VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

BIBLIOGRAFICKÉ CITACE V DOCBOOKU A JEJICH TRANSFORMACE POMOCÍ DOCBOOK XSL

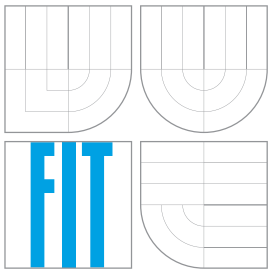
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

TOMÁŠ SOUKOP

BRNO 2009

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

BIBLIOGRAFICKÉ CITACE V DOCBOOKU A JEJICH TRANSFORMACE POMOCÍ DOCBOOK XSL

BIBLIOGRAPHIC CITATIONS IN DOCBOOK AND THEIR TRANSFORMATIONS VIA DOCBOOK XSL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

TOMÁŠ SOUKOP

VEDOUCÍ PRÁCE
SUPERVISOR

Mgr. MAREK RYCHLÝ

BRNO 2009

Zadání bakalářské práce/5095/2008/xsouko00

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2008/2009

Zadání bakalářské práce

Řešitel: **Soukop Tomáš**
Obor: Informační technologie
Téma: **Bibliografické citace v DocBooku a jejich transformace pomocí DocBook XSL**
Kategorie: Web

Pokyny:

1. Seznamte se s formátem DocBook, prostředky prezentace dokumentů v tomto formátu, a s balíkem DocBook XSL a jeho rozšiřitelností.
2. Zmapujte stav transformací bibliografických citací (dále jen BC) v DocBooku pomocí aktuální verze DocBook XSL. Zaměřte se především na způsob prezentace BC a vztah k uznávaným stylům (norma ČSN a jiné). Navrhněte způsoby prezentace DocBook dokumentu tak, aby BC ve výsledku odpovídaly uznávaným stylům.
3. Prozkoumejte nástroje pro práci s BC v DocBooku (např. *RefDB*, *BibTeX as XML* a *JReferences*), srovnejte jejich možnosti a způsoby začlenění BC do dokumentu a prezentace BC. Uplatnění nástrojů demonstrujte na vzorových work-flows. Srovnejte s nástrojem *BibTeX* pro systémy TeX.
4. Navrhněte modifikaci DocBook XSL pro prezentaci BC v souladu s uznávanými styly. Implementujte rozšíření pro některé významné styly (ČSN, IEEE apod.).
5. Zhodnoťte výsledky a navrhněte možná rozšíření.

Literatura:

- Norman Walsh. DocBook: *The Definitive Guide*. O'Reilly, 2005.
- Bob Stayton. *DocBook XSL: The Complete Guide*. Sagehill Enterprises. 2005. [http://www.sagehill.net/docbookxsl/]
- *RefDB Homepage*. [http://refdb.sourceforge.net/]
- *JReferences Documentation*. [http://jreferences.sourceforge.net/]
- *BibTeX as XML markup*. [http://bibtexml.sourceforge.net/]

Při obhajobě semestrální části projektu je požadováno:

- Bod 1. a 2.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Rychlý Marek, Mgr.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2008
Datum odevzdání: 20. května 2009

L.S.

doc. Dr. Ing. Dušan Kolář
vedoucí ústavu

Abstrakt

Bakalářská práce se zabývá prozkoumáním a vysvětlením pojmu DocBook XSL. Popisuje vývoj značkovacích jazyků od SGML ke XML a XSL, kde se vysvětluje, jak a s pomocí jakých nástrojů probíhají transformace mezi těmito formáty do jiných formátů. Zkoumá nástroje vytvořené pro správu bibliografických citací (RefDB, Bibtex to XML, jReferences, JabRef) a popisuje tvorbu vlastní modifikace (šablony), která formátuje citace v normě LNCS pro DocBook.

Abstract

This thesis describes condition of bibliographic citations via DocBook XSL. Ilustrates evolution from SGML to XML and XSL languages. How XSLT transformations work between these formats. How work tools special created for work with DocBook and bibliographic citations (RefDB, Bibtex to XML, jReferences, JabRef). Then thesis explains how was created a stylesheet for formatting citations in LNCS norm for DocBook.

