Univerzita Karlova v Praze, Matematicko-fyzikální fakulta

Rudolf Rosa, studijní program Informatika – Programování

Vedoucí práce: Mgr. Tomáš Knap

OprDU – software pro opravu domácích úkolů

Ročníkový projekt - Specifikace

Obsah

1. Úvod....3
    1. Srovnání s existujícími nástroji....3
        Papír a tužka....3
        Počítačové programy....3
        OprDU....4
    2. Součásti projektu....4
2. ODU 0.1 – XML formát pro texty....6
    1. Head (hlavička)....6
        Příklad hlavičky dokumentu....7
        Stav dokumentu....7
    2. Tělo dokumentu....7
        Formátovací tagy....7
        Korektorské tagy....8
3. ODUserver (front-end webová aplikace)....13
    1. Komunikace se studentem....13
    2. Správa a uchovávání textů....13
        Stav dokumentu....13
    3. Komunikace s ODUklientem....15
4. ODUklient (back-end aplikace pro provádění korektur textů)....16
    1. Komunikace s ODUserverem....16
    2. Oprava ODU dokumentů....16
    3. Další funkce a vlastnosti....17
5. Komunikace ODUserveru a ODUklienta....18
    1. List – Výpis souborů....18
        Request 1....18
        Request 2....18
        Response....18
    2. Set – nastavení stavu dokumentu....19
        Request....19
        Response 1 – v pořádku....19
        Response 2 – chyba....20
    3. Head – stažení hlavičky jednoho souboru....20
        Request....20
        Response 1 – v pořádku....20
        Response 2 – dokument nelze poskytnout....20
    4. Get – stažení jednoho souboru....21
        Request....21
        Response 1 – v pořádku....21
        Response 2 – dokument nelze poskytnout....21
    5. Put – uložení souboru na server....21
        Request 1....21
        Request 2....22
        Response 1 – v pořádku....22
        Response 2 – chyba....22
        Ping – Ověření funkce serveru a přihlašovacích údajů....22
        Request....23
        Response....23

1. Úvod

Projekt OprDU je komplexním řešením problematiky opravy domácích úkolů. Postihuje celý proces práce na domácím úkolu, od jeho vytvoření přes jeho odevzdání a opravení až po finální prezentaci textu.

1. Srovnání s existujícími nástroji

Nejde pochopitelně o nový problém a je v současné době řešen různými způsoby. Nenašel jsem ale způsob, který by problém řešil dostatečně komplexně a splňoval všechny požadavky dnešní doby. Podívejme se nyní na srovnání s nejčastěji používanými nástroji.

Papír a tužka

I v dnešní době většina učitelů při opravě úkolů sahá po tužce a opravy vpisuje ručně do textu. Mnoho studentů své práce píše na počítači, někteří z nich je pak učitelům zasílají e-mailem, a přesto jsou poté vytištěny a opraveny ručně na papíře. To je samozřejmě rychlé a jednoduché, ale má to řadu nevýhod:

- Ručně opravený text je nutné předat studentovi až na následující lekci. V mnohých případech se tato koná až za týden, často se navíc stane, že se jí student nebo učitel nemůže zúčastnit. Prodlužující se prodleva má přitom negativní vliv na benefit pro studenta – čím kratší je interval mezi napsáním textu a jeho opravou, tím lépe má student své dílo v paměti a tím více se poučí ze svých chyb.
- Text je nutné předat fyzicky, na papíře. To je problematické u dálkových studentů, navíc se občas stává, že se text ztratí nebo poškodí.
- Text existuje jen v jednom exempláři. Pokud jde o společnou práci více studentů, je pak nutné text okopírovat, čímž se může ztratit čitelnost vyznačených oprav.
- S textem není možné dále elektronicky pracovat. Mnohdy přitom může být žádoucí vystavení dokumentu na internetových stránkách; někdy zase nejde o finální stadium dokumentu, ale pouze o korekturu textu, který je součástí studentského projektu apod. V takovém případě je nutné text přepsat zpátky do počítače, nebo podle papírové opravy ručně opravit dokument v počítači.
- Rukopis učitele není vždy dobře čitelný, o rukopisu studentů nemluvě.

Počítačové programy

Samozřejmě existují i učitelé, kteří opravy textů provádějí na počítači. V současnosti běžně dostupné softwarové vybavení, které obvykle používají – typicky nějaký kancelářský softwarový balík typu Microsoft Office či OpenOffice.org – jim ale vycházejí vstříc pouze částečně. Některé programy podporují verzování (pod názvem sledování změn, správa verzí apod.), kde v textu vyznačují, jaké úpravy byly provedeny. Jsou zde ale stále některé nedostatky[1]:

- V souboru jsou vyznačená místa, kde byl přidán či odstraněn text. Není ale možné jednoduše vyznačit výměnu jednoho slova za jiné. Výměna slova je pak prezentována jako odstranění slova a vložení jiného.
- Změnu pořadí slov a 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, což nepostihuje provedenou opravu.

