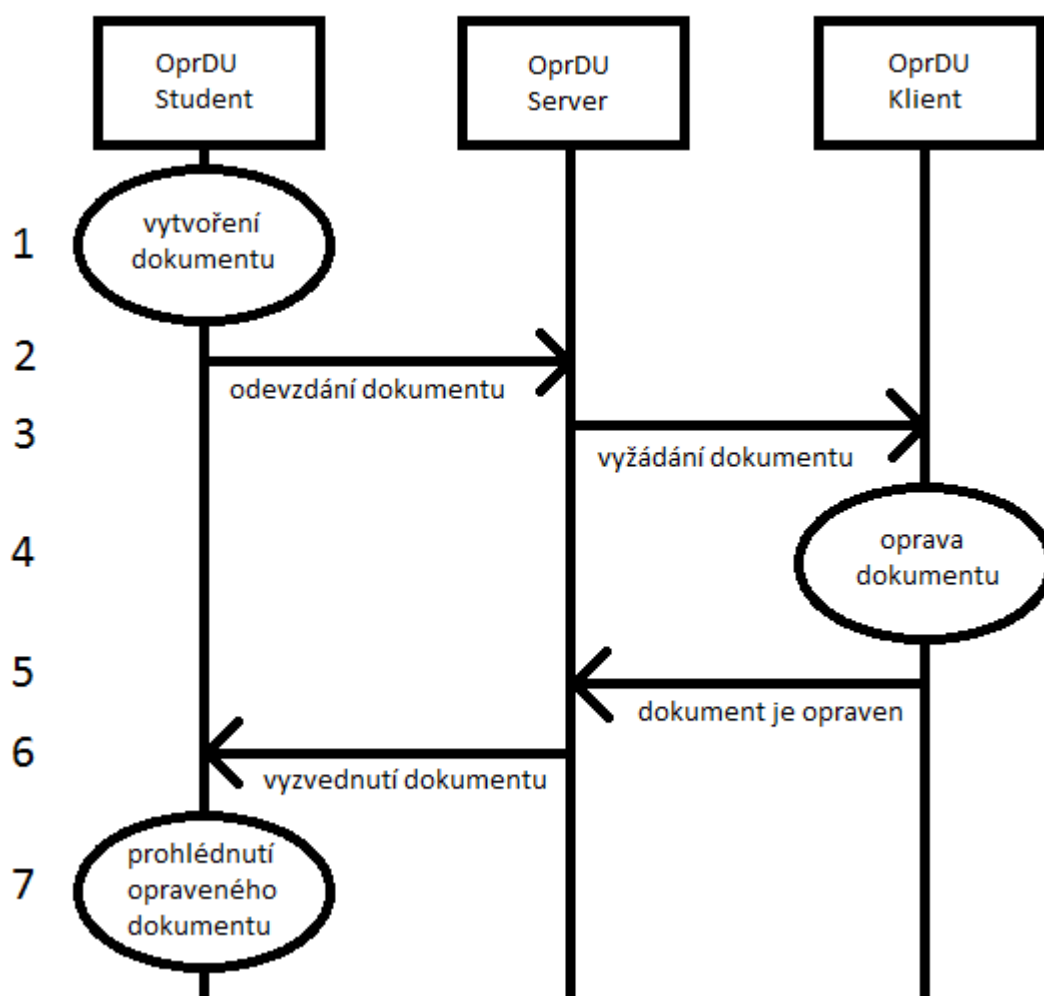
- *ODU* – XML datový formát pro dokumenty
- *OprDU Server* – správa dokumentů a uživatelů
- *OprDU Klient* – oprava dokumentů (učitel)
- *OprDU Student* – vytváření a prohlížení dokumentů (student)



Obrázek 1: Vzájemné propojení součástí OprDU

Práce na domácím úkolu při plném využití OprDU probíhá takto (čísla v následujícím seznamu odpovídají číslům v obrázku – viz Obrázek 2):

1. Student vytvoří dokument (OprDU Student)
2. Student odevzdá dokument učiteli (uložení na server)
3. Učitel si vyžádá dokument pro opravu (stažení ze serveru)
4. Učitel provede opravu dokument (OprDU Klient)
5. Učitel odevzdá opravený dokument (uložení na server)
6. Student si vyzvedne opravený dokument (stažení ze serveru)
7. Student si prohlédne opravený dokument (OprDU Student)



Obrázek 2: Práce na dokumentu při plném využití OprDU

3.2 Korektury v datovém formátu ODU

Pro práci s dokumenty byl navržen nový XML datový formát nazvaný *ODU*, který svou strukturou odpovídá potřebám aplikace. Při jeho návrhu byl kladen důraz na jeho jednoduchost, efektivitu zpracování dokumentů a srozumitelnost kódu. Zde jsou uvedeny nejdůležitější charakteristiky datového formátu ODU, detailní popis formátu v jazyce XML Schema je přiložen v Kapitole 10.1.

Formát se částečně inspiroval jazykem HTML [27] (mnohé elementy proto budou čtenáři připadat povědomé). Každý dokument se skládá z hlavičky (element `head`), která obsahuje metadata (název, autor, atd.), a těla (`body`), obsahujícího vlastní text dokumentu s opravami.

Podobně jako v HTML je text dokumentu obsažen přímo v elementu `body` (proložený elementy `br` pro zalomení řádku) a při opravách se chybné úseky textu obalují korektorskými elementy. Všechny elementy obalující úseky původního textu mají povinný celočíselný atribut `id`, který umožňuje snadnou orientaci a vyhledávání v dokumentu.

Dokumentu je při prvním uložení přiděleno *uid* – čtyřicetiznakový řetězec, pomocí kterého lze dokument jednoznačně identifikovat.

Vyznačování oprav – korektorské elementy

Jsou rozlišovány čtyři typy oprav (rozlišené čtyřmi korektorskými elementy): *Změna* (`chg`), *Vložení* (`ins`), *Smazání* (`del`) a *Změna pořadí slov* (`wo`); ke korektorským elementům lze dále přiřadit element *Komentář* (`c`). Původní text je obsažen přímo v kořenovém elementu opravy (s výjimkou *Změny pořadí slov*) a může být obalován dalšími korektorskými elementy, čímž je umožněno zanořování oprav. Další informace o opravě (správná varianta, komentář...) jsou obsaženy v attributech nebo subelementech korektorského elementu.

U *Změny pořadí slov* je původní text obalen tagy `order`, jejichž celočíselný atribut `o` určuje správné umístění textu – např. oprava textu *happy am I* → *I am happy* bude vyznačena následujícím způsobem (viz Příklad 1):

```
<wo>
  <order o="3">happy</order>
  <order o="2">am</order>
  <order o="1">I</order>
</wo>
```

Příklad 1: oprava typu Změna pořadí slov (pro názornost jsou vynechány další potřebné atributy, jako `id` a `ws` – komplexní příklad je připojen na konci kapitoly)

Bílé znaky

Formát se podobně jako HTML řídí pravidlem, že libovolný počet bílých znaků v textu znamená jednu mezeru. Na rozhraní textu a korektorského elementu jsou pak mezery určeny atributem `ws` (`whitespace`). Jde o nepovinný atribut, který je možné uvést u všech elementů opravy (tj. `chg`, `ins`, `del`, `wo`, `c`). Povolené hodnoty jsou:

- `all` (výchozí) – mezera z obou stran (*já **bysem** chtěl* → *já **bych** chtěl*)
- `left` – mezera nalevo, vpravo navazuje těsně – využití např. při opravě slova na konci věty (*Ahoj **Peter**!* → *Ahoj **Petře**!*)
- `right` – obdobně; využití např. při připojení písmen na konec slova (*čern Petr* → *černý Petr*)

- `none` – těsné přiléhání z obou stran – např. při opravě týkající se jen části slova (*autbus* → *autobus*)

Zalomení řádku se určuje explicitně pomocí elementu `br`.

Zobrazování a export dokumentů

Pro zobrazování a export dokumentů se používá několik XSLT skriptů. Nejdůležitějším z nich je `odu2rtfTagged` (přiložen v Kapitole 10.1), který transformuje ODU do *tagovaného RTF* s vyznačenými opravami – tento skript se používá v programu OprDU Klient pro zobrazení dokumentu v editačním okně. Díky vloženým tagům, tzv. *odu tagům*, se pak v editačním okně určuje poloha kurzoru v dokumentu, význam mají pro určení `id` vybrané opravy, apod. Editace nad tagovaným RTF byla zvolena proto, že přímá vizualizace a editace XML dat je u dokumentů textové povahy značně problematická.¹

Odu tagy nejsou XML tagy, ale jejich mechanismus je podobný a vznikají přímým přepisem původních XML elementů. Ukázka přepisu XML na tagované RTF (bez formátovacích značek RTF) viz Příklad 2:

```
<del id="12">the</del>
##odu@@+del 12##oduend@@ the ##odu@@-del 12##oduend@@
```

Příklad 2: Přepis XML na tagované RTF – na prvním řádku XML kód, na druhém odpovídající tagované RTF

Příklad ODU dokumentu

Všechny popisované vlastnosti formátu ODU si lze prohlédnout na příkladu ODU dokumentu (viz Příklad 3). Jsou vyplněny všechny údaje v hlavičce – stav dokumentu je 400 (opravený), element `maxid` označuje maximální přidělenou hodnotu atributu `id`. V těle dokumentu jsou použity všechny typy korektorských elementů včetně jednoho příkladu vnořených oprav (elementy s `id` 7 a 6). Jsou vyznačeny oba druhy komentářů – přímo u opravy (`id` 3) nebo jako samostatný komentář (`id` 17). Jsou použity různé hodnoty atributu `ws`, lze také vidět nahrazování speciálních znaků znakovými entitami (`id` 16).

Jsou také přiloženy ukázky tří XSLT exportů téhož dokumentu do RTF – jde o skutečný výstup z aplikace OprDU Klient. Na první ukázce (viz Příklad 4) je „původní dokument“, tj. export do podoby, v jaké byl dokument odevzdán studentem. Druhá ukázka (viz Příklad 5) je standardní zobrazení dokumentu, tj. s vyznačenými opravami – takto se dokument zobrazuje v klientských programech. Poslední ukázkou (Příklad 6) je „bezchybný dokument“, tj. export s aplikováním vyznačených oprav – aplikovány jsou všechny čtyři typy oprav, komentáře se zde nijak neprojevují.

¹ U dokumentů datové povahy, např. databázi v XML, je toto naopak snadné, neboť taková data lze obvykle zobrazit a editovat v podobě tabulky nebo stromu.

```

<?xml version="1.0" encoding="utf-8"?>
<odu>
  <head>
    <uid>a1269324443t123456789s380217339r</uid>
    <title>Ptakopysk podivný</title>
    <student>jarda.vopicka@seznam.cz</student>
    <studentName>Jaroslav Orangutan</studentName>
    <created>2010-02-02 21:22:48</created>
    <comment>Text o ptakopyskoví</comment>
    <teacher>rur@seznam.cz</teacher>
    <teacherName>Rudolf Rosa</teacherName>
    <status>400</status>
    <maxid>16</maxid>
  </head>
  <body>
    Ptakop
    <chg ws="none" id="3" title="vyjmenované slovo &quot;pysk&quot;">
      i
      <r>y</r>
    </chg>
    sk podivný
    <ins ws="all" id="4">
      <r>je</r>
    </ins>
    <c ws="left" id="17" title="velmi správně!">savec</c>
    , kt
    <ins ws="none" id="5">
      <r>e</r>
    </ins>
    rý ale
    <wo ws="all" id="7">
      <order o="3" id="14">snáší</order>
      <order o="2" id="11">
        jako ptá
        <del ws="none" id="6">á</del>
        ci
      </order>
      <order o="1" id="9">podobně</order>
    </wo>
    vejce.
    <br id="2" />
    Má také
    <del ws="all" id="15">také</del>
    <chg ws="all" id="16" title="&quot;divný&quot; se sem nehodí">
      divný
      <r>zvláštní</r>
      <other>zajímavý</other>
    </chg>
    zobák.
  </body>
</odu>

```

Příklad 3: Zdrojový kód ODU dokumentu, ve stavu 400 (opravený)

Jaroslav Orangutan: Ptakopysk podivný

Ptakopysk podivný savec, ktrý ale snáší jako ptááci podobně vejce.
Má také také divný zobák.

Příklad 4: Původní dokument (bez vyznačených oprav)

Jaroslav Orangutan: Ptakopysk podivný

Ptakopiⁱ_v [vyjmenované slovo "pysk"] sk podivný ^{je} savec [velmi správně!], který ale ^[3]snáší
^[2]jako ptá^áci ^[1]podobně vejce.

Má také také divný zvláštní / zajímavý ["divný" se sem nehodí] zobák.

Příklad 5: Standardní zobrazení dokumentu (s vyznačenými opravami)

Jaroslav Orangutan: Ptakopysk podivný

Ptakopysk podivný je savec, který ale podobně jako ptáci snáší vejce.
Má také zvláštní zobák.

Příklad 6: Bezchybný dokument (po aplikaci oprav)

3.3 Stavy dokumentu

Během svého životního cyklu dokument prochází různými stavy, které vyjadřují, co se s ním právě děje, a určují oprávnění uživatelů k dokumentu (oprávnění viz Tabulka 1). Jsou označeny trojčífernými kódy a slovním popisem (přehled viz Tabulka 2). Stav dokumentu určuje první číslice, zbývající dvě jsou rezervovány pro budoucí jemnější rozlišení. Aktuálně má každý stav kromě základní varianty ×00 ještě variantu ×50, což znamená, že dokument byl vytvořen učitelem, v programu OprDU Klient, a ne studentem v OprDU Student.