Klíčová slova

DocBook, transformace, XSL, XSLT, XML, SGML, DTD, CSS, HTML, FO, xsltproc, saxon, refdb, bibtexml, jreferences, jabref, LNCS

Keywords

DocBook, transformations, XSL, XSLT, XML, SGML, DTD, CSS, HTML, FO, xsltproc, saxon, refdb, bibtexml, jreferences, jabref, LNCS

Citace

Tomáš Soukop: Bibliografické citace v DocBooku a jejich transformace pomocí DocBook XSL, bakalářská práce, Brno, FIT VUT v Brně, 2009

Bibliografické citace v DocBooku a jejich transformace pomocí DocBook XSL

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Marka Rychlého

.....
Tomáš Soukop
19. května 2009

Poděkování

Chtěl bych poděkovat mému vedoucímu práce panu Marku Rychlému za ochotu a poskytnutou pomoc v rámci řešení problémů.

Obsah

Kapitola 1

Úvod

Ve světě publikací a dokumentů jsou kladeny stále přísnější požadavky pro jejich tvorbu. Vytváří se různá pravidla či normy, které se stávají platnými a které je potřeba v zájmu širší veřejnosti akceptovat a dodržovat.

Dávno uplynula doba, kdy se publikace tiskly mechanicky. S jejím vývojem se začal rozvíjet elektronický zápis, který je efektivní a více komfortní. Programátoři se začali zamýšlet, jak tuto tvorbu stále více obohacovat. V současnosti vznikají programy různých druhů, začínající WYSIWYG editory (intuitivní, člověk při psaní vidí vše tak, jak bude vypadat výsledný dokument) a končící editory či prostředími, ve kterých je spolu s textem vidět programátorský kód. Takový kód označuje určité chování nebo stav (zda bude vybrán tento text, nebo jiný, zda bude mít červenou barvu nebo strojový tvar písmene, atd.). Překlad pomocí různých dostupných nástrojů nám vytvoří ucelený dokument tak, jak jsme v praxi zvyklí tyto dokumenty vidět.

1.1 Požadavky na elektronické publikace

Požadavky lze shrnout do třech základních kategorií. Tyto kategorie jsou z obecného hlediska důležité pro všechny typy nástrojů, které tvoří elektronické dokumenty, nejen pro DocBook, o kterém se později rozhovořím.

1. generování dokumentů do různých formátů
   - formáty pro tisk - Postscript, PDF, RTF, aj.
   - formáty nápověd - HTML help, javahelp, info
   - formát pro web - HTML, XHTML
2. využití schopnosti výstupních formátů
   - obsahy
   - rejstříky
   - odkazy
   - atd.
3. podpora častých změn v dokumentech
   - generování výstupních formátů by se mělo provádět automaticky

1.2 O čem to vše vlastně bude

Bakalářská práce je zaměřena na studium prostředků pro prezentaci dokumentů v systému DocBook XSL a s jeho rozšířeními. Na základě studia tohoto systému je potřeba poznatky nějak srozumitelně popsat a zapsat tak, aby byla do budoucna veřejnosti dokumentační problematika více jasná. I když je DocBook už nějaký ten čas mezi námi, stále je moc „roztříštěn“. Nastává tedy částečně problém s různým nastavením, zprovozněním ap. Cíl této práce je rozdělen na dvě části. Jedním je snaha o vysvětlení, co vlastně DocBook, XML, XSLT je a jak to vše spolu funguje. Druhou částí cíle mojí práce je tvorba určitého rozšíření, konkrétně XSL šablony, upravující styl bibliografických citací podle normy LNCS, která je založena na šabloně LNCS, vydané v LaTeXu.

Popis systému DocBook a jeho transformací bude popsán do té míry, aby si čtenář dokázal udělat vlastní obrázek, vlastní představu, jak takový systém pracuje a co je k němu vše potřeba. V žádném případě zde nechci jednotlivé části popsat do hloubky. To by ani nešlo, už jenom proto, že bakalářská práce je limitována počtem stran a popsat náležitosti okolo DocBooku, jeho gramatiky, historie nebo transformací a nástrojů, které se systémem spolupracují v širším spektru, by vyšlo na opravdu rozsáhlou publikaci. Proto se chci zaměřit více na obecné informace. Některé informace, které nejsou pro představu zas tak důležité, naprosto vynechám, nebo se o nich jen letmo zmíním. Možná vám bude text částěčně připadat jako nějaká učebnice, ale pro představu a popis DocBooku a značkovacích jazyků je to z mého pohledu nezbytné.