1 Vycházím z programu Microsoft Word 2002.

- Více oprav na jednom řádku nelze dobře rozlišit[2]. Při větším množství oprav se pak dokument rychle stává nepřehledným.
- Opravy nelze zanořovat do sebe, přičemž například u změny pořadí slov jde o logický požadavek – slovo může být na špatném místě a ještě obsahovat např. pravopisnou chybu.
- U opravy je možné vložit textový komentář, žádné další metainformace ale podporované nejsou.
- Vzhled opravy není možné příliš upravit, je předdefinovaný výrobcem.
- 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.
- Není jednoduše možné přecházet k verzi před opravou a po ní.
- Export je poněkud rozpačitý (je možné, že v novějších verzích se zlepší).
- Software je často placený.
- Výhodou i nevýhodou může být používaný software. Kancelářské programy bývají většinou dobře rozšířené, v pokročilejších funkcích ale může být problémem nekompatibilita mezi různými kancelářskými balíky i mezi různými verzemi téhož software.

OprDU

Všechny výše uvedené nevýhody odstraňuje software OprDU. Dokument vzniká v elektronické podobě na serveru OprDU, poté je elektronicky odevzdán učiteli. Učitel si jej stáhne do svého počítače, do programu OprDU klient, ve kterém provede korektury, a uloží jej zpět na server, kde si student opravený text vyzvedne. Text je možné exportovat do různých formátů, přičemž výstup může, ale nemusí, obsahovat původní i opravený text.

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 vytisknou a předat studentovi ve fyzické podobě, pokud je to v dané situaci vhodnější.

Celé řešení je vyvíjeno jako veřejně dostupné, přinejmenším ODUserver zároveň jako open-source. Vše bude zdarma ke stažení na webu produktu http://oprdu.nikde.eu, kde bude po registraci také k dispozici instalace OprDU serveru.

Opravy jsou vyznačovány v XML formátu ODU, který je informačně bohatší než všechna srovnávaná řešení. Jako každé XML má zároveň tu výhodu, že jde o čistě textový formát. Půjde o veřejný formát, každá část řešení proto může být implementována i jinými programátory.

Poznámka: Software je primárně určen pro opravu domácích úkolů, proto v textu pro autora dokumentu používám označení student a pro korektora učitel. To samozřejmě nebrání použití software i k jiným účelům.

2. Součásti projektu

Software postihuje celý proces práce na dokumentu, přičemž různé úkoly mají na starosti různé součásti projektu. Další kapitoly se pak již věnují jednotlivým součástem projektu.

2 Nelze rozeznat, která oprava a který komentář se týká kterého místa v textu, pokud jsou tato místa na stejném řádku.

- Dokument je po celou dobu ve formátu ODU (viz kapitola 2). ODU je XML formát navržený pro účely tohoto projektu, který umožňuje vyznačit opravy a formátování dokumentu.
- Vytvoření dokumentu probíhá na ODUserveru (ODUserver viz kap. 3), který zároveň spravuje vytvořené dokumenty a umožňuje předání opraveného dokumentu studentovi. ODUserver je serverová aplikace, která podporuje registraci a přihlašování různých uživatelů, a která je studentům dostupná přes webové rozhraní.
- Oprava dokumentu probíhá v ODUklientovi (ODUklient viz kap. 4), což je Windows aplikace, kterou si instaluje učitel na svůj počítač.
- Komunikace ODUserveru a ODUklienta (viz kap. 5) probíhá pomocí Web Services (ODUserver je zde pochopitelně v roli serveru).

2. ODU 0.1 – XML formát pro texty

Všechny části software pracují s dokumenty ve formátu ODU. V této specifikaci popisuji verzi 0.1; součástí pilotního projektu bude verze 1.0.

Jde o XML formát, který bude definován pomocí XSD. Zde ukážu základní strukturu dokumentu na příkladech, neboť je to srozumitelnější a informačně bohatší než XSD.

V názvech elementů i atributů jsem se často inspiroval v jazyce HTML, u nových názvů vycházím z anglického jazyka. To má několik výhod:

- Zdrojový kód dokumentu je srozumitelný i pro člověka, který není seznámen s ODU formátem, ale orientuje se alespoň základním způsobem v HTML a angličtině. Pro téhož člověka není problémem ruční vytvoření ODU dokumentu.
- Převod do HTML (pomocí XSLT) je jednodušší a rychlejší.
- V nouzi je možné ODU dokument přímo interpretovat jako HTML se zachováním základní informační hodnoty obsahu.

Veškeré názvy jsou jednoslovné a sestávají pouze z malých písmen anglické abecedy.