<i>Značka</i>	<i>Název</i>	<i>Popis přístupového oprávnění</i>
L	list	Uživateli se dokument zobrazuje ve výpisu (zobrazuje se autor, datum a stav, nezobrazuje se obsah)
G	get	Uživatel smí stáhnout dokument do svého počítače
R	read	Uživatel smí prohlížet dokument
E	edit	Uživatel smí upravovat dokument (měnit textový obsah dokumentu)
C	correct	Uživatel smí opravovat dokument (vkládat/měnit korektorské značky)
D	delete	Uživatel smí smazat dokument
S	set	Uživatel smí změnit (zvýšit) stav dokumentu

Tabulka 1: Možná přístupová oprávnění uživatelů (tj. studenta a učitele)

<i>Kód</i>	<i>Název stavu</i>	<i>Popis stavu</i>	<i>Oprávnění studenta</i>	<i>Oprávnění učitele</i>
100	Rozepsaný (creating)	Výchozí stav dokumentu. Dokument v tomto stavu vzniká, když ho student píše a průběžně jej ukládá.	LGREDS	–
200	Odevzdaný (submitted)	Do tohoto stavu dokument přejde jeho odevzdáním. Dokument nyní může být vyžádán učitelem pro opravy.	LGR	LGRS
300	Opravuje se (correcting)	Tento stav znamená, že učitel si vyžádal dokument pro opravu (a lze předpokládat, že jej začal upravovat). Opravovaný dokument může být průběžně ukládán na server.	L	LGRSC
400	Opravený (corrected)	Když učitel dokončí opravu dokumentu a finální verzi uloží na server, přejde dokument do tohoto stavu. Od této chvíle si student může dokument vyzvednout.	LGS	LGR
500	Opravený a vyzvednutý (displayed)	Změna stavu ze 400 na 500 odpovídá „potvrzení o přečtení opravy“ – vyjadřuje, že si student vyzvedl opravený dokument.	LGR	LGR

Tabulka 2: Stavy dokumentu

3.4 OprDU Server

OprDU Server spravuje databázi dokumentů a uživatelů a klientským programům poskytuje přístup k ní pomocí webových služeb [28].

Server je implementován v jazyce PHP (dle [10] a [5]) s využitím databáze MySQL (dle [21] a [22]). Toto řešení bylo zvoleno z důvodu vysokého rozšíření těchto technologií a velmi nízkých nákladů na jejich implementaci. Pokud se neočekává vysoká zátěž serveru, je dokonce možné jej provozovat i na některých freehostingových službách.²

Pro administrátora serveru jistě nebude problémem větší či menší úprava zdrojových kódů, ať už z důvodu funkčních požadavků nebo kompatibility s prostředím.

Databáze

Databáze je řešena pomocí tří tabulek – `odudocument`, `oduteacher` a `odustudent`.³

Tabulka `odudocument` (schéma viz Tabulka 3) obsahuje vlastní dokumenty, které jsou identifikovány svým `uid`. Celý obsah těla dokumentu je uložen ve druhém sloupci tabulky, informace z hlavičky dokumentu jsou rozděleny do ostatních sloupců.

<i>Sloupec</i>	<i>Typ</i>	<i>Odkazuje na</i>
<code>uid</code>	<code>varchar(40)</code>	
<code>body</code>	<code>mediumtext</code>	
<code>student</code>	<code>varchar(255)</code>	<code>odustudent.student</code>
<code>teacher</code>	<code>varchar(255)</code>	<code>oduteacher.teacher</code>
<code>status</code>	<code>int(3)</code>	
<code>title</code>	<code>text</code>	
<code>created</code>	<code>datetime</code>	
<code>comment</code>	<code>text</code>	
<code>maxid</code>	<code>smallint(6)</code>	

Tabulka 3: Struktura tabulky `odudocument`

² OprDU Server byl úspěšně tetován na hostingu Webzdarma, <http://www.webzdarma.cz>

³ Schémata tabulek v této kapitole jsou generována programem PHPMyAdmin [2].

Tabulky *oduteacher* (schéma viz Tabulka 4) a *odustudent* (Tabulka 5) jsou velmi podobné, obsahují informace o učitelích a studentech. Slouží zejména k jejich přihlašování do systému, proto vždy obsahují jednoznačný identifikátor uživatele (e-mailová adresa) a hash jeho hesla (hashovaný algoritmem sha1 [17], jako sůl⁴ slouží identifikátor uživatele). Tabulka *oduteacher* navíc obsahuje ještě pole s nastaveními uživatele – sem se v aktuální verzi ukládají nastavení maker.

<i>Sloupec</i>	<i>Typ</i>
student	varchar(255)
password	varchar(40)
name	varchar(255)

Tabulka 4: Struktura tabulky odustudent

<i>Sloupec</i>	<i>Typ</i>
teacher	varchar(255)
password	varchar(40)
name	varchar(255)
settings	longtext

Tabulka 5: Struktura tabulky oduteacher

Pro podporu přihlašování do webové aplikace OprDU Student je zde ještě tabulka *oduloginstudent*. V případě nasazení, ve kterém nebude tato webová aplikace využívána (studenti k serveru nepřistupují nebo k němu přistupují pomocí jiné aplikace), není tato tabulka využívána.

Webové služby

Webové služby poskytují klientským aplikacím přístup k serveru. Poskytují metody pro stahování a ukládání dokumentů, a dále několik pomocných metod, jako je výpis dokumentů, výpis uživatelů, apod.

Služby jsou rozděleny do dvou *serverů webových služeb* (Web Services Server), přičemž jeden je určen učitelům (přístup pomocí programu OprDU Klient) a druhý studentům (program OprDU Student). Poskytované služby jsou v podstatě stejné – volají se většinou stejné obslužné metody, pouze s jinými parametry. Rozdělení je provedeno především z důvodu různých oprávnění učitelů a studentů.

Obslužné metody zajišťují autentifikaci uživatele, poskytnutí požadované funkce a základní kontrolu chyb a informování o nich, a to pomocí chybových kódů v odpovědi zaslané klientovi. O poskytnutí/neposkytnutí požadované služby se rozhoduje na základě korektnosti přihlášení, statutu uživatele (učitel/student), existence přiřazení dokumentu uživateli a stavu dokumentu.

⁴ Solený hash (Salted Hash) je technika, kterou se odstraňuje nežádoucí jev nastávající při obyčejném hashování hesel, kdy uživatelé se stejným heslem mají i stejný hash hesla. Spočívá v tom, že se řetězec představující heslo zřetěží s jiným řetězcem (označovaným jako sůl) a tento výsledný řetězec se zahashuje (podle [33] a [3]).

Pro implementaci webových služeb byl využit balík NuSOAP (viz [19] a [18]). Webové služby generují WSDL [29] – viz Kapitola 10.2.

3.5 OprDU Klient

OprDU Klient je klientskou aplikací pro učitele. Tato aplikace je srdcem celého systému OprDU a společně s datovým formátem je jeho nenahraditelnou součástí. Umožňuje vlastní opravu dokumentu, jsou zde přidružené další funkce (kontrola pravopisu, makra). Aplikace je určena pro práci online (s přístupem k OprDU Serveru), ale podporuje i práci offline, s načítáním dokumentů z disku.

Klient je napsán pro platformu .NET Frameworku verze 3.5 (viz [16]) v jazyce C# (viz [15], s využitím [24]), a je tedy prakticky jako jediná součást OprDU platformně závislý. Tato platforma byla zvolena pro dobrou podporu používaných technologií, zejména XML [26], XSLT [30], webových služeb [28] a RTF [14].

Aplikace je vytvářena jako vícejazyčná, implementována je česká a anglická verze. Použit je standardní lokalizační mechanismus .NETu, což umožňuje snadné vytvoření dalších jazykových verzí.

Práce s dokumenty

S dokumentem se vnitřně pracuje nad jeho XML reprezentací. Protože dokument je svou povahou lineární, používá se k jeho čtení `XmlTextReader`⁵ a k jeho modifikaci XSLT. K orientaci v dokumentu slouží atributy `id` u korektorských elementů.

Protože vhodné interaktivní zobrazení XML dat uživateli je komplikované (zejména v případě jako je tento, kdy data reprezentují textový dokument), provádí se samotná oprava dokumentu nad dokumentem transformovaným do *tagovaného RTF* (viz Kapitola 3.2). Dokument je zobrazován v `RtfBoxu`,⁶ který je zamknut jen pro čtení. Iluze přímého psaní do dokumentu je vytvářena za pomoci vyhledávání `id` korektorských elementů v tagovaném RTF (pro určení aktuální polohy v dokumentu), modifikace zdrojového XML, a konečně XSL transformace zpět do RTF, které je následně zobrazeno. Protože transformace celého dokumentu by byla zbytečně pomalá, je vyříznuta pouze aktuálně měněná část, která se transformuje a zřetězí s ostatními částmi dokumentu.

Export dokumentů

Pro export dokumentů je využito síly XSLT a .NETu: veškerý export probíhá přímo z XML pomocí ručně vytvořených XSLT skriptů.

Kontrola pravopisu

Kontrola pravopisu se provádí s pomocí slovníků v textovém formátu, kde jedno slovo odpovídá jednomu řádku. Kontrola probíhá velmi jednoduše:

⁵ `System.Xml.XmlTextReader`, třída poskytující rychlé, necachované a pouze dopředné čtení XML dat (volně citováno z [13]).

⁶ `System.Windows.Forms.RichTextBox`, reprezentuje textové pole s textem ve formátu RTF [14] (volně citováno z [13]).

- slovníky jsou načteny do struktury `SortedDictionary`⁷ (po spuštění aplikace)
- dokument je transformován (pomocí XSLT) do seznamu slov oddělených mezerami (u oprav se bere správná verze) a rozdělen do pole po jednotlivých slovech
- pole je procházeno, každé slovo vyhledáno ve slovníku a nenalezená slova zobrazena uživateli (po kliknutí je slovo nalezeno a označeno v dokumentu)

Kontrola pravopisu je implementována pro angličtinu, použit byl veřejně dostupný slovník SCOWL [9].

Makra pro časté opravy

Pro často prováděné opravy lze definovat makra. U makra lze definovat stejné položky jako při vkládání opravy; definice maker se ukládají na OprDU Server (popřípadě lokálně na disk).

Komunikace s OprDU Serverem

Pro komunikaci s OprDU Serverem je s výhodou použita velmi silná podpora webových služeb v .NETu. Pomocí příslušného nástroje byly importovány informace o webové službě (z WSDL souboru, viz Kapitola 10.2) a vygenerovaný kód byl pouze mírně upraven, např. pro podporu nastavení adres serverů nebo pro akceptaci nedůvěryhodných SSL certifikátů.⁸ Složitěji strukturované odpovědi serveru jsou pak zpracovány opět za použití `XmlTextReader` nebo `XmlDocument`.⁹

3.6 OprDU Student

Aplikace OprDU Student je tenký jednoduchý klient pro studenta. Umožňuje vytvoření dokumentu, jeho odeslání na OprDU Server a následně vyzvednutí opraveného dokumentu.

Součástí bakalářské práce jsou dvě implementace OprDU Studenta – webová aplikace v jazyce PHP a desktopová aplikace v jazyce Java [20]. Jde tedy o multiplatformní aplikaci.

Webová aplikace je jednoduchá, podporuje základní operace (vytvoření dokumentu, jeho odevzdání a následné prohlédnutí opraveného dokumentu). Je vytvářena s ohledem na maximální jednoduchost použití (dle [8]). Musí být spuštěna přímo na OprDU Serveru, přistupuje totiž přímo do databáze dokumentů a uživatelů.

Javová implementace nabízí všechny základní funkce práce s dokumentem (vytvoření, smazání, editace, odevzdání, prohlížení). Se serverem komunikuje pomocí webové služby, může proto být spuštěna na libovolném počítači se síťovým přístupem k OprDU

⁷ `System.Collections.Generic.SortedDictionary`, kolekce párů klíč/hodnota seřazených podle klíče (volně citováno z [13]).

⁸ Tímto sleduji možnost co nejjednoduššího a nejlevnějšího nasazení systému OprDU. Získání důvěryhodného certifikátu je finančně náročné a jeho instalace je v některých případech technicky složitá. Samopodepsaný (a tedy nedůvěryhodný) certifikát lze naopak získat snadno a zadarmo, přičemž poskytuje srovnatelnou úroveň zabezpečení jako certifikát důvěryhodný.

⁹ `System.Xml.XmlDocument`, DOM [25] reprezentace XML dokumentu (volně citováno z [13]).

Serveru. Aplikace je okenního typu, grafické rozhraní je vykreslováno pomocí balíčku *Swing*.¹⁰ Dále je využito pět balíčků od The Apache Software Foundation [4] (*Axis*,¹¹ *Commons Lang*,¹² *Commons Logging*,¹³ *Xalan*¹⁴ a *Xerces*¹⁵) a balíček *JavaMail* od společnosti Sun Microsystems (nyní Oracle).¹⁶ Podrobnosti o jednotlivých balíčcích lze nalézt na přiloženém disku (viz Kapitola 8).

10 Balíček *Swing* umožňuje pro vytváření GUI a je standardní součástí Javy.

11 Apache *Axis*, implementace SOAP mechanismu, domovská stránka <http://ws.apache.org/axis/>

12 Apache *Commons Lang*, pomocné utility pro balíček `java.lang`, domovská stránka <http://commons.apache.org/lang/>

13 Apache *Commons Logging*, rozhraní a implementace pro vytváření logů, domovská stránka <http://commons.apache.org/logging/>

14 *Xalan-Java*, XSLT procesor pro transformaci XML dokumentů, domovská stránka <http://xml.apache.org/xalan-j/>

15 *Xerces2 Java Parser*, XML parser, domovská stránka <http://xerces.apache.org/xerces2-j/>

16 *JavaMail API*, podpora práce s e-maily, domovská stránka <http://www.oracle.com/technetwork/java/index-jsp-139225.html>

Kapitola 4

Návrh řešení případových scénářů

Podívejme se nyní, jak lze pomocí OprDU řešit Scénáře z Kapitoly 2. Ne vždy je nezbytné použít všechny jeho součásti, tj. OprDU Klient, Server i Student.

4.1 Scénář 1: Státní škola

Jde o ukázkovou situaci, na kterou je systém OprDU uzpůsoben. Řešení bude obsahovat všechny součásti systému, tj. OprDU Klient, Server, Student. Nasazení systému bude na počátku představovat jednorázové (především časové) náklady na zavedení systému, následně ale bude potřeba provádět pouze zaškolování nových studentů a učitelů a instalace na nové počítače.

OprDU Server

Administrátor školního serveru instaluje OprDU Server. Školní server musí podporovat PHP a MySQL a být dostupný z internetu.