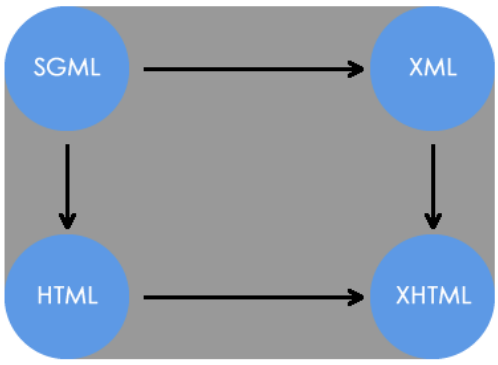
Kapitola 2

Odněkud se začít muselo

Největším přínosem pro tvorbu dokumentů byl bezesporu vznik univerzálního značkovacího jazyka `SGML` [19, 20, 17]. Tento jazyk vznikl v rámci projektu ODA (Open Document Architecture), jehož cílem bylo stvořit standard pro vytváření, uchování, zpracování různých dokumentů v elektronické podobě. Jeho autory jsou Charles Goldfarb, Edward Mosher a Raymond Lorie. Všichni tři autoři pracovali v roce 1960 ve firmě IBM a tehdy stvořili jazyk známý jako `Generalized Markup Language`. Iniciály jejich příjmení vytvořily jazyku zkratku jako GML. Později se tento jazyk stal standardem ISO a byl přejmenován na `Standard Generalized Markup Language`.

Definice 1 *Značkovací jazyk je takový jazyk, který dokáže pomocí značek (též markup) rozlišit text tak, že výsledný dokument se bude formátovat, uspořádávat podle těchto značek a je tedy možné vidět text tak, jak bylo zamýšleno.*

Ač SGML neobsahuje žádnou sémantiku, nabídl člověku nový pohled na tvorbu dokumentů. Postupem času začal být používanější a nároky na něj byly stále větší a větší. Proto se začalo s tímto jazykem experimentovat a nakonec vznikly dvě sestřičky - `XML` (eXtensible Markup Language) a `HTML` (Hypertext Markup Language). Jazyky XML a HTML se později spojily dohromady a vznikl nám hybrid známý jako `XHTML` (eXtensible Markup Language). Schéma vývoje je na obrázku 2.1.



Obrázek 2.1: Schéma vývoje značkovacích jazyků

Jazyk HTML byl vytvořen výhradně pro prostředí webu (internetu, chcete-li). Vychází ze SGML jako aplikace a jednotlivým značkám přiřazuje pevnou sémantiku. Což znamená, že značky rozlišují, jak se mají chovat ke svým informacím.

Definice 2 *Univerzální jazyk je takový jazyk, který neobsahuje sémantiku.*

XML vzniklo jako revize standardu SGML. Tedy původní význam SGML byl zachován. Šlo o vylepšený způsob zápisu elementů a jejich atributů. Původní univerzální značkování, tedy značkování bez známek sémantiky, je zde zachováno. Jazyk je oproti SGML značně „vyčištěn“ a zjednodušen.

Jazyk XML má v dnešní době nejširší využití v oblasti tvorby webových stránek. A to v případě XHTML, kdy došlo k rozšíření HTML o XML elementy. Tím se dosáhlo rychlejšího a jednoduššího přístupu čtení stránek.

XML ovšem není jen o Webu. Jeho uplatnění lze také nalézt v databázích, jako jsou různé seznamy klientů nebo konfiguračních souborů. A v neposlední řadě právě při tvorbě publikací, což je také tématem mé bakalářské práce.

Kapitola 3

DocBook

V poslední době se stále více a více rozšiřuje trend, kdy potřebujeme dokumenty zpracovat nejen v tištěné podobě, ale i ve formě elektronické. Existuje celá škála programů vytvořených pro papírový tisk dokumentů. Nicméně pro elektronický tisk takových programů či systémů moc není. Jedním z hojně se rozšiřujících systémů je právě DocBook. Jedná se o aplikaci, která se vyvinula z jazyků SGML/XML, a co se týče rozšířenosti, stojí hned za HTML.