Každý dokument má kořenový element `odu`, který obsahuje právě dva elementy - `head` (hlavičku) a `body` (tělo). Tyto elementy obsahují další elementy - hlavička metainformace (název, autor, apod.) a tělo vlastní text dokumentu.

```
<?xml version="1.0" encoding="utf-8"?>
<odu>
   <head>
     ...elementy s metainformacemi...
   </head>
   <body>
     ...vlastní text dokumentu...
   </body>
</odu>
```

1. Head (hlavička)

Hlavička obsahuje veřejné informace o souboru, které se zobrazují ve výpisech apod.

Hlavička ODU dokumentu povinně obsahuje právě tyto metainformace:

- `title` (název dokumentu)
- `student` (jméno autora dokumentu)
- `created` (datum vytvoření dokumentu)
- `comment` (nepovinný komentář studenta k dokumentu – obsah elementu může být prázdný)
- `teacher` (učitel, kterého student pověřil opravou dokumentu)
- `status` (stav dokumentu, viz níže)

Příklad hlavičky dokumentu

```
<head>
   <title>O žížalách</title>
   <student>R.U.R.</student>
   <created>2009-04-16</created>
   <comment>Základní informace o žížalách</comment>
   <teacher>Rosa</teacher>
   <status>200</status>
</head>
```

Stav dokumentu

Každý dokument prochází postupně různými stavy, které jsou definovány nezápornými celými čísly (tvořícími rostoucí posloupnost). Všechna čísla jsou třímístná, ve verzi 0.1 je významná pouze první číslice (zbývající dvě jsou rezervovány pro potenciální využití v dalších verzích, například pro zjemnění určení stavu).

Stav určuje přístupová oprávnění studenta a učitele k dokumentu. Tato práva spravuje především ODUserver, na tomto místě proto pouze uvedu seznam možných stavů. Podrobněji o stavech viz kapitola 3 ODUserver.

- 100 Dokument je rozpracován
- 200 Dokument je odevzdán
- 300 Dokument je opravován
- 400 Dokument je opraven
- 500 Dokument je vyzvednut

2. Tělo dokumentu

Tělo dokumentu obsahuje vlastní text dokumentu, obalený formátovacími a korektorskými tagy.

Formátovací tagy

Formátovací tagy do textu vkládá student při psaní. Ve verzi 0.1 jsou formátovací možnosti omezené, počítám s jejich rozšířením v dalších verzích.

Podporované jsou tyto tagy (přejaté z HTML):

- `h1`, `h2`, `h3` - vymezuje nadpis první, druhé a třetí úrovně (párový)
- `p` - vymezuje odstavec (párový)
- `br` - označuje řádkový zlom (nepárový)

Tělo dokumentu (rozuměj ve stavu < 300) smí obsahovat pouze tyto elementy, veškerý text musí být uzavřen v nadpisech a odstavcích. Nadpisy a odstavce do sebe není možné zanořovat, řádkové zlomy se mohou vyskytovat v odstavcích i v nadpisech. Nadpis první úrovně smí být použit nejvýše jednou, a to jako první element těla.

Příklad těla ODU dokumentu ve stavu 200

Úryvek z české Wikipedie.

```
<body>
<h1>
   Žížala obecná
</h1>
<p>
   Žížala obecná (Lumbricus terrestris) je eurasijský kroužkovec.
Má složené svaly ze dvou vrstev - příčných a podélných. Napnutím
příčného svalstva se pohne dopředu přední část jejího těla.
Vyvolané stahy svaloviny projdou celým tělem a umožní tak pohyb
zadní části. Pak podélné svaly posunou ocasní částí. To už se ale
napnulo příčné svalstvo a vše se znovu opakuje.
</p>
<h2>
   Hlen
</h2>
<p>
   Hlen, který vylučují její žlázy, jí usnadňuje dýchání a pohyb
po nerovném povrchu a zabraňuje vysychání kůže. (Z tohoto důvodu
se jí ale řada lidí štítí.)
   <br/>
   Na přídi těla má hmatový prstík, který jí slouží k snazší
orientaci. Při jemném promnutí tenké pokožky jsou zřetelně cítit
redukované štětinky.
</p>
<p>
   Žížala dorůstá délky přibližně 9 až 30 cm. Její příbuzní však
mohou dosahovat mnohem větších rozměrů.
</p>
<br/>
<br/>
<p>
   Zdroj: cs.wikipedia.org
</p>
</body>
```

Korektorské tagy

Korektorské tagy do textu vkládá učitel. Tag obaluje úsek původního textu, kterého se korektura týká, a může obsahovat další upřesňující atributy a tagy. Pro zjednodušení vyjadřování budu pro úsek textu, kterého se oprava týká, jakožto i pro text, kterým má být chybný text nahrazen, používat termín slovo – může ale pochopitelně jít i o jednotlivá písmena nebo naopak o více slov.