Server lze použít tak, jak je (po provedení nutných nastavení), přístup k registraci by ale neměl být ponechán veřejně přístupný. Jediné, co musí být přístupné, jsou rozhraní webových služeb. Každému studentovi a každému učiteli (případně těm učitelům, kteří si o to zažádají) bude zřízen účet na serveru.

Alternativně lze OprDU server napojit na existující databázi učitelů a/nebo studentů. Toto řešení si vyžádá úpravu zdrojových kódů serveru, především funkcí pro přihlašování uživatelů a pro ukládání dokumentů (viz Kapitola 9.3).

Data na serveru musejí být pravidelně zálohována, aby se předešlo ztrátě prací studentů i učitelů. Musí být zálohována zejména tabulka `odudocuments`, která obsahuje vlastní dokumenty a jejíž data jinde v systému zálohována nejsou. Při ztrátě registračních údajů studentů nebo učitelů stačí vytvořit nové účty se stejnými e-mailovými adresami, příslušné dokumenty k nim budou opětovně přiřazeny automaticky.¹⁷

OprDU Klient

Na školní počítače bude instalován OprDU Klient – počítače musejí mít operační systém Microsoft Windows XP nebo novější a musí na ně být instalován .NET Framework verze alespoň 3.5; samozřejmě musejí mít přístup k OprDU Serveru. V nastavení klienta bude změněna adresa výchozího OprDU Serveru na adresu instalované instance, konkrétně na rozhraní její webové služby pro učitele.

Učitelé dostanou instruktáž zacházení s programem. Volitelně (a doporučeně) si učitelé instalují OprDU Klienta na své osobní počítače (domácí počítače, notebooky...). Díky centrálnímu úložišti dat na OprDU Serveru mohou bez problémů na opravě úkolů

¹⁷ Uživatel je vždy identifikován na základě své e-mailové adresy.

pracovat z libovolného počítače, na kterém se v OprDU Klientovi přihlásí do svého účtu – dokumenty budou vždy ve stavu, v jakém byly uloženy. Mohou tedy například začít na opravách pracovat již ve škole a dokončit je po příchodu domů.

OprDU Student

Všichni studenti budou instruováni, jak si instalovat a používat program OprDU Student. Pomocí něj budou zasílat domácí úkoly svým učitelům a následně si vyzvedávat opravené dokumenty.

V případě technických problémů (např. nefunkční připojení k internetu) má student vždy možnost předat dokument učiteli jiným způsobem (e-mail, flashdisk...), neboť učitel může ručně vložit dokument do systému ze svého OprDU Klienta. Stejně tak může učitel opravený dokument například zaslat studentovi emailem nebo předat vytištěný, pokud by student měl s jeho vyzvednutím problémy.

4.2 Scénář 2: Soukromý učitel

Na tuto situaci je opět vhodným řešením systém OprDU. V tomto případě si ale učitel pravděpodobně vystačí pouze s OprDU Klientem. Ostatní součásti může také výhodně využít, musí si ale rozmyslet, zda by nešlo o naddimenzované řešení – zejména v případě jednorázových studentů je zbytečné, aby si kvůli několika málo dokumentům instalovali a provozovali OprDU Studenta.

OprDU Klient

Učitel bude v OprDU Klientovi pracovat pouze „offline“, tedy se soubory na disku. Vytvoří si na disku složku, do které bude ODU dokumenty ukládat. Z textu, který dostane e-mailem, vytvoří nový dokument, který uloží na disk, provede nad ním opravy, nakonec jej exportuje (např. do RTF) a zašle e-mailem studentovi.

Volitelně OprDU Server

Funkci OprDU Serveru zde v podstatě nahrazuje poštovní klient. Slouží k přijímání i odesílání dokumentů, zároveň (pokud uchovává přijaté i odeslané zprávy) v něm lze dohledat v minulosti opravené dokumenty. Díky předpokládanému nízkému počtu studentů a nízkému množství úkolů by toto mělo být postačující.

Pro dokonalejší a přehlednější správu dokumentů si učitel může instalovat lokální OprDU Server, zejména pokud doma používá několik počítačů (pak ale musí být tento server dostupný ze všech těchto počítačů) nebo pokud je množství opravovaných dokumentů větší.

Pokud má učitel větší počet stálých studentů, může instalovat OprDU Server na nějaký veřejně dostupný server a své studenty instruovat k instalaci programu OprDU Student (viz následující podkapitola). Je možné použít i freehosting¹⁸ nebo použít veřejný server na adrese <http://oprdu.nikde.eu>.

18 OprDU Server byl úspěšně testován na hostingu Webzdarma, <http://www.webzdarma.cz>

Volitelně OprDU Student

Program OprDU Student není nutný, pokud učitel zašle studentům opravený úkol v příloze e-mailu. Měl by jej zasílat exportovaný do formátu, který bude pro studenta dobře čitelný, například RTF¹⁹ – může přitom být dobré poslat více exportů, např. text s vyznačenými opravami a správný text (s aplikovanými opravami).

Student si může instalovat OprDU Studenta, poté postačuje, aby mu učitel zaslal ODU dokument, není nutné provádět export.

Pokud se učitel rozhodne používat nějaký veřejný OprDU Server, je vhodné, aby si stáli studenti instalovali OprDU Studenta a zasílali úkoly pomocí něj.

4.3 Scénář 3: Korektor textů

Korektor většinou provádí korekturu pokaždé pro někoho jiného, proto mu typicky bude postačovat pouze aplikace OprDU Klient (a práce „offline“, bez OprDU Serveru, podobně jako ve Scénáři 2 – viz 4.2).

Věnuje-li se korekturám ve větší míře, bude pro něj přínosem OprDU Server, neboť mu umožní lepší organizaci dat.

Pokud korektor pracuje v nějaké větší firmě, případně je dokonce korektorů více, vyplatí se použít všechny součásti řešení, kdy OprDU Server spravuje zaměstnavatel a OprDU Student se používá pro zadávání textů korektorům a následné přebírání vypracovaných korektur. OprDU Server napomáhá dobré organizaci práce a pořádku v dokumentech, OprDU Student pak může sloužit libovolnému zaměstnanci firmy pro zaslání textu ke korektuře firemnímu korektorovi.

¹⁹ Jde sice o proprietární formát Microsoft Corporation, ale je platformě nezávislý a jeho specifikace je veřejná, měl by být proto bez problémů zobrazitelný na libovolném počítači.

Kapitola 5

Srovnání s existujícími nástroji

V současnosti učitelé pro opravu domácích úkolů používají převážně dvě metody – klasickou metodu „papír a tužka“ a textové procesory z kancelářských balíků.

5.1 Papír a tužka

Je až s podivem, že, přestože většina studentů své práce vypracovává na počítači, mnoho učitelů je stále vyžaduje v papírové podobě (ti trochu pokrokovější je přijímají i elektronicky, aby si je pak vytiskli sami). Na opravu práce si vystačí s červenou tužkou, ignorující tak veškerý technický pokrok posledních let.

Výhody

Výhody tohoto postupu samozřejmě spočívají právě v jeho jednoduchosti a univerzalitě.

Student ani učitel nepotřebují žádné speciální vybavení, potřebné nástroje je možné zakoupit velmi levně. Opravu může učitel provést kdykoliv, kdekoliv a velice rychle, není nutné řešit otázky formátu dokumentu ani způsobu jeho ukládání. Opravy se neomezuji na souvislé texty, bez zvýšené složitosti je možné opravovat komplikovaně strukturované dokumenty, ale i obrázky apod.

Nevýhody

Jakmile ale student nebo učitel požaduje jakékoli pokročilejší vlastnosti opravy, nenabízí toto řešení vůbec nic.

Přehlednost oprav je diskutabilní – pokud jsou opravy v textu velmi řídké, pravděpodobně zde problém nenastane, avšak s jejich zvyšující se četností se učitelovy vpisky stávají méně a méně přehlednými, až vzniká nesrozumitelná změť slov, čar a šipek. Prostor pro opravy je fyzicky omezen, učitel proto občas musí vtěsnat příliš mnoho informací na příliš malou plochu, případně použít šipky, odkazy, hvězdičky apod. Hodnota takové opravy pro studenta je pak velmi nízká, až žádná – zejména má-li učitel navíc těžce čitelný rukopis.

Tužka z počítačového pohledu vytváří typicky prostý text („plaintext“), popřípadě částečně formátovaný text. Styl vyznačování oprav je ale libovolný, učitele nic nepodporuje v tom, aby ve všech opravách dodržel stejné formátování. Je přitom logickým požadavkem, aby styl opravy vyjadřoval její typ, druh chyby, její závažnost apod., a to natolik konzistentně, aby student rozpoznal logiku tohoto systému a dobře se v dokumentu orientoval. Snažit se o toto plně ručně je však zbytečně náročné.

Změny v dokumentu jsou nevratné (jen pokud učitel použije obyčejnou tužku, lze je odmazat, avšak ani tak výsledek nebývá dokonalý). Je-li studentem odevzdaný text pouze kopií, není toto zásadním problémem, jsou-li však opravy prováděny přímo v originálním dokumentu, je nenávratně poškozen.²⁰

Není-li text pouze samoúčelným úkolem, tedy má-li být zveřejněn nebo použit jiným podobným způsobem, je nutné jej přepsat. To samozřejmě znamená zbytečnou duplikaci téže práce. Tento účel navíc nemusí být zřejmý již od začátku.

Je nutné fyzické setkání studenta a učitele, alespoň pro předání opravy (je samozřejmě také možné dokument vytisknout, opravit a naskenovat, ale pak se již můžeme ptát, proč tedy nebyl rovnou opraven na počítači). Jakkoli je toto běžné a vhodné, není to vždy možné, a možnost jednoduché a pružné elektronické komunikace by jistě byla přínosná. S tím souvisí i často dlouhá doba mezi odevzdáním úkolu a vyzvednutím jeho opravy.

Hodnocení

Papír a tužka jistě stále má mezi nástroji na opravu úkolů své místo a pro rychlé okamžité opravy nebo pro opravy dokumentů s netextovou povahou pravděpodobně ještě dlouho bude nejefektivnějším nástrojem. Pokud ale nepožadujeme maximální jednoduchost ani vysoký stupeň univerzality, je jistě vhodné použít některý ze sofistikovanějších nástrojů.

5.2 Textové procesory

Mnozí učitelé si povšimli nástupu informačních technologií a pro opravu úkolů – alespoň těch elektronicky odevzdaných – používají nějaký textový procesor. Jedinými dvěma typickými zástupci této kategorie jsou Microsoft Word [12] a OpenOffice.org Writer [23].²¹

Oba tyto programy podporují verzování (pod názvem sledování změn, správa verzí apod.), kdy v textu vyznačují, jaké úpravy byly provedeny. Jde ovšem o programy univerzální, proto je jejich záběr širší, ale nejsou nijak speciálně uzpůsobeny pro opravu domácích úkolů.

Výhody

Textové procesory již nabízejí pohodlný a efektivní způsob opravování elektronických textových dokumentů. Velice jednoduše, pouze zapnutím příslušné funkce, se procesor přepne do režimu sledování změn. Od této chvíle se každá provedená změna zaznamenává, takže učitel normálně opraví text, jako by jej opravoval například sám po sobě, a výsledkem je dokument s vyznačením provedených oprav. Programy navíc poskytují čím dál tím více pokročilých funkcí – kontrolu pravopisu, někdy i kontrolu gramatiky, vkládání komentářů, export do různých formátů a podobně. Dobrou funkcí také je, že student může jednotlivé úpravy přijmout nebo odmítnout.

²⁰ Lze se pozastavit nad tvrzením, že oprava dokument poškodí. Přinejmenším z čistě vizuálního hlediska, tj. bez ohledu na textový obsah, je to ale pravda. Vzhledově je krasopisně napsaný text, byť třeba s gramatickými chybami, jistě lepší, než tentýž text s červenými vpisky.

²¹ K dipozici jsem měl program Word 2007 a OpenOffice.org Writer 3. Existují samozřejmě i další takové programy, ale jimi nabízené služby jsou typicky podmnnožinou výše uvedených, proto je zbytečné se jimi detailněji zabývat.

Téměř každý uživatel má na svém počítači některý z procesorů instalován, pořizovací náklady lze proto zanedbat, stejně jako čas potřebný k tomu, aby se uživatel naučil aplikaci používat, za vyřešený lze považovat i problém formátu, neboť formát Microsoft Word Document File (DOC) se stal de facto standardem pro předávání dokumentů.

Nevýhody

Přestože jde o nástroje velice kvalitní, stále nejsou pro požadovaný úkon ideální. Jde o nástroje všestranné, proto neobsahují (a snad by ani obsahovat neměly) funkce specializované na opravu domácích úkolů. Podívejme se tedy, které vlastnosti implementace verzování v obou aplikacích opravu domácích úkolů znepřijemňují.

Jako největší problém vidím, že oba textové procesory chápou prováděné opravy pouze jako editace dvou typů – smazání textu a vložení textu. Přitom logicky existují minimálně čtyři typy editací – smazání, vložení, změna a změna pořadí. Smazání i vložení obě aplikace zobrazují srozumitelně. Změna části textu (např. náhrada jednoho slova jiným) je ale prezentována jako odstranění slova a vložení jiného, a změnu pořadí slov, či obecněji přesuny úseků textu, nelze vůbec vyznačit (v dokumentu se tato operace chová jako smazání slova na jednom místě a vložení nového slova na jiném místě). Takto zachycené editace ale nepostihují provedenou opravu, minimálně ze sémantického hlediska – např. prohození dvou slov („obecná žížala“ → „žížala obecná“) zkrátka není smazání jednoho slova „žížala“ a vložení jiného slova „žížala“.

Dalším významným problémem je, že opravy nelze zanořovat do sebe. Například u změny pořadí slov přitom jde o logický požadavek – slovo může být na špatném místě a ještě obsahovat např. pravopisnou chybu (Byl spokojený sem. → Byl jsem spokojený.). V popisovaných aplikacích ale místo toho vždy „převládne“ nejvnějšnější oprava, vnořené opravy jsou tím ztraceny.