DocBook vznikl v roce 1991. Na jeho vývoji se podílelo mnoho firem, například Novell, Hewlett-Packard nebo Futjitsu. V roce 1999 spadá DocBook pod sdružení OASIS.

3.1 Co je to DocBook?

Je to systém tvorby dokumentů založených na XML [13, 5, 12]. DocBook není nic jiného než `DTD` (Document Type Definition) [16, 15] právě pro XML. DTD je typ gramatiky, která definuje, jaké elementy a atributy můžeme v XML dokumentu používat. A protože se DTD neustále vyvíjí, odlišují se tím i verze DocBooku. Dva základní typy DocBooku můžete vidět v sekci 3.2. Nevýhodou DTD gramatiky je zastaralost, težkopádnost, nízká flexibilita popisu a hlavně její definice nejsou přímo součástí XML, ale jsou odlišeny v souborech `.dtd`. Poslední verze 5 je kompletně přepsána do lepší gramatiky `Relax NG`, která už součástí XML dokumentů je.

Příklad jednoduchého DTD dokumentu `gramatika.dtd`:

```
<!ELEMENT zamestnanec (jmeno, adresa*)>
<!ELEMENT jméno       (#PCDATA)>
<!ELEMENT adresa      (ulice?, psc?, mesto)>
<!ELEMENT ulice       (#PCDATA)>
<!ELEMENT psc         (#PCDATA)>
<!ELEMENT mesto       (#PCDATA)>
```

A k němu XML soubor `deklarace.xml`:

```
<?xml version=,,1.0‘‘ encoding=,,UTF-8‘‘?>
<!DOCTYPE zamestnanec SYSTEM ,,gramatika.dtd‘‘>
<zamestnanec>
  <jméno>Tomáš Soukop</jméno>
  <adresa>
```

```
      <ulice>Zlatá Hora 1370</ulice>
      <psc>604 01</psc>
      <mesto>Slavkov u˜Brna</mesto>
    </adresa>
  </zamestnanec>
```

Jak takové DTD nebo Relax NG pracuje, není součástí bakalářské práce, a proto zde není pro hlubší průzkum místo. Pokud by toto téma někoho zajímalo hlouběji, doporučuji příslušnou literaturu.

3.2 Verze

DocBook se stále vyvíjí a jeho verze se dnes objevují ve dvou variantách. Jedná se o verze 4.5 (3.2) a 5 (3.2). Musíme si dát ovšem pozor na hlavní verze. Odlišnosti v jejich zápisu vidíte níže. Obvykle totiž nejsou zpětně kompatibilní. Jedině verze vnitřní, tím myslím například verze 4.2 nebo 4.3, jsou zpětně kompatibilní. Verze 4.5 je poslední ze čtvrté řady a také poslední verzí, která využívá DTD gramatiku. Verze 5 už kompletně využívá vylepšenou gramatiku Relax NG.

DocBook verze 4.5 v DTD

```
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE book PUBLIC '-//OASIS//DTD DocBook XML V4.5//EN'
         'http://www.oasis-open.org/docbook/xml/4.5/docbookx.dtd'>
<book lang=,,cs‘‘>
  <bookinfo>
    <title>Jednoduchá kniha</title>
    <subtitle>pro DocBook</subtitle>
  </bookinfo>
  <preface>
    <title>Úvod</title>
    <para>Odstavec s textem úvodu</para>
  </preface>
  <chapter>
    <title>Název kapitoly</title>
    <para>Odstavec první kapitoly</para>
  </chapter>
</book>
```

DocBook verze 5 v Relax NG

```
<?xml version=,,1.0‘‘ encoding=,,UTF-8‘‘?>
<book xml:id=,,jednoducha_kniha‘‘
    xmlns=,,http://docbook.org/ns/docbook‘‘
    version=,,5.0‘‘ xml:lang=,,cs‘‘>
    <title>Jednoduchá kniha</title>
    <chapter xml:id=,,Kapitola_1‘‘>
        <title>Kapitola 1</title>
        <para>Odstavec první kapitoly</para>
```

```
    </chapter>
    <chapter xml:id=,,Kapitola_2‘‘>
        <title>Kapitola 2</title>
        <para>Odstavec druhé kapitoly</para>
    </chapter>
</book>
```

Před skoro 20 lety XML ještě neexistovalo. Proto DocBook existuje ještě ve formě SGML.