Učitel nesmí upravovat hlavičku dokumentu ani obsah, který vytvořil student – smí pouze přidávat korektorské značky. ODUklient neumožní učiteli provádět nepovolené úpravy. ODUserver učiteli vůbec neumožní měnit hlavičku souboru, měl by kontrolovat i to, zda učitel nezměnil obsah dokumentu.

Korektorské tagy tvoří dvě skupiny – chyby a komentáře.

Chyby

Pro vyznačení chyby se používají čtyři elementy. Pro podrobnější specifikaci korektury je možné (a doporučené) u elementu použít atribut `class` s vyznačením třídy chyby, a/nebo vnořené elementy.

Typ elementu vyjadřuje typ chyby – vyjadřuje, jak je chyba opravena. Možnosti jsou:

- `<chg>` (change) – chybné slovo má být zaměněno za jiné slovo; jde zároveň o obecný typ chyby – blíže nespecifikovaná anebo kombinovaná chyba spadá také do typu `ch`
- `<ins>` (insert) – má být vloženo nové slovo; tento typ chyby tedy na rozdíl od ostatních neobaluje žádné původní slovo, obsahuje pouze elementy, přičemž by měla obsahovat element `r`
- `<del>` (delete) – označené slovo má být z textu vypuštěno; tato chyba nesmí obsahovat element `r`
- `<wo>` (word order) – slova jsou ve špatném pořadí; chyba tohoto typu obsahuje elementy `order`, které určují správné pořadí jednotlivých slov, zároveň může obsahovat další vnořené chyby ostatních typů

Třída chyby (atribut `class`) je nepovinná, ve smysluplných případech je doporučená pro zvýšení informační hodnoty korektury. Zde do další verze formátu předpokládám další úpravy a rozšíření (po rozsáhlejším vyzkoušení v praxi).

- `spell` (spelling) – chyba v pravopisu slova, překlep
- `voc` (vocabulary) – nevhodný výraz
- `grammar` – gramatická chyba
- `style` – stylistický nedostatek
- `phrase` – chyba ve vazbě (ustálená slovní spojení apod.)

U jedné chyby lze vyznačit i více tříd, oddělených mezerami.

Vnořené elementy:

- `<r>` (right) – obsahuje slovo, kterým se má nahradit chybné slovo (a to doslova, tedy tak, aby po nahrazení byl text správně)
- `<other>` – tento element se může vyskytovat v libovolném počtu, jeho obsahem je alternativní oprava (tedy jiný možný obsah pro element `r`)
- `<order>` – nepárový tag, smí se vyskytovat pouze v elementu `wo`; povinný kladný celočíselný atribut `o` obsahuje pořadí slova v úseku obsaženém v elementu `wo`, přičemž slovo začíná tagem `order` a končí dalším tagem `order` nebo koncem elementu `wo`

Komentáře

Komentář se vyznačuje tagem `<c>`, text komentáře je obsahem povinného atributu `title`. Komentář může obalovat slovo, kterého se týká, nebo může být prázdný. Komentář k opravě se vyznačuje elementem `c` vnořeným do korektorského elementu, neobsahujícím žádný obsah.

Bílé znaky

Formát se podobně jako HTML řídí pravidlem, že libovolný počet bílých znaků znamená jednu mezeru; řádkové zlomy lze explicitně určit s pomocí formátovacích tagů.

Korektorský element může obsahovat atribut `ws` (whitespace), který upravuje mezery oddělující obsah tagu a jeho okolí. Povolené hodnoty jsou:

- `all` (výchozí) – mezera z obou stran
- `left` – mezera nalevo, vpravo navazuje těsně – využití např. při opravě slova na konci věty
- `right` (obdobně)
- `none` – těsné přiléhání z obou stran – např. při opravě týkající se jen části slova