Také nelze žádným vestavěným způsobem označit chybné místo bez uvedení správné varianty, je nutné použít nějaké vlastní fyzické formátování – tato funkce přitom logicky souvisí s ostatními opravami a bylo by tedy vhodnější, kdyby byla vyznačena stejným způsobem. V obou programech je ale pouze možné takovou část textu smazat, že by se tam měl vložit jiný text lze zapsat pouze komentářem.

Další nevýhody uvedu již stručněji:

- Není jednoduše možné přecházet k verzi před opravou a po ní.
- V pokročilejších funkcích může být problémem nekompatibilita mezi různými kancelářskými balíky.
- Export je poněkud rozpačitý (je možné, že v novějších verzích se zlepší).
- U opravy je možné vložit textový komentář, žádné další metainformace ale podporované nejsou.

Některé problémy jsou pak specifické pouze pro jednu z aplikací:

Microsoft Word

- Více oprav blízko u sebe, zvláště na stejném řádku, nelze pouhým pohledem dobře rozlišit. Při větším množství oprav se pak dokument rychle stává nepřehledným.

- Styl označení smazaného textu a komentářů je podobný, ale je výrazně odlišný od stylu označení vloženého textu. Má to svou logiku (přímo v textu je právě ten text, který je správně) a je to pohodlné pro provádění opravy, ale zhoršuje to orientaci studenta v dokumentu, zejména v případě oprav typu změna nebo změna pořadí.
- Může být považováno za nevýhodu, že celý softwarový balík MS Office včetně používaných formátů dokumentů je proprietární.

OpenOffice.org Writer

- Komentáře lze vkládat dvěma způsoby, z nichž ani jeden není ideální.
 - Prvním způsobem je vložení komentáře k opravě. Tento způsob je dobrý, ale komentář se bohužel téměř nikde nezobrazuje (student musí vynaložit nepřiměřené úsilí, aby si přečetl komentář k provedené opravě).
 - Druhým způsobem je vložení poznámky. Ta ale zabere zbytečně mnoho místa, takže jich nelze použít mnoho.
- Rozšíření programu není tak vysoké, aby bylo možné jeho nativní formáty bez obav používat pro výměnu dokumentů. Naštěstí program velmi dobře pracuje i s formátem DOC [11].

Hodnocení

Základní podpora korektur je dobrá, pro texty s nízkou frekvencí chyb a pro méně náročné uživatele jde o postačující nástroj; stejně tak pro někoho, kdo texty opravuje pouze příležitostně a nechce si za tím účelem pořizovat specializovaný program. Pro učitele z povolání, kteří často dostávají písemné domácí úkoly od studentů v elektronické podobě, a další lidi, kteří často provádějí korekturu textů, ale rozhodně stojí za zvážení pořízení specializovaného softwaru.

5.3 Specializovaný software

Jak jsem ukázal, všechny výše uvedené nástroje mají nevýhody, které brání jejich skutečně efektivnímu nasazení na opravu domácích úkolů. Lze tedy očekávat, že by bylo nejvhodnější použít software na to přímo specializovaný.

Z důvodu vysoké pořizovací ceny a absence zkušebních verzí jsem bohužel většinu potenciálně vhodných programů nemohl vyzkoušet, přesto jsem nabídku zmapoval dostatečně detailně na to, abych si dovolil tvrdit, že software OprDU je dílem unikátním. Hodiny strávené hledáním na internetu, česky i anglicky, nepřinesly žádný výsledek – další podobná aplikace s největší pravděpodobností neexistuje.

TurnItIn

Tato aplikace je primárně určená k odhalování plagiátů – učitelům umožňuje odhalit, zda je studentem odevzdaná práce zkopírovaná z jiné práce, studentům pak pomáhá vyhnout se neúmyslnému plagiátorství.

Ve své druhé verzi, která by měla vyjít v průběhu roku 2010, by dle informací na webových stránkách²² měla navíc nabízet zde požadovanou funkcionalitu, tj. provádění korektur dokumentů, vzájemnou komunikaci studenta a učitele, apod. V současnosti bohužel nelze získat podrobnější informace o konkrétních plánovaných funkcích aplikace.

Webkorr

Tento nástroj je možná nejpodobnější aplikaci OprDU Klient. Podle informací na jeho domovské stránce slouží pro editory textů na webových stránkách, kteří potřebují zaměstnavateli nebo zákazníkovi předvést provedenou práci, tj. umožňuje zobrazit, jaké změny editor na stránce provedl.

Bohužel se mi tuto aplikaci nepodařilo stáhnout, neboť všechny nalezené odkazy pro její stažení jsou nefunkční.²³

Software pro studenty

Příbuzné nástroje, se kterými jsem se setkal nejčastěji, slouží především studentovi, kterému během práce na dokumentu poskytují různé pokročilé funkce, počínaje kontrolou pravopisu a pokročilými gramatickými a stylistickými analýzami konče. Jejich výsledkem je ale pouze správný text, provedené úpravy nelze zjistit jinak než srovnáním opraveného textu s textem původním, o nějakých metainformacích ani nemluvě.

Jde o software jistě velice užitečný, ale bohužel náš problém neřeší. Svým zaměřením odpovídá aplikaci OprDU Student (tedy té části určené pro vytváření dokumentů). Přesto je možné i studentům, kteří jsou uživateli systému OprDU, použití takového software doporučit, neboť oproti OprDU Studentovi nabízí mnohé funkce, z nichž některé mohou být relativně jednoduše doplněny i do OprDU Studenta (např. kontrola pravopisu podle slovníku), ale jiné pravděpodobně v dohledné době jeho součástí nebudou.²⁴

Drobnou nevýhodou je, že všechny nalezené nástroje jsou placené a jejich cena se pohybuje kolem 100 amerických dolarů. Taková částka je přijatelná pro copywritery a podobné osoby, které se živí psaním textů, avšak málokterý student ji pravděpodobně bude chtít investovat do kvality svých domácích úkolů. Navíc tyto programy nemívají zkušební verzi.

Dvěma největšími zástupci této kategorie jsou WhiteSmoke Writer²⁵ a Ginger.²⁶

5.4 OprDU

Jako reakce na chybějící software, který by splňoval všechny požadavky na provádění oprav domácích úkolů, byl vyvinut software OprDU. Při jeho vývoji mezi hlavní cíle patří efektivita provádění oprav, uživatelská přívětivost a dostatečné pokrytí celé problematiky.

22 Informace o programu lze nalézt na adrese <http://www.turnitin.com/>.

23 Informace o programu lze nalézt na adrese <http://webkorr.smartcode.com/info.html>.

24 Tyto programy často nabízejí například databázi vzorových textů, jazykové příručky a návody a pokročilé jazykové funkce (kontrola gramatiky, slovník synonym a idiomů apod.), jejichž vývoj je časově i finančně velice nákladný.

25 Informace o programu lze nalézt na adrese <http://www.whitesmoke.com/>.

26 Informace o programu lze nalézt na adrese <http://www.gingersoftware.com/>.

Výhody

Software OprDU se snaží odstranit výše zmíněné nevýhody alternativních řešení, při zachování co největšího počtu jejich výhod.

- Datový formát pro dokumenty je jednoduchý XML formát, které umožňuje vyznačení oprav všech čtyř typů. Specifikace tohoto formátu je veřejná.
- U každé opravy lze nastavit doplňující informace (metainformace) – komentář nebo alternativní opravu.
- Dokument je možné pomocí XSLT skriptů exportovat do různých formátů. Mechanismus exportu je snadno rozšiřitelný i upravitelný.
- V dokumentu je obsažen původní text i opravy, pomocí XSLT skriptů lze proto snadno zobrazit původní text, text s opravami nebo text po aplikování oprav bez modifikace dokumentu.
- Řešení je vyvíjeno jako zdarma dostupné, s veřejným zdrojovým kódem.²⁷

Libovolnou část (pochopitelně kromě opravy samotné) lze provést i jiným způsobem – student například může text zaslat v příloze e-mailu, a učitel jej do ODU formátu převede až na svém počítači. Stejně tak může učitel po provedení opravy text vytisknout a předat studentovi ve fyzické podobě, pokud je to v dané situaci vhodnější. Navíc díky veřejným specifikacím rozhraní i zdrojovým kódům může být každá část řešení implementována i jinými programátory.

Nevýhody

Systém OprDU přes všechny své výhody stále má i několik nevýhod, z nichž některé jsou inherentní, jiné mohou být v budoucnosti odstraněny.

Jednou z nevýhod je, že OprDU není rozšířené, pro jeho použití je tedy nutné provést instalaci (příčemž instalace serveru je určena pro pokročilejší uživatele) a je nutné se s ním naučit zacházet. Instalace OprDU Klienta ale je přístupná pro běžného uživatele a ovládání je do značné míry intuitivní.

Další nevýhodou je absence exportu do PDF, které má oproti RTF některé výhody, zejména garantované shodné zobrazení na všech počítačích. Tento nedostatek lze ale snadno obejít pomocí tisku na virtuální PDF tiskárnu (viz Export dokumentu v Kapitole 9.1).

Za nevýhodu lze též považovat závislost na platformě .NET Frameworku a tedy OS Windows. Je ale nutné poukázat na to, že platformně závislá je pouze jedna část řešení, a to aplikace OprDU Klient. Tato platforma byla zvolena pro dobrou podporu používaných technologií, jako je XML, XSLT, Web Services nebo RTF. U cílové skupiny uživatelů OprDU Klienta (učitelé) lze předpokládat vysoké zastoupení tohoto operačního systému; .NET Framework postačující verze je pak již součástí nových verzí Windows. Ostatní uživatelé mohou využít emulační software; ve vývoji je také platformně nezávislá verze .NET Frameworku pod názvem Mono,²⁸ která bohužel v současné verzi není pro spuštění OprDU Klienta postačující.

27 Vše bude zdarma ke stažení na webu produktu <http://oprdu.nikde.eu>, kde bude po registraci také k dispozici instalace OprDU serveru.

28 Informace o projektu lze nalézt na adrese <http://www.mono-project.com/>

Hodnocení

Ze všech dostupných nástrojů pokrývá OprDU problematiku opravy domácích úkolů nejkomplexněji. Na drobné nebo zřídka prováděné opravy postačuje použití některého z výše uvedených jednodušších řešení, v ostatních případech je ale software OprDU vyhovujícím nástrojem.

Kapitola 6

Závěr

Výsledkem práce je komplexní řešení problematiky opravy domácích úkolů, které je mírou pokrytí unikátní. Byl vytvořen klient, který umožňuje provádění korektur textů, komunikaci se serverem, export do několika formátů a další funkce (kontrola pravopisu, makra). Byl navržen datový formát XML, který popisuje opravy provedené v dokumentu a podporuje export. Také byl implementován server, který spravuje dokumenty a uživatelské účty a komunikuje s klientskými programy. Pro studenty byl vytvořen tenký klientský program, který umožňuje odesílání domácích úkolů a přijímání jejich korektur. Server a klientské programy si předávají dokumenty a další data pomocí webových služeb. Vytvořený systém splňuje a dokonce mírně rozšiřuje zadání práce.

Vývoj OprDU je a bude motivovaný praxí, je určen pro reálné nasazení v případech popsaných v Kapitole 2. To vede k neustálému zdokonalování systému. Při testování v praxi se komplexní systém OprDU v současné verzi ukázal jako efektivní nástroj pro opravu textů, vyhovující učitelům svou jednoduchostí a srozumitelností. V budoucnosti by bylo dobré do aplikace doplnit pokročilejší lingvistické techniky, zejména kontrolu gramatiky.

Kapitola 7

Seznam použité literatury

- [1] Bradley, N.: *XML: kompletní průvodce*, Grada, Praha, 2000.
- [2] Delisle, M.: *phpMyAdmin*, 2003 - 2009, dostupné online: <http://www.phpmyadmin.net/>, cit. 2.9.2009
- [3] Jicha, R.: *Salted hash - další krok ke zvýšení bezpečnosti*, 2005, dostupné online: <http://interval.cz/clanky/salted-hash-dalsi-krok-ke-zvyseni-bezpecnosti/>, cit. 4.9.2009
- [4] Komunita vývojářů: *The Apache Software Foundation*, 1999 - 2010, dostupné online: <http://apache.org/>, cit. 20.5.2010
- [5] Kosek, J.: *PHP - tvorba interaktivních internetových aplikací: podrobný průvodce*, Grada, Praha, 1999.
- [6] Kosek, J.: *XML - eXtensible Markup Language*, 1999, dostupné online: <http://www.kosek.cz/clanky/xml/>, cit. 20.8.2009
- [7] Kosek, J.: *XML pro každého: podrobný průvodce*, Grada, Praha, 2000.
- [8] Krug, S.: *Webdesign - Nenutíte uživatele přemýšlet!*, Computer Press, Brno, 2003.
- [9] Kuenning, G., Atkinson, K.: *Spell Checking Oriented Word Lists (SCOWL)*, 2004, dostupné online: , cit.
- [10] Lerdorf, R. a kol.: *PHP*, 2001, dostupné online: <http://php.net>, cit. 2.9.2009
- [11] Microsoft Corporation: *Microsoft Office Word 97-2007 Binary File Format (.doc) Specification*, 2007, dostupné online: [http://download.microsoft.com/download/0/B/E/0BE8BDD7-E5E8-422A-ABFD-4342ED7AD886/Word97-2007BinaryFileFormat\(doc\)Specification.pdf](http://download.microsoft.com/download/0/B/E/0BE8BDD7-E5E8-422A-ABFD-4342ED7AD886/Word97-2007BinaryFileFormat(doc)Specification.pdf), cit. 28.6.2010
- [12] Microsoft Corporation: *Microsoft Word 2010*, 2010, dostupné online: <http://office.microsoft.com/cs-cz/word/>, cit. 28.6.2010
- [13] Microsoft Corporation: *MSDN*, 2010, dostupné online: <http://msdn.microsoft.com>, cit. 10.5.2009 - 3.8.2010
- [14] Microsoft Corporation: *Rich Text Format (RTF) Specification, version 1.6*, 1999, dostupné online: [http://msdn.microsoft.com/en-us/library/aa140277\(office.10\).aspx](http://msdn.microsoft.com/en-us/library/aa140277(office.10).aspx), cit. 16.8.2009
- [15] Microsoft Corporation: *The C# Language*, 2001 - 2010, dostupné online: <http://msdn.microsoft.com/en-us/vsharp/aa336809.aspx>, cit. 10.5.2009 - 3.8.2010
- [16] Microsoft Corporation: *.NET Framework Developer Center*, 2007 - 2010, dostupné online: [http://msdn.microsoft.com/cs-cz/netframework/default\(en-us\).aspx](http://msdn.microsoft.com/cs-cz/netframework/default(en-us).aspx), cit. 10.5.2009 - 3.8.2010