Samotný DocBook ke své práci ještě potřebuje tzv. styly. Styly určují, jak bude výsledný dokument vypadat a jak bude zformátován. Jedná se o obdobu kaskádových stylů CSS pro HTML, ale i tyto styly může DocBook spracovávat. Nicméně CSS není tak rozšířené, jako jsou XSL nebo DSSSL styly (bráno s ohledem na DocBook, ne na HTML). Více o stylech a transformacích najdete v kapitole 4.

3.3 Výhody publikování

Publikování dokumentů v XML má jednu obrovskou výhodu oproti publikování v jiných formátech a prostředích. Tou výhodou je opravdu jednoduchá syntaxe a tedy i snadný přístup datům, jakožto textu. Toho se dá samozřejmě využít v různých nástrojích, které ze XML umí číst. XML dokumenty jsou hierarchicky uspořádány, proto se v nich dá rychle zorientovat. Implemtentace rozpoznání XML syntaxe v takových nástrojích je přitom velice přehledná, rychlá a hlavně jednoduchá.

Publikování v DocBooku se vyplatí, pokud:

- výstup dokumentu potřebujeme mít v různých formátech - PS, PDF, HTML, Nápovědy apod. Díky jednoduchosti XML je konverze do jiných formátů velice snadná.
- vytváříme dokumenty rozsáhlejšího charakteru - technické, vědecké práce, programové dokumentace, encyklopedie atd. Jiné dokumentační systémy mohou být limitovány svojí velikostí nebo nestabilitou při takové velikosti.
- vytváříme větší množství dokumentů - časopisy, sborníky, technické podpory. Jsou to dokumenty s pevnou strukturou, ve které se mění jen obsah, ale grafické uspořádání musí být stejné.

Kapitola 4

Transformace

Pojem `transformace` značí určitou přeměnu v něco. V našem případě se jedná o přeměnu kombinace dvou a více formátů v jeden výstupní, který může, ale nemusí být zpracováván dále (viz. obrázek 4.1). K transformaci potřebujeme určitě text ve formátu XML. Mluvíme-li stále o dokumentech a publikacích, budeme mít text zapsán v DocBooku.

Značení v XML [10] neříká nic o tom, jak bude text zformátován. Základní výhodou XML dokumentu je totiž jeho oddělení informačního textu od grafického zpracování. Grafický vzhled se potom definuje pomocí `stylů` nebo také `stylových jazyků`. V takových jazycích můžeme nadefinovat vlastní vzhled elementů a atributů obsažených v XML.

Výhodou stylových jazyků je, že zachovávají jednotný vzhled dokumentu a také fakt, že lze kombinovat několik stylových souborů stejného druhu do sebe. Tedy pokud zamýšlíme například změnit pozici obrázku v dokumentu nebo přebarvit nadpis na červeno, stačí příslušnou změnu provést v souboru se stylem. Kombinací stylů je myšlen záměr, kdy z jednoho stylu voláme například tři další, kde každý z nich provádí něco jiného. Tento způsob zápisu pak vytváří velice přehledný prostor, kde je potom v kódu jednoduché nebo jednodušší se zorientovat. Všechny dokumenty, na které se pak styl aplikuje, získají nový vzhled.

Stylových formátů je poměrně dost druhů a právě dost jich je podporováno DocBookem. Tedy můžeme si vybrat, který z nich to bude. Existuje jich celá škála, ale nejrozšířenější jsou `CSS`, `DSSSL` a `XSL` [9].