Příklad těla ODU dokumentu ve stavu 400

Mnohé chyby nejsou autentické, jsou do textu zanesené úmyslně pro demonstraci různého použití korektorských tagů.

```
<body>

<h1>
   Žížala obecná
</h1>

<p>
   <wo class="phrase">
     <order o="2" />
     <chg ws="none"> O <r>o</r> </chg>
     becná
     <order o="1" />
     <chg ws="none"> ž <r>Ž</r> </chg>
     ížala
   </wo>

   (Lumbricus terrestris) je eurasijsk

   <chg class="spel" ws="right">
     í
     <r>ý</r>
   </chg>

 kroužkovec. Má

   <wo>
     <order o="2" />složené <order o="1" />svaly
   </wo>

   ze dvou

   <chg class="voc">
     vrstev
     <r>typů</r>
```

```
    <other>druhů</other>
    <other>skupin</other>
  </chg>

  - příčných a podélných. Napnutím příčného svalstva se pohne
dopředu přední část jejího těla. Vyvolané stahy svaloviny projdou
celým tělem a umožní tak pohyb zadní části. Pak podélné svaly
posunou ocasní částí.

  <chg class="style" ws="right">
    To
    <r>V tomto okamžiku</r>
    <other>V této chvíli</other>
    <c title="nehodí se do odborného textu" />
  </chg>

  už se ale napnulo příčné svalstvo a vše se znovu opakuje.
</p>

<h2>
  Hlen
</h2>

<p>
  Hlen, který v

  <chg class="gram" ws="none">
    i
    <r>y</r>
    <c title="vyjmenovaná předpona 'vy-'" />
  </chg>

lučují její žlázy,

  <del class="style">
    jí
  </del>

usnadňuje dýchání a pohyb po nerovném povrchu a zabraňuje
vysychání kůže. (Z tohoto důvodu se jí ale řada lidí štítí.)
  <br/>
  Na přídi těla má hmatový prstík, který jí slouží k snazší
orientaci. Při jemném promnutí

  <ins class="style">
    <r>její</r>
  </ins>

  tenké pokožky jsou zřetelně cítit redukované štětinky.
</p>
```

```
<p>
   Žížala dorůstá délky přibližně 9 až 30 cm. Její příbuzní však
mohou dosahovat mnohem
   <chg class="gram phr" ws="left">
     větší rozměry
     <r>větších rozměrů</r>
     <c title="dosahovat koho čeho (2.pád)" />
   </chg>
.
</p>
<ins class="style">
   <c title="text není ukončen" />
</ins>
<br/>
<br/>
<p>
   <c title="wikipedia není důvěryhodným zdrojem">
     Zdroj: cs.wikipedia.org
   </c>
</p>
</body>
```

3. ODUserver (front-end webová aplikace)

ODUserver bude webová aplikace, která bude zajišťovat tři níže uvedené funkce. Obecně je implementačně nezávislá, já ji budu realizovat nad technologiemi PHP, MySQL, HTML, CSS, JS a souvisejícími.

1. Komunikace se studentem

Aplikace bude jednoduchým způsobem podporovat registraci a přihlášení studenta, vytvoření textu a jeho odevzdání, a následné vyzvednutí a zobrazení opraveného textu. Primárně bude tyto funkce poskytovat přes HTTPS (popř. HTTP) v HTML formě, některé výstupy bude volitelně zasílat také pomocí e-mailu.

Aplikace bude zajišťovat všechny základní funkce tak, aby byly dostupné v běžných grafických webových prohlížečích.

2. Správa a uchovávání textů

Aplikace bude uchovávat soubory, které byly na server uloženy, a na základě přístupových oprávnění je bude poskytovat uživatelům. Samotné ODU soubory bude ukládat na disk jako soubory, metainformace o nich a další potřebná data do databáze.

O každém souboru bude server spravovat tyto informace (odpovídají metainformacím v hlavičce ODU dokumentu):

- `student` (autor dokumentu)
- `title` (název dokumentu)
- `created` (datum vytvoření dokumentu)
- `teacher` (učitel, kterého student pověřil opravou dokumentu)
- `comment` (nepovinný komentář studenta k dokumentu)
- `status` (stav dokumentu, viz níže)

Každý dokument vystupuje pod svým `uid`. To je unikátní identifikátor, který vzniká při prvním uložení dokumentu na server (přiděluje jej server). Jde o řetězec složený čísel a malých písmen anglické abecedy, jehož minimální délka je 4 znaky[3] a maximální 40 znaků[4].

Stav dokumentu

Každý dokument prochází postupně různými stavy, které jsou definovány nezápornými celými čísly tvořícími rostoucí posloupnost. Všechna čísla jsou třímístná, v první verzi je významná pouze první číslice (zbývající dvě jsou rezervovány pro potenciální využití v dalších verzích, například pro zjemnění určení stavu).

Stav určuje přístupová oprávnění studenta (autora dokumentu) a učitele (korektora dokumentu), které mohou být:

- L (list): uživateli se dokument zobrazuje v listingu (zobrazuje se autor, datum a stav, nezobrazuje se obsah)

3 `uid` délky 1 až 3 jsou rezervována pro případné jiné využití

4 To umožňuje spravovat až téměř 37^{40} souborů a zároveň nezpůsobí problémy ve většině používaných souborových systémů. Přesto se pro větší přehlednost doporučuje volit kratší řetězce, je-li to možné.

- G (get): uživatel si smí stáhnout dokument do svého počítače
- R (read): uživatel si smí prohlížet dokument
- E (edit): uživatel smí upravovat dokument (tedy upravovat textový obsah dokumentu)
- C (correct): uživatel smí opravovat dokument (tedy upravovat v dokumentu značky pro korekturu)
- D (delete): uživatel smí smazat dokument
- S (set): uživatel smí změnit stav dokumentu

100 Dokument je rozpracován

Výchozí stav dokumentu. Dokument v tomto stavu vzniká, když ho student píše a průběžně jej ukládá.