- [17] National Institute of Standards and Technology: *Secure Hash Standard*, 2002, dostupné online: <http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf>, cit. 4.9.2009
- [18] Nichol, S.: *Introduction to NuSOAP*, 2004, dostupné online: <http://www.scottnichol.com/nusoapintro.htm>, cit. 3.9.2009
- [19] NuSphere Corporation: *NuSOAP - SOAP Toolkit for PHP*, 2002 - 2009, dostupné online: <http://nusoap.sourceforge.net/>, cit. 3.9.2009
- [20] Oracle: *Java*, 2010, dostupné online: <http://www.java.com>, cit. 18.5.2010
- [21] Oracle: *MySQL*, 1997 - 2009, dostupné online: <http://www.mysql.com/>, cit. 2.9.2009
- [22] Oracle: *MySQL 5.1 Reference Manual*, 1997, dostupné online: <http://dev.mysql.com/doc/refman/5.1/en/index.html>, cit. 2.9.2009
- [23] Oracle: *OpenOffice.org Writer*, 2010, dostupné online: <http://www.openoffice.org/product/writer.html>, cit. 28.6.2010
- [24] Troelsen, A. W.: *Pro C# 2008 and the .NET 3.5 platform*, Apress, New York, 2007.
- [25] W3C: *Document Object Model (DOM) Level 1 Specification*, 1998, dostupné online: <http://www.w3.org/TR/REC-DOM-Level-1/>, cit. 14.6.2010
- [26] W3C: *Extensible Markup Language (XML) 1.0 (Fifth Edition)*, 2008, dostupné online: <http://www.w3.org/TR/2008/REC-xml-20081126/>, cit. 20.8.2009
- [27] W3C: *HTML 4.01 Specification*, 1999, dostupné online: <http://www.w3.org/TR/html401/>, cit. 24.8.2009
- [28] W3C: *Web Services Activity*, 2002 - 2007, dostupné online: <http://www.w3.org/2002/ws/>, cit. 3.9.2009
- [29] W3C: *Web Services Description Language (WSDL) 1.1*, 2001, dostupné online: <http://www.w3.org/TR/wsdl>, cit. 4.9.2009
- [30] W3C: *XSL Transformations (XSLT)*, 1999, dostupné online: <http://www.w3.org/TR/xslt>, cit. 29.3.2010
- [31] W3Schools: *XML Tutorial*, 2010, dostupné online: <http://www.w3schools.com/xml/>, cit. 20.8.2009
- [32] W3Schools: *XSLT Tutorial*, 1999-2010, dostupné online: <http://www.w3schools.com/xsl/>, cit. 29.3.2010
- [33] Wille, C.: *Storing Passwords - done right!*, 2004, dostupné online: <http://www.aspheute.com/english/20040105.asp>, cit. 4.9.2009

Kapitola 8

Příloha A: Obsah přiloženého disku

Obsah přiloženého média, podle složek:

- OprDUklient
 - `bin` – přeložená aplikace, lze spustit přímo z disku
 - `install` – instalátor, obsahuje i instalaci .NET Frameworku 3.5 a PDFCreatoru
 - `src` – zdrojové kódy (včetně souborů `sln` a `csproj`)
 - `scowl` – anglický slovník (wordlist) využitý pro kontrolu pravopisu
- OprDUstudent
 - `bin` – přeložená aplikace, lze spustit přímo z disku
 - `install` – instalace prostředí Java
 - `dist` – distribuční balíček, obsahuje i dokumentaci apod.
 - `src` – zdrojové kódy (včetně souboru `build.xml` pro přeložení)
 - `lib` – knihovny využité v aplikaci
- OprDUserver
 - `www` – instalace OprDU Serveru se všemi součástmi:
 - `oduserver` – OprDU Server pro učitele
 - `oduserverStudent` – OprDU Server pro studenty
 - `student` – webový OprDU Student
 - `teacher` – webové rozhraní pro učitele
 - `common` – funkce sdílené výše uvedenými součástmi
 - `oprdu.sql` – příkazy pro vytvoření tabulek v databázi MySQL
 - `install` – instalační soubor lokálního serveru EasyPHP
 - `nusoap` – knihovna použitá pro implementaci webových služeb
- `text` – text bakalářské práce v PDF
- `example` – vzorová data, která lze použít na testování

Kapitola 9

Příloha B: Uživatelské příručky

OprDU je aplikačním software, uživatelské příručky jsou proto důležitou součástí projektu.

9.1 Příručka pro učitele

Před prvním spuštěním

Instalace OprDU Klienta

Pro používání OprDU Klienta je zapotřebí počítač s operačním systémem Microsoft Windows verze alespoň Windows XP a instalovaný .NET Framework verze alespoň 3.5.

Instalace .NET Frameworku je součástí disku (viz Kapitola 8), pro instalaci jej stačí spustit.

Samotný program OprDU Klient není nutné instalovat, lze jej spouštět přímo z CD. Pro komfortnější používání programu je ale doporučeno jej instalovat na pevný disk počítače, buď spuštěním instalátoru (viz Kapitola 8) nebo zkopírováním složky `bin`.

Registrace na OprDU Serveru

Pro plnohodnotné využití možností, které OprDU nabízí, je vhodné využít služeb OprDU Serveru. V případě použití serveru totiž veškerá komunikace se studentem probíhá prostřednictvím programu OprDU Klient – stažení dokumentu od studenta do počítače, ukládání dokumentu i jeho odevzdání studentovi. Bez využití serveru je nutné dokument ručně vložit do programu (např. zkopírovat z e-mailu), ukládat si jej na svůj pevný disk, a po jeho opravení jej exportovat a nějakým způsobem předat studentovi.

Pokud máte administrátora serveru, jistě Vám sdělí potřebné pokyny k registraci. V opačném případě využijte veřejný server na adrese <http://oprdu.nikde.eu>.

Pro registraci stačí vyplnit jméno, heslo a e-mailovou adresu. E-mailová adresa se stává Vaším uživatelským jménem, pomocí ní a zvoleného hesla se budete přihlašovat do systému. Jako jméno vyplňte Vaše jméno tak, jak chcete, aby se zobrazovalo studentům. Jméno i heslo je možné později změnit, e-mailovou adresu ale nikoli, proto věnujte jejímu správnému vyplnění zvláštní pozornost.

Před opravou dokumentu

Přihlášení k serveru / práce offline

Spustěte program.

Pokud již máte registraci na OprDU serveru a máte funkční připojení k internetu, vyplňte přihlašovací údaje a přihlaste se k serveru (adresu serveru Vám sdělí Váš administrátor k serveru; máte-li registraci na výše zmíněném veřejném serveru, je správná hodnota <https://oprdu.nikde.eu/oduserver/>).

Pokud nemáte zřízenou registraci, nebo se z nějakého důvodu nemůžete nebo nechcete připojit k serveru, zvolte Pracovat offline. K serveru se můžete přihlásit i později.

Seznam dokumentů a stavy dokumentů

Po přihlášení k serveru se načte seznam dostupných dokumentů. Zvolte požadovaný dokument a akci, kterou chcete provést – opravovat dokument, zobrazit dokument, dokončit opravu dokumentu apod.

Pokud je dokument „odevzdán“, musíte si jej před započítím oprav „vyžádat pro opravu“, čímž si dokument zamknete pro sebe a zároveň informujete studenta, že začínáte pracovat na opravě – dokument přejde do stavu „opravuje se“.

Když opravu dokumentu dokončíte, tlačítkem „dokument je opraven“ přepnete dokument do stavu „opravený“ – od tohoto okamžiku Vám již program neumožní dokument dále opravovat. Nyní se opravený dokument zpřístupní studentovi; pokud student nepoužívá OprDU, dokument exportujte do vhodného formátu (např. RTF) a zašlete studentovi e-mailem, vytiskněte apod. Pro zaslání e-mailu studentovi lze použít i tlačítko v seznamu dokumentů – spustí se Váš výchozí e-mailový klient s předvyplněnou adresou studenta; chcete-li ale e-mailem zaslat studentovi dokument, musíte jej k tomuto e-mailu ručně přiložit.

Jakmile si student v systému OprDU dokument vyzvedne, přejde dokument do stavu „opravený a vyzvednutý“. Pokud student nepoužívá systém OprDU, můžete dokument do tohoto stavu přepnout ručně – v tomto případě se stav zobrazí jako „opravený (a vyzvednutý)“.

Otevření dokumentu

Můžete buď otevřít dokument ze seznamu dokumentů ze serveru, nebo otevřít dokument z disku. Lze jej buď opravovat, tedy otevřít pro čtení i zápis, nebo jen zobrazit, tj. otevřít pouze pro čtení.

Vytvoření nového dokumentu

Pokud student nevložil dokument do systému OprDU, ale předal Vám jej jinou cestou (např. e-mailem), musíte jej do systému vložit sami. Slouží k tomu volba Nový dokument (klávesová zkratka Ctrl+N).

Oprava dokumentu

Pokud máte otevřený dokument pro opravy, dostáváte se k hlavní funkci programu OprDU Klient. Nyní můžete do dokumentu vyznačit jednotlivé opravy a případně přidat komentáře.

OprDU rozlišuje čtyři základní typy oprav – tři opravy jednoduché (*Změna*, *Vložení*, *Smazání*), a pak *Změnu pořadí slov*.

Jednoduché opravy

Volba opravy

Podle typu chyby vyberte správný typ opravy. Jak opravu provést se dozvíte v následujících odstavcích – pro provedení opravy typu *Změnit* nebo *Vložit* lze použít tzv. Přirozené zadávání, pro všechny typy jsou pak definované Klávesové zkratky.

- Je-li někde chyba, označte chybnou část textu a použijte opravu *Změnit* (bilina → bylina, bysem → bych).
- Pokud je nějaké písmeno, slovo nebo jiná část textu navíc, označte úsek textu, který chcete vymazat, a použijte opravu *Smazat* (votěřel → otevřel, ~~víee~~ chytřejší → chytřejší).
- Chybí-li slovo či jeho část, použijte opravu *Vložit* (jdu na Harry Potter → jdu na Harryho Pottera, jdu doktorovi → jdu k doktorovi). Pokud přidáváte jen část slova, nic neoznačujte, pouze umístěte kurzor tam, kam se má provést vkládání. Pokud chcete vložit celé slovo nebo několik slov, označte mezeru, na jejímž místě by se měla slova objevit – jinak bude vložený text „nalepený“ na slovo před nebo za ním.

Přirozené zadávání

Opravy typu *Vložit* a *Změnit* lze zadávat přímo „psáním do dokumentu“ – typ opravy je zvolen automaticky. Pokud není vybrán žádný text (nebo je vybrána pouze mezera – viz výše) a začnete psát, bude vytvořena oprava typu *Vložit* a napsaný text se stane její součástí. Pokud je vybrán nějaký text, je obdobně vytvořena oprava typu *Změnit*.

Přirozené zadávání se aktivuje stiskem libovolné klávesy s písmenem anglické abecedy (které se stane součástí opravy). Má-li oprava začínat jiným znakem (například číslem), aktivujte zadávání stiskem mezerníku (nestane se součástí opravy) nebo klávesovou zkratkou.

Klávesové zkratky

Všechny opravy mají také definované klávesové zkratky. U oprav *Vložit* a *Změnit* jde o alternativu k přirozenému zadávání.

- *Smazat*: klávesa Delete (Del), Backspace (←), mínus (-) nebo Ctrl+D
- *Vložit*: klávesa Insert (Ins), plus (+) nebo Ctrl+F
- *Změnit*: klávesa hvězdička (*) nebo Ctrl+G
- *Změnit* opravu: klávesa Enter

V pravém panelu se po zvolení opravy zobrazí editační panel s názvem zvolené opravy. U opravy typu *Vložit* vyplňte do prvního políčka správný text, u *Změnit* sem můžete zadat správný text, kterým se má nahradit text chybný. Další položky jsou nepovinné, viz níže. Opravu potvrdíte tlačítkem Použít (nebo stisknutím klávesy Enter), případně ji můžete stornovat tlačítkem Zrušit (nebo klávesou Escape).

Změna pořadí slov

Jsou-li slova ve špatném pořadí (Jsem domů přišel. → Přišel jsem domů.), označte celý úsek, který je potřeba opravit, a vyvolejte opravu *Změna pořadí slov*. Opravu lze vyvolat klávesou lomeno (/) nebo klávesovou zkratkou Ctrl+H.

V otevřeném okně zadejte pro některá slova jejich pozici – sledujte náhled opravy v dolní části okna.

Komentáře

Každá oprava u sebe může mít komentář, který se zobrazí za opraveným textem. Navíc můžete vkládat i komentáře samostatné – v takovém případě označte část textu, které se komentář týká, a vložte komentář. Klávesová zkratka pro vložení komentáře je Ctrl+K.

Mezery

Jistě si povšimnete, že u každé opravy se zobrazují zaškrťovací políčka „mezera před“ a „mezera za“. Nastavuje se zde, zda má nebo nemá být oprava oddělená od okolního textu mezerami. Pokud je všechno tak jak má být, nemusíte si jich vůbec všimnout – program správné nastavení zvolí automaticky. Někdy se ale stane, že se splete, a text opravy se nalepí na jiný text, od kterého by měl být oddělen, nebo naopak se objevuje mezera tam, kde by být neměla (tímto případem není oprava typu změnit, kde špatný a správný text je od sebe oddělen mezerou vždy). V takovém případě správného zobrazení docílíte vhodným nastavením těchto přepínačů.

Úprava již vložené opravy

Chyby dělá i učitel, a proto je občas potřeba již zadanou opravu změnit nebo zrušit. V takovém případě označte opravu, kterou chcete změnit (postačuje umístění kurzoru dovnitř opravy) a zvolte Změnit opravu (klávesová zkratka Ctrl+M).