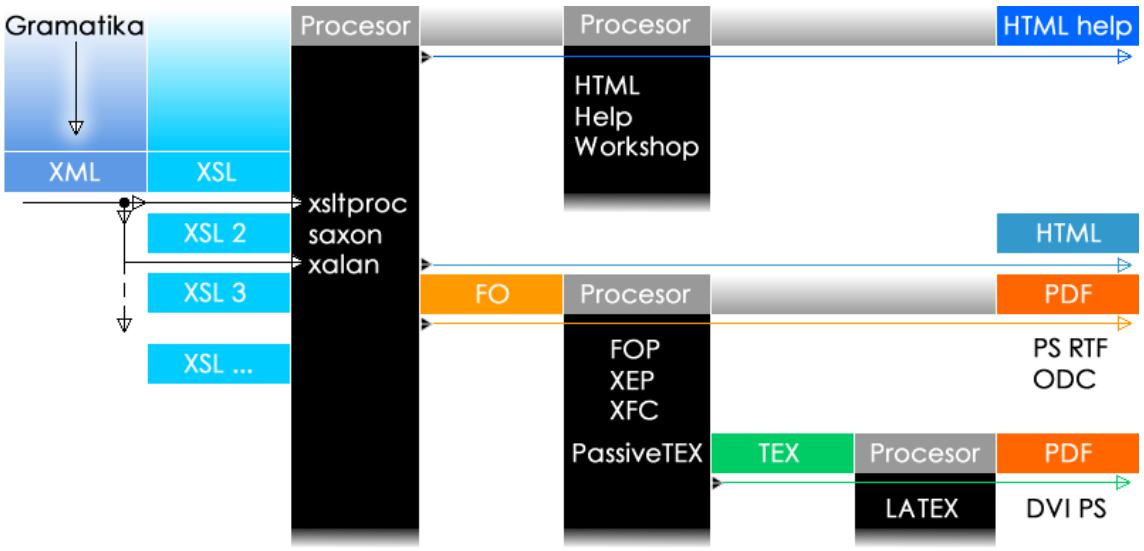
- CSS (Cascading Style Sheets) - je styl určený převážně pro HTML a zpracování na internetu. Podporuje jej zdaleka nejvíce aplikací, protože CSS patří k nejstarším stylům a zároveň vyniká svojí jednoduchostí zápisu. V dnešní době má CSS vytvořeny dvě své verze. Na třetí se prozatím pracuje. Původní verze byla určena spíše pro prezentaci dokumentů na obrazovce. Druhá verze rozšiřuje CSS o parametry pro tištěný výstup nebo množstvím rozšíření pro potřeby XML.

- DSSSL (Document Style Semantics and Specification Language) - tyto styly jsou takovým opakem k CSS. Je to rozsáhle vyvinutý jazyk, který byl původně vytvořen pro SGML, ale podpora pro XML tu je také. Komplexnost DSSSL je tak velká, že jej mnoho aplikací nepodporuje. Pro potřeby elektronického publikování byla vytvořena „ořezaná“ verze DSSSL-O, která je vhodná pro méně naročné aplikace.

- XSL (eXtensible Stylesheet Language) - tvorba stylů v tomto jazyce je pro mnohé aplikace střední cestou mezi CSS a DSSSL, alespoň co se týká jednoduchosti zápisu. Výhodou je, že se XSL zapisuje stejným způsobem jako XML, tedy při práci můžeme využívat stejného editoru.

O XSL (4.2) se zde rozhovořím podstatně více, protože je součástí zadání mé práce a také proto, že si prostě zaslouží více prostoru, hlavně při práci s XML.

Poznámka - pokud se někde v textu zmíním o překladači, transformátoru nebo procesoru, jedná se o ten samý program, který dokáže transformovat jeden formát na druhý (xsltproc, saxon, FOP, atd.).



Obrázek 4.1: Schéma transformace

4.1 Jmenné prostory

Každé rozšíření o jakékoli XML, XSL nebo FO a další značky je samozřejmě potřeba náležitě oznámit. K tomuto oznámení nám slouží tzv. `Jmenné prostory` (Namespace). Jmenné prostory je potřeba zapsat hned na začátek každého XML nebo XSL dokumentu, aby se vědělo, jaký seznam tagů, neboli značek, se v dokumentu vyskytuje. Je vhodné říci, že již zmíněná gramatika DTD se musí zde také zapsat, protože se v ní vyskytují pravidla a značky, které potom můžeme v XML dokumentu používat. Zápis gramatiky v DocBooku jako jmenný prostor vidíte níže.

```
                <?xml version='1.0' encoding='utf-8'?>
                <!DOCTYPE book PUBLIC '-//OASIS//DTD DocBook XML V4.5//EN'
namespace ->    'http://www.oasis-open.org/docbook/xml/4.5/docbookx.dtd'>
                <book lang=,,cs‘‘>
                ...
```