Oprávnění

Student: LGREDS

Učitel: -

200 Dokument je odevzdán

Do tohoto stavu dokument přejde jeho odevzdáním. Student může dokument odevzdat najednou několika učitelům, jeden z nichž jej opraví.

Oprávnění

Student: LGR

Učitel: LGRS

300 Dokument je opravován

Tento stav znamená, že jeden z určených učitelů začal dokument opravovat. Opravovaný dokument může být průběžně ukládán na server.

Student: L

Učitel: LGRCS

400 Dokument je opraven

Když učitel dokončí opravu dokumentu a finální verzi uloží na server, přejde dokument do stavu 400. Od této chvíle si student může dokument vyzvednout. Vyzvednutí může být provedeno automatickým odesláním dokumentu na email studenta - v takovém případě dokument rovnou přechází do stavu 500.

Student: LGS

Učitel: LGR

500 Dokument je vyzvednut

Změna stavu ze 400 na 500 odpovídá "potvrzení o přečtení opravy" - vyjadřuje, že si student vyzvedl opravený dokument.

Student: LGR

Učitel: LGR

3. Komunikace s ODUklientem

Aplikace bude ODUklientovi umožňovat stahovat ze serveru soubory, které jsou určeny pro daného uživatele, a po opravení je na server opět uložit. Pro bližší popis této funkce viz následující kapitolu.

4. ODUklient (back-end aplikace pro provádění korektur textů)

ODUklient je aplikace pro učitele. Bude umožňovat především opravu ODU dokumentů (tedy zanášení korektorských značek do textu) a komunikaci s ODUserverem. Bude napsán v jazyce C#, pro svůj běh bude využívat prostředí .NET Framework 3.5.

Základním rysem ODUklienta je maximální možná uživatelská přívětivost a snadnost ovládání, tak aby umožňoval učitelům efektivní práci.

1. Komunikace s ODUserverem

Komunikace bude probíhat pomocí webových služeb, jak je podrobně popsáno v kapitole 5 Komunikace ODUserveru a ODUklienta. Program bude používat všechny operace, které server podporuje.

Po spuštění program pomocí funkce ping ověří přihlašovací údaje a funkci ODUserveru; v případě potíží informuje učitele a nabídne mu řešení problému (zadat znovu přihlašovací údaje, zaregistrovat se na serveru, připojit se k jinému serveru apod.).

Na vyžádání program stáhne ze serveru seznam dostupných dokumentů (list); standardně stáhne seznam dokumentů čekajících na opravu a seznam rozopravovaných dokumentů (tj. dokumentů, které učitel již začal opravovat, ale opravu ještě nedokončil), případně je možné si zvolit jinou podmnožinu dokumentů.

Ze seznamu dokumentů si učitel jeden vybere a program jej stáhne ze serveru do počítače, na kterém pak uživatel může provádět korekturu dokumentu - ten je průběžně (automaticky nebo na vyžádání) ukládán zpět na server. Kopie dokumentu se může ukládat na lokální disk, ale zejména jako záloha - směrodatná je vždy ta verze, která je uložená na serveru. Díky tomu je bez problémů možné, aby učitel opravu započal na jednom počítači a dokončil na jiném, není k tomu zapotřebí soubor nijak přenášet. Je proto pochopitelně silně nedoporučováno, aby stejný učitel měl spuštěný program v jednu chvíli více než jednou, neboť by mohlo dojít k přepisování jednoho dokumentu různými jeho verzemi a tedy ke ztrátě provedených korektur.

Po dokončení opravy program uloží dokument na server a zvýší jeho stav na 400 (dokument je opraven).

Pomocí ODUklienta může učitel sledovat stav opravených dokumentů - tedy může zjistit, zda si student opravený dokument již vyzvedl.

2. Oprava ODU dokumentů

Po načtení ODU dokumentu se otevře editační okno. V něm učitel vyznačí jednotlivé chyby. Učitel nemá možnost zasahovat do textu dokumentu, smí pouze přidávat korektorské značky (a případně tyto dále měnit nebo odebírat).

Po označení úseku textu (několika písmen nebo slov) učitel klikne na tlačítko pro vyznačení daného druhu chyby chyby (`chg`, `ins`, `del`, `wo`), volitelně pak zadá její podrobnější specifikaci – jaká je správná varianta, o jakou třídu chyby se jedná, komentář k chybě, apod.

Opravy v dokumentu se zaznamenávají do XML, to je průběžně ukládáno na lokální disk a/nebo na server. Opravy se zároveň zobrazují v editačním okně, kde je možné je dále upravovat.

Pro práci s XML i pro komunikaci se serverem jsou použity příslušné funkce .NETu, rozhraní je řešeno pomocí WinForms (je možné použití WPF).