V pravém panelu se otevře editační panel s vlastnostmi vybrané opravy; pokud se ve výběru nachází více oprav, otevře se pro každou samostatný editační panel.

V editačním panelu můžete změnit libovolné vlastnosti opravy. Změny uložíte tlačítkem Použít (nebo klávesou Enter), případně můžete opravu zcela zrušit.

Kontrola pravopisu

Program umožňuje kontrolu pravopisu podle slovníků. V současné době má program instalovaný pouze anglický slovník.

Po spuštění kontroly se otevře okno kontroly pravopisu, ve kterém jsou vypsána všechna slova, která nebyla ve slovníku nalezena. Po kliknutí na slovo je v dokumentu nalezen jeho první výskyt a slovo je označeno. Vyhledávání probíhá v opraveném textu, nejsou proto zobrazena slova, která už jste opravil(a).

Makra

Makra jsou předdefinované opravy pod klávesovými zkratkami Alt+Shift+písmeno. Nastavíte je v menu Makra, vyvolat je můžete také z menu nebo klávesovou zkratkou. Lze předdefinovat opravy typu *Změnit*, *Vložit*, *Smazat* a *Komentář*, přičemž lze nastavit stejné parametry jako při vkládání opravy.

Při nastavování mezer jsou, na rozdíl od zadávání opravy při korektuře dokumentu, zaškrťovací políčka třístavová. Výchozí „neurčitý“ stav (tj. políčko není ani prázdné, ani zaškrtnuté) znamená, že přítomnost mezery bude rozhodnuta až při aplikaci makra, stejným způsobem jako při vkládání běžné opravy.

Upozornění

Při vybírání textu pro opravu nelze použít klávesové zkratky Shift+Home a Shift+End pro přechod na začátek/konec odstavce (totéž platí pro Ctrl+Shift+Home, Ctrl+Shift+End) – vybírají totiž i znak začátku resp. konce řádku, zobrazí se proto upozornění o nepovolené opravě. Ostatní pokročilé klávesy fungují správně (Home, End, Ctrl+šipka, Ctrl+Shift+šipka apod.).

Export dokumentu

Dokument je možné exportovat do několika formátů, nejuniverzálnějším z nichž je *Rich Text Format* (RTF). Dokument v tomto formátu lze zobrazit na všech běžných platformách, v operačním systému Microsoft Windows např. pomocí programu Wordpad.

OprDU Klient přímo neumožňuje export do PDF, ale protože umožňuje tisk dokumentu, lze export do PDF provést pomocí tisku na virtuální PDF tiskárnu. Pokud takovou nemáte, můžete si nainstalovat program PDF Creator.²⁹

9.2 Příručka pro studenta (OprDU Student)

Pro práci v systému OprDU můžete využít desktopovou *Java aplikaci OprDU Student*, kterou budete spouštět na svém počítači, nebo *internetovou aplikaci OprDU Student*, ke které můžete přistupovat pomocí internetového prohlížeče. Můžete samozřejmě kombinovat použití obou aplikací – pokud budete využívat stále stejný OprDU Server, své dokumenty budete mít dostupné prostřednictvím obou aplikací.

Pro použití systému OprDU je třeba se registrovat na OprDU Serveru. Vše pak probíhá prostřednictvím programu OprDU Student – vytvoření dokumentu, jeho odeslání učiteli i prohlédnutí opraveného dokumentu. Pokud máte administrátora serveru, jistě Vám sdělí potřebné pokyny k registraci. V opačném případě využijte veřejný server na adrese <http://oprdu.nikde.eu>.

Pro registraci stačí vyplnit jméno, heslo a e-mailovou adresu. E-mailová adresa se stává Vaším uživatelským jménem, pomocí ní a zvoleného hesla se budete přihlašovat do systému. Jako jméno vyplňte Vaše jméno tak, jak chcete, aby se zobrazovalo učitelům. Jméno i heslo je možné později změnit, e-mailovou adresu ale nikoli, proto věnujte jejímu správnému vyplnění zvláštní pozornost.

²⁹ Instalační soubor je k dispozici na příloženém médiu nebo na <http://www.pdfforge.org/pdfcreator>

Použití internetové aplikace OprDU Student

Internetovou aplikaci najdete na adrese, na které jste se registroval/a do systému OprDU (nesdělí-li Vám Váš administrátor serveru jiné pokyny).

Seznam dokumentů

Po přihlášení k serveru se zobrazí seznam dokumentů, které můžete *Zobrazit*, dokumenty ve stavu „rozepsaný“ i *Upravit* a případně *Odevzdat* učiteli nebo *Smazat*.

Vytvoření nového dokumentu

Volba *Nový dokument* Vám umožní vytvořit nový text, které později odevzdáte svému učiteli. Název dokumentu vyplňte pozorně, není možné je později měnit, stejně jako zvoleného učitele. Můžete vyplnit i nepovinný komentář, který se poté zobrazuje Vám i učiteli opravujcímu dokument.

Po potvrzení tlačítkem *Vytvořit dokument* vložte nebo vepište obsah dokumentu. Dokument průběžně ukládáte – dokument je ukládán na server, takže je zachován i po ukončení zavření prohlížeče; v práci na rozpracovaném dokumentu je možné pokračovat i na jiném počítači nebo z Java aplikace OprDU Student.

Až bude dokument hotový, vyberte volbu *Odevzdat*.

Prohlížení dokumentu

Každý dokument, který není momentálně opravován učitelem, lze prohlížet (zvolte volbu *Zobrazit* u seznamu dokumentů). U opravených dokumentů máte několik možností zobrazení – s vyznačením oprav, bez vyznačení oprav, nebo tzv. bezchybný dokument, tj. dokument, kde byly použity všechny naznačené opravy.

Použití Java aplikace OprDU Student

Instalace OprDU Studenta

Klienta není nutné instalovat, stačí balíček rozbalit do libovolného adresáře. Pro běh programu je nutné mít nainstalovanou Java Virtual Machine alespoň verze 1.6.³⁰

Přihlášení k serveru

Spustíte program, vyplňte přihlašovací údaje a přihlaste se k serveru (adresu serveru Vám sdělí Váš administrátor k serveru; máte-li registraci na výše zmíněném veřejném serveru, je správná hodnota <https://oprdu.nikde.eu/oduserverStudent/>).

Pokud neprovedete přihlášení, můžete pracovat offline – aplikace Vám umožní prohlížet si ODU soubory uložené na Vašem disku.

³⁰ Instalační soubor je k dispozici na přiloženém médiu nebo na <http://www.java.com/>

Seznam dokumentů

Po přihlášení k serveru se načte seznam dostupných dokumentů, které můžete zobrazit, dokumenty ve stavu „rozepsaný“ i editovat a případně odevzdat. Pro otevření dokumentu lze použít klávesu Enter.

Vytvoření nového dokumentu

Volba Nový dokument (klávesová zkratka Ctrl+N) Vám umožní vytvořit nový text, které poté odevzdáte svému učiteli. Informace o dokumentu vyplňte pozorně, není možné je později měnit, stejně jako zvoleného učitele.

Poté vložte nebo vepište obsah dokumentu. Je možné dokument průběžně ukládat – dokument je ukládán na server, takže je zachován i po ukončení programu; dokonce je možné v práci na rozpracovaném dokumentu pokračovat na jiném počítači nebo v internetové aplikaci OprDU Student.

Až bude dokument hotový, vyberte volbu Odevzdat učiteli.

Prohlížení dokumentu

Každý dokument, který není momentálně opravován učitelem, lze prohlížet (zvolte volbu Zobrazit u seznamu dokumentů). Lze přepínat mezi třemi zobrazeními – text s opravami (výchozí, jsou vyznačeny provedené opravy), původní text (text před vyznačením oprav) a správný text (text, v němž byly opravy již aplikovány a tedy je teoreticky bez chyby).

9.3 Příručka pro administrátora serveru

Potřebná konfigurace

Nutnou podmínkou pro zprovoznění serveru je PHP a MySQL server, přičemž PHP musí být alespoň verze 4. Doporučena je podpora SSL (je doporučené, ale ne nutné, aby certifikát byl důvěryhodný). Server musí být dostupný pro studenty i učitele.

Postačuje-li server lokální, lze použít např. program EasyPHP,³¹ jehož instalace je popsána níže.

Po větším či menším přepsání zdrojových kódů je pochopitelně možná libovolná jiná konfigurace – viz podkapitola Přizpůsobení funkcí.

Instalace

Pro instalaci OprDU Serveru je potřebný HTTP server s podporou PHP a MySQL. Pokud takový server již máte, pokračujte na podkapitolu Instalace OprDU Serveru.

Budete-li OprDU Server používat pouze na jednom počítači, tj. nepotřebujete, aby byl dostupný ze sítě LAN ani ze sítě internetu, můžete si instalovat lokální server. Návod na instalaci jednoho takového serveru (s názvem EasyPHP) najdete níže.

³¹ Instalační soubor je k dispozici na přiloženém médiu nebo na www.easyphp.org/

Potřebujete-li server dostupný z více počítačů (prostřednictvím LAN nebo internetu), instalujte si k tomu určený server. Potřebné informace Vám poskytne správce Vaší sítě.

Instalace lokálního serveru (EasyPHP)

Tato část popisuje instalaci s použitím lokálního serveru, pro případ, že budete na OprDU server přistupovat pouze z jednoho počítače. Popisovaný postup platí pro OS MS Windows, uživatelé jiných operačních systémů si jistě poradí.

Je popisována instalace serveru pomocí balíčku EasyPHP, jehož instalační soubor je přiložen. Je ale možné použít obecně libovolný PHP a MySQL server.

1. Nainstalujte server spuštěním instalačního souboru.
2. Spusťte server, měl by signalizovat, že Apache i MySQL běží. Na některých systémech budete muset programu přidělit vyšší než výchozí oprávnění, jinak budou některé jeho funkce blokovány. Pro využití služeb OprDU Serveru budete muset vždy spustit EasyPHP, můžete proto v jeho nastavení zvolit automatické spouštění po spuštění operačního systému.
3. Ve složce, do které se server instaloval (nejčastěji `C:\Program Files\EasyPHP` nebo podobně), najděte adresář `www` a vytvořte v něm novou složku `oprdu`. Do této složky nakopírujete obsah složky `www` z instalačního disku.
4. Proved'te další kroky popsané v podkapitole Instalace.
5. Zadejte do webového prohlížeče adresu `http://localhost/oprdu/`, měla by se zobrazit úvodní stránka serveru.
6. Pro připojení k serveru z aplikace OprDU Klient používejte adresu serveru `http://localhost/oprdu/oduserver/`.

Instalace OprDU Serveru

1. Nakopírujte obsah složky `www` do zvolené složky na serveru.
2. Na svém MySQL serveru nechte vykonat příkazy v souboru `oprdu.sql`.
3. Zeditujte libovolné soubory dle potřeby, zejména:
 1. Soubor `index.php` v kořenové složce – jde o soubor, který se zobrazí jako hlavní stránka serveru, ve výchozí podobě je to rozcestník k jednotlivým službám.
 2. Soubory v podsložce `common`, zejména `func-mysql.php`, pokud používáte jiný SŘBD. Po příslušných úpravách tohoto souboru by měl být server plně funkční – problém mohou působit rozdíly v použité SQL syntaxi, v takovém případě je nutné příkazy v kódu najít a přepsat. Můžete nový soubor uložit pod jiným názvem, v takovém případě zeditujte i soubor `req-db.php` – zejména na šestém řádku mezi uvozovky vepište nové jméno souboru.

Nastavení

Provádí se editací souboru `settings.php`. všechny volby jsou okomentované, podrobnější komentář asi zasluhuje `$ODU['serverId']`. Součástí každého souboru v systému OprDU je jeho jedinečný identifikátor (`uid`), jehož součástí je zde nastavené `id` serveru (`uid` dokumentů totiž generuje výhradně OprDU server). Pro celosvětově jednoznačnou identifikaci dokumentu by toto `id` mělo být voleno jedinečně. Doporučuji proto zvolit náhodné, pokud možno devítimístné, číslo – číslo smí být i kratší, ale za žádných okolností nesmí být delší.

Adresy webových služeb

Pro přihlášení k serveru potřebují uživatelé znát správné adresy webových služeb. Ve výchozím stavu je adresa serveru pro učitele (tj. uživatele aplikace OprDU Klient) `<adresa serveru>/oduserver/` a pro studenty (aplikace OprDU Student) `<adresa serveru>/oduserverStudent/`. Pokud je tedy adresa serveru `http://www.example.com` a OprDU Server je v podsložce `/oprdu`, je adresa učitelského serveru `http://www.example.com/oprdu/oduserver/`.

Odstávka serveru

V případě odstávky serveru pokud možno v souboru `settings.php` změňte hodnotu volby `$ODU['serverStatus']`. Výchozí hodnota `'ok'` znamená, že server je plně funkční, jiný text znamená nefunkční server a nastavený text je pak zobrazován uživatelům jako doplňující informace – vhodný text proto může být např. `'Server je odstaven z důvodu udrzby do 21.12.2012 23:00.'`

Přizpůsobení funkcí

Funkce serveru lze libovolně přizpůsobovat případným specifickým požadavkům – nejčastěji půjde pravděpodobně o úpravu přístupu k databázi, přihlašování uživatelů a/nebo ukládání dokumentů.

Pro komunikaci s databází slouží funkce s předponou `db`, definované v souboru `/common/func-mysql.php`. Jejich úpravou lze server přizpůsobit jinému databázovému serveru. Přihlašování studentů i učitelů (předpona `login`) řídí skript `/common/func-login.php` a správu dokumentů (předpona `store`) skript `func-store.php`. Nezapomeňte, že tyto funkce volají nejen obslužné metody serverů webových služeb, ale i webové rozhraní aplikace OprDU Student.

Kapitola 10

Příloha C: Datová příloha XML

V této příloze je výpis několika XML dokumentů, které jsou klíčové pro systém OprDU.

10.1 ODU dokumenty

Oba dokumenty v této kapitole jsou vytvořené ručně a jsou opatřené komentáři.