3. Další funkce a vlastnosti

Program bude umožňovat vytvoření nového dokumentu z plaintextu (např. pro domácí úkoly došlé e-mailem), případně načtení ODU dokumentu z disku. Bude také umožňovat komunikaci pomocí e-mailu – např. zaslání opraveného dokumentu e-mailem studentovi.

Program bude podporovat export opraveného dokumentu do různých výstupních formátů a jeho tisk.

Program bude postaven jako multijazyčný, v této verzi s možností výběru mezi češtinou a angličtinou

5. Komunikace ODUserveru a ODUklienta

ODU dokumenty budou vznikat na straně ODUserveru, ale pracovat se s nimi bude v ODUklientovi, proto je nutná vzájemná komunikace těchto dvou částí.

Komunikace bude probíhat pomocí SOAP, definovaného pomocí WSDL. Zde uvádím pouze komentovaný příklad komunikace, ze kterého je vidět, jak bude probíhat; obalovací tagy pro názornost vynechávám.

Komunikace bude probíhat bezstavově pomocí protokolu HTTPS. Použití HTTP nebude přímo zakázané, nicméně vzhledem k tomu, že součástí každého dotazu budou přihlašovací údaje, bude důrazně nedoporučené. Mělo by se využít pouze v případě nouze, například když server na kterém běží ODUserver nepodporuje HTTPS nebo když zabezpečené komunikaci brání firewall či proxy server.

1. List – Výpis souborů

Nejprve klient získá ze serveru výpis souborů, které jsou pro něj určeny.

Request 1

Bez výběru stavu - použije se implicitní hodnota 200.

```
<list xmlns="http://oprdu.nikde.eu/odu01ws">
   <login>rur</login>
   <password>rrr</password>
</list>
```

Request 2

Vypsat opravené dokumenty, které si student dosud nevyzvedl.

```
<list xmlns="http://oprdu.nikde.eu/odu01ws">
   <login>rur</login>
   <password>rrr</password>
   <status>400</status>
</list>
```

Response

V odpovědi server zašle seznam zvolených dokumentů. Pro každý dokument uvede jeho `uid` a hlavičku. Pokud dotazu neodpovídají žádné dokumentu, obsahuje `number` nulu a `items` má prázdný obsah.

```
<listResponse xmlns="http://oprdu.nikde.eu/odu01ws">
   <number>2</number>
   <items>
     <item>
          <uid>cx6565chnm65xcgfn65<uid>
          <head>
```

```
                <title>O žížalách</title>
                <student>Rudolf Rosa</student>
                <created>2009-04-16</created>
                <comment>Základní informace o žížalách</comment>
                <teacher>rur</teacher>
                <status>200</status>
            </head>
        </item>
        <item>
            <uid>kj762kj435nk34kn456kjn<uid>
            <head>
                <title>Our Trip to Mexico</title>
                <student>José Fernando Rodriguez</student>
                <created>2009-04-17</created>
                <comment></comment>
                <teacher>rur</teacher>
                <status>200</status>
            </head>
        </item>
    </items>
</listResponse>
```

2. Set – nastavení stavu dokumentu

Klient může změnit stav dokumentu, pokud k tomu má oprávnění. Stav je možné pouze zvyšovat.

Request

```
<set xmlns="http://oprdu.nikde.eu/odu01ws">
    <login>rur</login>
    <password>rrr</password>
    <uid>cx6565chnm65xcgfn65</uid>
    <status>300</status>
</set>
```

Response 1 – v pořádku

```
<setResponse xmlns="http://oprdu.nikde.eu/odu01ws">
    <uid>cx6565chnm65xcgfn65</uid>
    <result>ok</result>
    <status>300</status>
</setResponse>
```

Response 2 – chyba

Pokud položka `result` obsahuje jiný text než „`ok`“, akce se nezdařila (obsahem je chybové hlášení). Položka `status` vrací aktuální stav dokumentu, ať už došlo ke změně či nikoli.

```
<setResponse xmlns="http://oprdu.nikde.eu/odu01ws">
   <uid>cx6565chnm65xcgfn65</uid>
   <result>fail - cannot decrement status</result>
   <status>300</status>
</setResponse>
```

3. Head – stažení hlavičky jednoho souboru

Klient požádá o hlavičku souboru pomocí jeho `uid`. Má-li potřebná oprávnění, dostane v odpovědi hlavičku v ODU formátu.