XML Schema pro ODU

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" version="1.1">

  <!-- MAIN ELEMENTS -->

  <xs:element name="odu">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="head" />
        <xs:element ref="body" />
      </xs:sequence>
    </xs:complexType>
    <xs:key name="id">
      <xs:selector
        xpath="//chg | ../ins | ../del | ../wo | ../order | ../br" />
      <xs:field xpath="@id" />
    </xs:key>
  </xs:element>

  <xs:element name="head">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="uid"
          type="uid" />
        <xs:element name="title"
          type="xs:string" />
        <xs:element name="student"
          type="email" />
        <xs:element name="studentName"
          type="xs:string" minOccurs="0" />
        <xs:element name="created"
          type="xs:dateTime" />
        <xs:element name="comment"
          type="xs:string" minOccurs="0" />
        <xs:element name="teacher"
          type="email" />
        <xs:element name="teacherName"
          type="xs:string" minOccurs="0" />
        <xs:element name="status"
          type="xs:nonNegativeInteger" />
        <xs:element name="maxid">
          <xs:complexType>
```

```

        <xs:attribute name="id" type="id" use="required" />
    </xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>

<xs:element name="body">
    <xs:complexType mixed="true">
        <xs:sequence>
            <xs:element ref="idElement"
                minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
    </xs:complexType>
</xs:element>

<xs:element name="idElement" abstract="true"
    type="idElementType" nillable="true" />

<!-- FORMATTING ELEMENTS -->

<xs:element name="br" substitutionGroup="idElement">
    <xs:complexType>
        <xs:complexContent>
            <xs:extension base="idElementType">
                <!-- currently no extension -->
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
</xs:element>

<!-- CORRECTION ELEMENTS -->

<xs:element name="correction" substitutionGroup="idElement"
    abstract="true" type="correctionType" nillable="true" />

<xs:element name="chg" substitutionGroup="correction" type="chgType" />

<xs:element name="del" substitutionGroup="correction" type="delType" />

<xs:element name="ins" substitutionGroup="correction" type="insType" />

<xs:element name="wo" substitutionGroup="correction" type="woType" />

<xs:element name="c" substitutionGroup="idElement">
    <xs:complexType mixed="true">
        <xs:complexContent>
            <xs:extension base="idElementType">
                <xs:attribute ref="title" />
                <xs:attribute ref="ws" />
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
</xs:element>

<!-- CORRECTION ELEMENTS TYPES -->

<xs:complexType name="idElementType" abstract="true" mixed="true">
    <xs:attribute ref="id" use="required" />
</xs:complexType>

```

```

<xs:complexType name="correctionType" abstract="true" mixed="true">
  <xs:complexContent>
    <xs:extension base="idElementType">
      <xs:attribute ref="title" use="optional" />
      <xs:attribute ref="ws" use="optional" />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="chgType">
  <xs:complexContent mixed="true">
    <xs:extension base="correctionType">
      <xs:sequence>
        <xs:element name="r" type="xs:string" />
        <xs:element name="other" type="xs:string"
          minOccurs="0" maxOccurs="unbounded" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="delType">
  <xs:complexContent mixed="true">
    <xs:extension base="correctionType">
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="insType">
  <xs:complexContent mixed="true">
    <xs:extension base="correctionType">
      <xs:sequence>
        <xs:element name="r" type="xs:string" />
        <xs:element name="other" type="xs:string"
          minOccurs="0" maxOccurs="unbounded" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="woType" mixed="false">
  <xs:complexContent mixed="true">
    <xs:extension base="correctionType">
      <xs:sequence>
        <xs:element name="order" maxOccurs="unbounded">
          <xs:complexType mixed="true">
            <xs:choice minOccurs="0" maxOccurs="unbounded">
              <xs:element ref="correction" />
              <xs:element ref="c" />
            </xs:choice>
            <xs:attribute name="o"
              type="xs:nonNegativeInteger" />
            <xs:attribute ref="id" />
            <xs:attribute ref="ws" />
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

```

<!-- TYPES -->

<xs:simpleType name="uid">
  <xs:restriction base="xs:string">
    <xs:minLength value="4" />
    <xs:maxLength value="40" />
    <xs:pattern value="[a-zA-Z0-9]+" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="email">
  <xs:restriction base="xs:string">
    <xs:pattern
      value="[a-zA-Z0-9_\-\.]+@[a-zA-Z0-9_\-\.]+\.[a-zA-Z]{2,}" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="id">
  <xs:restriction base="xs:nonNegativeInteger">
    <!-- no current restriction -->
  </xs:restriction>
</xs:simpleType>

<!-- ATTRIBUTES -->

<xs:attribute name="id" type="id" />

<xs:attribute name="title" type="xs:string" />

<xs:attribute name="ws" default="all">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="all" />
      <xs:enumeration value="left" />
      <xs:enumeration value="right" />
      <xs:enumeration value="none" />
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>

</xs:schema>

```

XSLT skript odu2rtfTagged – transformace ODU dokumentu pro zobrazení v klientských programech

```

<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:msxsl="urn:schemas-microsoft-com:xslt" exclude-result-
prefixes="msxsl"
>

<xsl:output method="text" indent="no"/>

<!-- ROOT, HEAD, BODY -->

<!-- root -->
<xsl:template match="/">
  <!-- inner content -->
  <xsl:apply-templates/>

```



```

</xsl:template>

<!-- head -->
<xsl:template match="head">
  <!-- no information from head is exported -->
</xsl:template>

<!-- body -->
<xsl:template match="body">
  <!-- RTF header -->
  <xsl:text>{\rtf1\ansi\ansicpg1250\deff0\deflang1029{\fonttbl{\f0\fni
1\fcharset238 Microsoft Sans Serif;}}
{\colortbl ;\red255\green0\blue0;\red0\green127\blue0;}
\viewkind4\uc1\pard\v\f0\fs24 {\cf1 .} {\cf2 .} ##oduS@@ \v0
</xsl:text>
  <!-- inner content -->
  <xsl:apply-templates/>
  <!-- RTF footer -->
  <xsl:text>{\v ##oduE@@ \v0}
  </xsl:text>
</xsl:template>

<!-- FORMATTING -->

<!-- text -->
<xsl:template match="text()">
  <xsl:value-of select="normalize-space(self::text())"/>
</xsl:template>

<!-- headline level 1 -->
<xsl:template match="h1">
  <!-- odu tag open -->
  <xsl:text>{\v ##odu@@+h1 </xsl:text>
  <xsl:value-of select="@id" />
  <xsl:text>##oduend@@ \v0</xsl:text>
  <!-- formatting -->
  <xsl:text>{\fs50 </xsl:text>
  <!-- inner content -->
  <xsl:apply-templates/>
  <!-- newline -->
  <xsl:text>\par}</xsl:text>
  <!-- odu tag close -->
  <xsl:text>\v ##odu@@-h1 </xsl:text>
  <xsl:value-of select="@id" />
  <xsl:text>##oduend@@ \v0}</xsl:text>
</xsl:template>

<!-- headline level 2 -->
<xsl:template match="h2">
  <!-- odu tag open -->
  <xsl:text>{\v ##odu@@+h2 </xsl:text>
  <xsl:value-of select="@id" />
  <xsl:text>##oduend@@ \v0</xsl:text>
  <!-- formatting -->
  <xsl:text>{\fs40 </xsl:text>
  <xsl:apply-templates/>
  <!-- inner content -->
  <xsl:apply-templates/>
  <!-- newline -->
  <xsl:text>\par}</xsl:text>
  <!-- odu tag close -->

```

```

    <xsl:text>\v ##odu@@-h2 </xsl:text>
    <xsl:value-of select="@id" />
    <xsl:text>##oduend@@ \v0}</xsl:text>
</xsl:template>

<!-- headline level 3 -->
<xsl:template match="h3">
    <!-- odu tag open -->
    <xsl:text>{\v ##odu@@+h3 </xsl:text>
    <xsl:value-of select="@id" />
    <xsl:text>##oduend@@ \v0}</xsl:text>
    <!-- formatting -->
    <xsl:text>{\fs30 </xsl:text>
    <!-- inner content -->
    <xsl:apply-templates/>
    <!-- newline -->
    <xsl:text>\par}</xsl:text>
    <!-- odu tag close -->
    <xsl:text>\v ##odu@@-h3 </xsl:text>
    <xsl:value-of select="@id" />
    <xsl:text>##oduend@@ \v0}</xsl:text>
</xsl:template>

<!-- newline -->
<xsl:template match="br">
    <!-- newline -->
    <xsl:text>\line</xsl:text>
    <!-- odu tag empty -->
    <xsl:text>{\v ##odu@@/br </xsl:text>
    <xsl:value-of select="@id" />
    <xsl:text>##oduend@@ \v0}</xsl:text>
</xsl:template>

<!-- CORRECTIONS -->

<!-- Correction - root elements -->

<!-- "Change" correction -->
<xsl:template match="chg">
    <!-- odu tag open -->
    <xsl:text>{\v ##odu@@+chg </xsl:text>
    <xsl:value-of select="@id" />
    <xsl:text>##oduend@@}</xsl:text>
    <!-- left whitespace -->
    <xsl:text>{\v0 </xsl:text>
    <xsl:call-template name="wsLeft"/>
    <!-- formatting -->
    <xsl:text>{\uld\cf1 </xsl:text>
    <!-- inner content -->
    <xsl:apply-templates/>
    <xsl:text>}</xsl:text>
    <!-- comment -->
    <xsl:call-template name="title"/>
    <xsl:text>}</xsl:text>
    <!-- odu tag close -->
    <xsl:text>{\v ##odu@@-chg </xsl:text>
    <xsl:value-of select="@id" />
    <xsl:text>##oduend@@}</xsl:text>
    <!-- right whitespace -->
    <xsl:call-template name="wsRight"/>
</xsl:template>

```

```

<!-- "Insert" correction -->
<xsl:template match="ins">
  <!-- odu tag open -->
  <xsl:text>{\v ##odu@@+ins </xsl:text>
  <xsl:value-of select="@id" />
  <xsl:text>##oduend@@}</xsl:text>
  <!-- left whitespace -->
  <xsl:text>{\v0 </xsl:text>
  <xsl:call-template name="wsLeft"/>
  <!-- formatting -->
  <xsl:text>{\uld\b\cf1 </xsl:text>
  <!-- inner content -->
  <xsl:apply-templates/>
  <xsl:text>}</xsl:text>
  <!-- comment -->
  <xsl:call-template name="title"/>
  <xsl:text>}</xsl:text>
  <!-- odu tag close -->
  <xsl:text>{\v ##odu@@-ins </xsl:text>
  <xsl:value-of select="@id" />
  <xsl:text>##oduend@@}</xsl:text>
  <!-- right whitespace -->
  <xsl:call-template name="wsRight"/>
</xsl:template>

<!-- "Delete" correction -->
<xsl:template match="del">
  <!-- odu tag open -->
  <xsl:text>{\v ##odu@@+del </xsl:text>
  <xsl:value-of select="@id" />
  <xsl:text>##oduend@@}</xsl:text>
  <!-- left whitespace -->
  <xsl:text>{\v0 </xsl:text>
  <xsl:call-template name="wsLeft"/>
  <!-- formatting -->
  <xsl:text>{\uld\strike\cf1 </xsl:text>
  <!-- inner content -->
  <xsl:apply-templates/>
  <xsl:text>}</xsl:text>
  <!-- comment -->
  <xsl:call-template name="title"/>
  <xsl:text>}</xsl:text>
  <!-- odu tag close -->
  <xsl:text>{\v ##odu@@-del </xsl:text>
  <xsl:value-of select="@id" />
  <xsl:text>##oduend@@}</xsl:text>
  <!-- right whitespace -->
  <xsl:call-template name="wsRight"/>
</xsl:template>

<!-- "Word order" correction -->
<xsl:template match="wo">
  <!-- odu tag open -->
  <xsl:text>{\v ##odu@@+wo </xsl:text>
  <xsl:value-of select="@id" />
  <xsl:text>##oduend@@}</xsl:text>
  <!-- left whitespace -->
  <xsl:text>{\v0 </xsl:text>
  <xsl:call-template name="wsLeft"/>
  <!-- formatting -->

```

```

<xsl:text>{\uld\i </xsl:text>
<!-- inner content -->
<xsl:apply-templates/>
<xsl:text></xsl:text>
<!-- comment -->
<xsl:call-template name="title"/>
<!-- right whitespace -->
<xsl:call-template name="wsRight"/>
<xsl:text></xsl:text>
<!-- odu tag close -->
<xsl:text>{\v ##odu@@-wo </xsl:text>
<xsl:value-of select="@id" />
<xsl:text>##oduend@@</xsl:text>
</xsl:template>

<!-- Correction - subelements -->

<!-- Correct text -->
<xsl:template match="r">
  <!-- whitespace -->
  <xsl:if test="parent::chg">
    <xsl:text> </xsl:text>
  </xsl:if>
  <!-- formatting -->
  <xsl:text>{\cf2\strike0 </xsl:text>
  <!-- inner content -->
  <xsl:apply-templates/>
  <xsl:text></xsl:text>
</xsl:template>

<!-- Alternative correct text -->
<xsl:template match="other">
  <!-- formatting -->
  <xsl:text>{\sub\cf2\i0\b0\strike0 / </xsl:text>
  <!-- inner content -->
  <xsl:apply-templates/>
  <xsl:text></xsl:text>
</xsl:template>

<!-- Word order subelement -->
<xsl:template match="order">
  <!-- odu tag open begin -->
  <xsl:text>{\v ##odu@@+order </xsl:text>
  <xsl:value-of select="@id" />
  <xsl:text>##</xsl:text>
  <!-- left whitespace -->
  <xsl:text>{\v0 </xsl:text>
  <xsl:if test="not(position()=1)">
    <xsl:text> </xsl:text>
  </xsl:if>
  <xsl:text></xsl:text>
  <!-- order number -->
  <xsl:if test="not(@o='0')">
    <!-- formatting -->
    <xsl:text>{\v0\super\cf0\ulnone [</xsl:text>
    <!-- number -->
    <xsl:value-of select="@o"/>
    <xsl:text>]}</xsl:text>
  </xsl:if>
  <!-- odu tag open end -->
  <xsl:text>{\v ##oduend@@</xsl:text>

```

```

    <!-- inner content -->
    <xsl:apply-templates/>
    <!-- comment -->
    <xsl:call-template name="title"/>
    <!-- odu tag close -->
    <xsl:text>{\v ##odu@@-order </xsl:text>
    <xsl:value-of select="@id" />
    <xsl:text>##oduend@@}</xsl:text>
</xsl:template>

<xsl:template match="c">
    <!-- odu tag open -->
    <xsl:text>{\v ##odu@@+c </xsl:text>
    <xsl:value-of select="@id" />
    <xsl:text>##oduend@@}</xsl:text>
    <!-- left whitespace -->
    <xsl:text>{\v0 </xsl:text>
    <xsl:call-template name="wsLeft"/>
    <!-- formatting -->
    <xsl:text>{\uld </xsl:text>
    <!-- inner content -->
    <xsl:apply-templates/>
    <xsl:text>}</xsl:text>
    <!-- comment -->
    <xsl:call-template name="title"/>
    <!-- right whitespace -->
    <xsl:call-template name="wsRight"/>
    <xsl:text>}</xsl:text>
    <!-- odu tag close -->
    <xsl:text>{\v ##odu@@-c </xsl:text>
    <xsl:value-of select="@id" />
    <xsl:text>##oduend@@}</xsl:text>
</xsl:template>

<!-- NAMED TEMPLATES -->

<!-- correction comment -->
<xsl:template name="title">
    <xsl:if test="@title">
        <!-- formatting -->
        <xsl:text>{\uld\sub\cf0\i0\b0\strike0 [</xsl:text>
        <!-- comment -->
        <xsl:value-of select="@title"/>
        <xsl:text>] }</xsl:text>
    </xsl:if>
</xsl:template>

<!-- left whitespace -->
<xsl:template name="wsLeft">
    <xsl:if test="not (@ws='none' or @ws='right')">
        <!-- whitespace -->
        <xsl:text>\v0 </xsl:text>
    </xsl:if>
</xsl:template>

<!-- right whitespace -->
<xsl:template name="wsRight">
    <xsl:if test="not (@ws='none' or @ws='left')">
        <!-- whitespace -->
        <xsl:text>\v0 </xsl:text>
    </xsl:if>

```

```
</xsl:template>
```

```
</xsl:stylesheet>
```

10.2 OprDU Server

Dokumenty v této kapitole jsou automaticky generované OprDU Serverem.

WSDL pro OprDU server pro učitele

```
<?xml version="1.0" encoding="utf-8"?>
<definitions xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:tns="http://oprdu.nikde.eu/oduserver/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns="http://schemas.xmlsoap.org/wsdl/"
targetNamespace="http://oprdu.nikde.eu/oduserver/">

<types>
<xsd:schema targetNamespace="http://oprdu.nikde.eu/oduserver/">
  <xsd:import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
  <xsd:import namespace="http://schemas.xmlsoap.org/wsdl/" />
</xsd:schema>
</types>

<message name="itemsRequest">
  <part name="login" type="xsd:string" />
  <part name="password" type="xsd:string" />
  <part name="statusFrom" type="xsd:int" />
  <part name="statusTo" type="xsd:int" /></message>
<message name="itemsResponse">
  <part name="result" type="xsd:string" />
  <part name="items" type="xsd:string" /></message>
<message name="setRequest">
  <part name="login" type="xsd:string" />
  <part name="password" type="xsd:string" />
  <part name="uid" type="xsd:string" />
  <part name="newstatus" type="xsd:int" /></message>
<message name="setResponse">
  <part name="result" type="xsd:string" />
  <part name="status" type="xsd:int" /></message>
<message name="headRequest">
  <part name="login" type="xsd:string" />
  <part name="password" type="xsd:string" />
  <part name="uid" type="xsd:string" /></message>
<message name="headResponse">
  <part name="result" type="xsd:string" />
  <part name="head" type="xsd:string" /></message>
<message name="getRequest">
  <part name="login" type="xsd:string" />
  <part name="password" type="xsd:string" />
  <part name="uid" type="xsd:string" /></message>
<message name="getResponse">
  <part name="result" type="xsd:string" />
  <part name="head" type="xsd:string" />
  <part name="body" type="xsd:string" /></message>
<message name="putRequest">
  <part name="login" type="xsd:string" />
  <part name="password" type="xsd:string" />
```

```

    <part name="uid" type="xsd:string" />
    <part name="body" type="xsd:string" />
    <part name="maxid" type="xsd:int" /></message>
<message name="putResponse">
    <part name="result" type="xsd:string" /></message>
<message name="putnewRequest">
    <part name="login" type="xsd:string" />
    <part name="password" type="xsd:string" />
    <part name="head" type="xsd:string" />
    <part name="body" type="xsd:string" /></message>
<message name="putnewResponse">
    <part name="result" type="xsd:string" />
    <part name="uid" type="xsd:string" /></message>
<message name="studentsRequest">
    <part name="login" type="xsd:string" />
    <part name="password" type="xsd:string" /></message>
<message name="studentsResponse">
    <part name="result" type="xsd:string" />
    <part name="students" type="xsd:string" /></message>
<message name="settingsGetRequest">
    <part name="login" type="xsd:string" />
    <part name="password" type="xsd:string" /></message>
<message name="settingsGetResponse">
    <part name="result" type="xsd:string" />
    <part name="settings" type="xsd:string" /></message>
<message name="settingsPutRequest">
    <part name="login" type="xsd:string" />
    <part name="password" type="xsd:string" />
    <part name="settings" type="xsd:string" /></message>
<message name="settingsPutResponse">
    <part name="result" type="xsd:string" /></message>
<message name="pingRequest">
    <part name="login" type="xsd:string" />
    <part name="password" type="xsd:string" /></message>
<message name="pingResponse">
    <part name="server" type="xsd:string" />
    <part name="user" type="xsd:string" /></message>

<portType name="ODUserServerPortType">
    <operation name="items">
        <input message="tns:itemsRequest"/>
        <output message="tns:itemsResponse"/>
    </operation>
    <operation name="set">
        <input message="tns:setRequest"/>
        <output message="tns:setResponse"/>
    </operation>
    <operation name="head">
        <input message="tns:headRequest"/>
        <output message="tns:headResponse"/>
    </operation>
    <operation name="get">
        <input message="tns:getRequest"/>
        <output message="tns:getResponse"/>
    </operation>
    <operation name="put">
        <input message="tns:putRequest"/>
        <output message="tns:putResponse"/>
    </operation>
    <operation name="putnew">
        <input message="tns:putnewRequest"/>

```



```

        <output message="tns:putnewResponse"/>
    </operation>
    <operation name="students">
        <input message="tns:studentsRequest"/>
        <output message="tns:studentsResponse"/>
    </operation>
    <operation name="settingsGet">
        <input message="tns:settingsGetRequest"/>
        <output message="tns:settingsGetResponse"/>
    </operation>
    <operation name="settingsPut">
        <input message="tns:settingsPutRequest"/>
        <output message="tns:settingsPutResponse"/>
    </operation>
    <operation name="ping">
        <input message="tns:pingRequest"/>
        <output message="tns:pingResponse"/>
    </operation>
</portType>

<binding name="ODUserServerBinding" type="tns:ODUserServerPortType">
    <soap:binding style="rpc"
transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="items">
        <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserver/index.php/items"
style="rpc"/>
        <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
        <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
    </operation>
    <operation name="set">
        <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserver/index.php/set"
style="rpc"/>
        <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
        <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
    </operation>
    <operation name="head">
        <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserver/index.php/head"
style="rpc"/>
        <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
        <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
    </operation>
    <operation name="get">
        <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserver/index.php/get"
style="rpc"/>
        <input><soap:body use="encoded"

```

```

namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
  <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
</operation>
  <operation name="put">
    <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserver/index.php/put"
style="rpc"/>
    <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
    <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
  </operation>
  <operation name="putnew">
    <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserver/index.php/putnew"
style="rpc"/>
    <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
    <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
  </operation>
  <operation name="students">
    <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserver/index.php/student
s" style="rpc"/>
    <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
    <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
  </operation>
  <operation name="settingsGet">
    <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserver/index.php/setting
sGet" style="rpc"/>
    <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
    <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
  </operation>
  <operation name="settingsPut">
    <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserver/index.php/setting
sPut" style="rpc"/>
    <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
    <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
  </operation>

```

```

        <operation name="ping">
            <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserver/index.php/ping"
style="rpc"/>
            <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" /></input>
            <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserver/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" /></output>
        </operation>
    </binding>

    <service name="ODUserver">
        <port name="ODUserverPort" binding="tns:ODUserverBinding">
            <soap:address
location="http://localhost/nikde.eu/oprdu/oduserver/index.php"/>
        </port>
    </service>

</definitions>

```

WSDL pro OprDU server pro studenty

```

<definitions xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:tns="http://oprdu.nikde.eu/oduserverStudent/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns="http://schemas.xmlsoap.org/wsdl/"
targetNamespace="http://oprdu.nikde.eu/oduserverStudent/">

    <types>
        <xsd:schema targetNamespace="http://oprdu.nikde.eu/oduserverStudent/"
>
            <xsd:import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
            <xsd:import namespace="http://schemas.xmlsoap.org/wsdl/" />
        </xsd:schema>
    </types>

    <message name="itemsRequest">
        <part name="login" type="xsd:string" />
        <part name="password" type="xsd:string" />
        <part name="statusFrom" type="xsd:int" />
        <part name="statusTo" type="xsd:int" /></message>
    <message name="itemsResponse">
        <part name="result" type="xsd:string" />
        <part name="items" type="xsd:string" /></message>
    <message name="setRequest">
        <part name="login" type="xsd:string" />
        <part name="password" type="xsd:string" />
        <part name="uid" type="xsd:string" />
        <part name="newstatus" type="xsd:int" /></message>
    <message name="setResponse">
        <part name="result" type="xsd:string" />
        <part name="status" type="xsd:int" /></message>
    <message name="headRequest">
        <part name="login" type="xsd:string" />
        <part name="password" type="xsd:string" />

```

```

    <part name="uid" type="xsd:string" /></message>
<message name="headResponse">
    <part name="result" type="xsd:string" />
    <part name="head" type="xsd:string" /></message>
<message name="getRequest">
    <part name="login" type="xsd:string" />
    <part name="password" type="xsd:string" />
    <part name="uid" type="xsd:string" /></message>
<message name="getResponse">
    <part name="result" type="xsd:string" />
    <part name="head" type="xsd:string" />
    <part name="body" type="xsd:string" /></message>
<message name="putRequest">
    <part name="login" type="xsd:string" />
    <part name="password" type="xsd:string" />
    <part name="uid" type="xsd:string" />
    <part name="body" type="xsd:string" />
    <part name="maxid" type="xsd:int" /></message>
<message name="putResponse">
    <part name="result" type="xsd:string" /></message>
<message name="putnewRequest">
    <part name="login" type="xsd:string" />
    <part name="password" type="xsd:string" />
    <part name="title" type="xsd:string" />
    <part name="teacher" type="xsd:string" />
    <part name="comment" type="xsd:string" /></message>
<message name="putnewResponse">
    <part name="result" type="xsd:string" />
    <part name="uid" type="xsd:string" /></message>
<message name="teachersRequest">
    <part name="login" type="xsd:string" />
    <part name="password" type="xsd:string" /></message>
<message name="teachersResponse">
    <part name="result" type="xsd:string" />
    <part name="teachers" type="xsd:string" /></message>
<message name="pingRequest">
    <part name="login" type="xsd:string" />
    <part name="password" type="xsd:string" /></message>
<message name="pingResponse">
    <part name="server" type="xsd:string" />
    <part name="user" type="xsd:string" /></message>

<portType name="ODUserServerStudentPortType">
    <operation name="items">
        <input message="tns:itemsRequest"/>
        <output message="tns:itemsResponse"/>
    </operation>
    <operation name="set">
        <input message="tns:setRequest"/>
        <output message="tns:setResponse"/>
    </operation>
    <operation name="head">
        <input message="tns:headRequest"/>
        <output message="tns:headResponse"/>
    </operation>
    <operation name="get">
        <input message="tns:getRequest"/>
        <output message="tns:getResponse"/>
    </operation>
    <operation name="put">
        <input message="tns:putRequest"/>

```

```

        <output message="tns:putResponse"/>
    </operation>
    <operation name="putnew">
        <input message="tns:putnewRequest"/>
        <output message="tns:putnewResponse"/>
    </operation>
    <operation name="teachers">
        <input message="tns:teachersRequest"/>
        <output message="tns:teachersResponse"/>
    </operation>
    <operation name="ping">
        <input message="tns:pingRequest"/>
        <output message="tns:pingResponse"/>
    </operation>
</portType>

<binding name="ODUserServerStudentBinding"
type="tns:ODUserServerStudentPortType">
    <soap:binding style="rpc"
transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="items">
        <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserverStudent/index.php/
items" style="rpc"/>
        <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
        <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
    </operation>
    <operation name="set">
        <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserverStudent/index.php/
set" style="rpc"/>
        <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
        <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
    </operation>
    <operation name="head">
        <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserverStudent/index.php/
head" style="rpc"/>
        <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
        <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
    </operation>
    <operation name="get">
        <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserverStudent/index.php/
get" style="rpc"/>
        <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
        <output><soap:body use="encoded"

```

```

namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
</operation>
<operation name="put">
  <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserverStudent/index.php/
put" style="rpc"/>
  <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
  <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
</operation>
<operation name="putnew">
  <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserverStudent/index.php/
putnew" style="rpc"/>
  <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
  <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
</operation>
<operation name="teachers">
  <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserverStudent/index.php/
teachers" style="rpc"/>
  <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
  <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
</operation>
<operation name="ping">
  <soap:operation
soapAction="http://localhost/nikde.eu/oprdu/oduserverStudent/index.php/
ping" style="rpc"/>
  <input><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></input>
  <output><soap:body use="encoded"
namespace="http://oprdu.nikde.eu/oduserverStudent/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"></output>
</operation>
</binding>

<service name="ODUserverStudent">
  <port name="ODUserverStudentPort"
binding="tns:ODUserverStudentBinding">
    <soap:address
location="http://localhost/nikde.eu/oprdu/oduserverStudent/index.php"/>
  </port>
</service>

</definitions>

```