Request

```
<head xmlns="http://oprdu.nikde.eu/odu01ws">
   <login>rur</login>
   <password>rrr</password>
   <uid>cx6565chnm65xcgfn65</uid>
</head>
```

Response 1 – v pořádku

```
<headResponse xmlns="http://oprdu.nikde.eu/odu01ws">
   <uid>cx6565chnm65xcgfn65</uid>
   <head>
     <title>O žížalách</title>
     <student>Rudolf Rosa</student>
     <created>2009-04-16</created>
     <comment>Základní informace o žížalách</comment>
     <teacher>rur</teacher>
     <status>200</status>
   </head>
</headResponse>
```

Response 2 – dokument nelze poskytnout

Dokument buď neexistuje, nebo k němu učitel nemá přístup.

```
<headResponse xmlns="http://oprdu.nikde.eu/odu01ws">
   <uid></uid>
   <odu></odu>
</headResponse>
```

4. Get – stažení jednoho souboru

Ze seznamu si klient vybere soubor, který chce stáhnout, a požádá o něj pomocí identifikátoru souboru. V odpovědi dostane přímo vložený ODU kód.

Request

```
<get xmlns="http://oprdu.nikde.eu/odu01ws">
   <login>rur</login>
   <password>rrr</password>
   <uid>cx6565chnm65xcgfn65</uid>
</get>
```

Response 1 – v pořádku

```
<getResponse xmlns="http://oprdu.nikde.eu/odu01ws">
   <uid>cx6565chnm65xcgfn65</uid>
   <odu>
     <head>
          ...elementy s metainformacemi...
     </head>
     <body>
          ...vlastní text dokumentu...
     </body>
   </odu>
</getResponse>
```

Response 2 – dokument nelze poskytnout

Dokument buď neexistuje, nebo k němu učitel nemá přístup.

```
<getResponse xmlns="http://oprdu.nikde.eu/odu01ws">
   <uid></uid>
   <odu></odu>
</getResponse>
```

5. Put – uložení souboru na server

Klient po provedení oprav uloží dokument na server (může jej ukládat na server i průběžně během oprav). Pokud tento text předtím získal ze serveru, uloží ho pomocí identifikátoru souboru. Pokud ho získal jiným způsobem, musí navíc specifikovat některé metainformace o souboru.

Request 1

Klient získal dokument z tohoto serveru, ukládá ho tedy pod jeho `uid` a posílá pouze tělo dokumentu (hlavička se zachovává původní, neboť tu klient nesmí v žádném případě měnit).

```
<put xmlns="http://oprdu.nikde.eu/odu01ws">
   <uid>cx6565chnm65xcgfn65</uid>
   <odu>
```

```
      <body>
            ...vlastní text dokumentu...
      </body>
   </odu>
</put>
```

Request 2

Klient nahrává na server dokument, který na něm ještě není. Ukládá ho proto včetně hlavičky a `uid` ponechává prázdné.

```
<put xmlns="http://oprdu.nikde.eu/odu01ws">
   <uid></uid>
   <odu>
      <head>
            ...elementy s metainformacemi...
      </head>
      <body>
            ...vlastní text dokumentu...
      </body>
   </odu>
</put>
```

Response 1 – v pořádku

Server vrací v odpovědi `uid` uloženého dokumentu, což význam především při ukládání nového dokumentu – pak jde o nově přidělené `uid`.

```
<putResponse xmlns="http://oprdu.nikde.eu/odu01ws">
   <result>ok</result>
   <uid>cx6565chnm65xcgfn65</uid>
</putResponse>
```

Response 2 – chyba

Pokud `result` obsahuje libovolný jiný text než „`ok`“, jde o chybu (položka obsahuje chybové hlášení) a `uid` je prázdné.

```
<putResponse xmlns="http://oprdu.nikde.eu/odu01ws">
   <result>fail - unknown student "Žabka"</result>
   <uid></uid>
</putResponse>
```

Ping – Ověření funkce serveru a přihlašovacích údajů

Pro otestování funkce serveru může klient poslat ping požadavek. Server odpoví, zda je v pořádku, a zároveň ověří přihlašovací údaje.

Request

```
<ping xmlns="http://oprdu.nikde.eu/odu01ws">
   <login>rur</login>
   <password>rrr</password>
</ping>
```

Response

`<server>`: `ok` = v pořádku; *cokoliv jiného* = nefunguje (obsahem může být chybové hlášení)

`<login>`: `ok` = v pořádku; `bad` = neplatné přihlašovací údaje; *cokoliv jiného* = jiná chyba (obsahem může být chybové hlášení)

Response 1 – vše v pořádku

```
<pingResponse xmlns="http://oprdu.nikde.eu/odu01ws">
   <server>ok</server>
   <login>ok</login>
</pingResponse>
```

Response 2 – neplatné přihlašovací údaje

```
<pingResponse xmlns="http://oprdu.nikde.eu/odu01ws">
   <server>ok</server>
   <login>bad</login>
</pingResponse>
```

Response 3 – server nefunguje

```
<pingResponse xmlns="http://oprdu.nikde.eu/odu01ws">
   <server>down</server>
   <login>fail</login>
</pingResponse>
```