Pro DocBook verzi 5 v gramatice Relax NG lze zapsat jmenný prostor následovně:

```
                <?xml version=,,1.0‘‘ encoding=,,UTF-8‘‘?>
                <book xml:id=,,jednoducha_kniha‘‘
namespace ->    xmlns=,,http://docbook.org/ns/docbook‘‘
                version=,,5.0‘‘ xml:lang=,,cs‘‘>
                ...
```

Pokud budeme v využívat stylový soubor XSL, ve kterém budeme používat značky přislušné XSL a FO objektům, musíme nadeklarovat na začátek dokumentu neco takového:

```
              <xsl:stylesheet
namespace ->  xmlns:xsl=,,http://www.w3.org/1999/XSL/Transform‘‘
namespace ->  xmlns:fo=,,http://www.w3.org/1999/XSL/Format‘‘
              version=,,1.0‘‘>
              ...
```

4.2 XSL

Jazyk XSL je asi nejvýhodnější variantou mezi CSS a DSSSL pro XML. Jedná se o univerzální procedurální jazyk pro tvorbu stylů, který nabízí ve větší míře funkčnost ostatních stylových jazyků a zároveň něco navíc.

Podle standardu *Extensible Stylesheet Language (XSL) Version 1.0 W3C Working Draft* [4] rozdělujeme XSL na dvě hlavní části. První se věnuje transformacím, tzv. `eXtensible Stylesheet Language Transfomations` (XSLT). Druhá `formátovacím objektům` (FO), které slouží jako abstraktní popis vzhledu stránek a jejích komponent a i tyto formátovací objekty mají syntaxi XML.

4.2.1 XSLT

Transformace XSL se skládají ze šablon. Šablony určují na jaké části vstupního dokumentu jsou použity a jak budou části transformovány na výstup.

Pro zápis průchodu vstupního dokumentu se v šablonách využívá jazyka `XPATH` (XML Path Language). Je to speciální jazyk, který slouží k výběru částí XML dokumentu. Dokument je pro tento jazyk chápán jako stromová hierarchie, kde uzly jsou tvořeny elementy, atributy a obsahem elementů. Jazyk potom prochází jednotlivé elementy, popřípadě atributy, a vybírá jejich obsah a zpracovává jej dál.

Šablona zapsaná v XSL, která vybírá určité elementy může vypadat například následovně:

```
<xsl:template match=,,shop‘‘>
    <xsl:call-template name=,,newmusic‘‘/>
</xsl:template>

<xsl:template name=,,newmusic‘‘>
    <xsl:text>Toto je vzorek pro vložení nové muziky</xsl:text>
</xsl:template>
```

Tahle šablona nám říká:

> *Jestliže existuje element* `shop`*, pak zavolej vzorek* `newmusic`*, který vytiskne text* `Toto je vzorek pro vložení nové muziky`.

V XPATH je zavedeno velké množství pravidel, které říkají jaké části XML dokumentu se mají vybrat, pomocí kterých atributů, jaký počet elementů atp. Na ukázku přikládám a vysvětlím pár z nich.

- `.//para` - vybere všechny elementy `para`, které jsou potomky aktuálního. uzlu.
- `para[@type=warning][5]` - vybere všechny elementy `para`, kde atribut v `para` se jmenuje `type` a jemu přiřazena hodnota `warning`. Číslo `[5]` říká, že jestli se v dokumentu vyskytuje pět takových stejných elementů, pak se vybere teprv páty z nich.
- `kapitola[nadpis=Úvod]` - vybere všechny elementy `kapitola`, kde uvnitř kapitoly existuje element `nadpis` a tento element `nadpis` obsahuje text `Úvod`.

S XPATH ještě souvisí další jazyk. Jedná se o `XLINK (XML Linking Language)` a jeho rozšíření `XPointer` (XML Pointer Language). Xlink je jazyk drobně rozšiřující působnost XPATH a je zahrnut mezi standardy konsorcia W3C. Pokud potřebujeme vytvořit odkazy mezi dokumenty, pomůže nám k tomu právě Xlink. XPointer do Xlinku přidává možnost připojit na konec URL výraz, který určuje část celého dokumentu.

Malý příklad pro vytvoření odkazů:

```
Na stránce
<xlink:simple href=,,http://www.kosek.cz‘‘>
pana Koska
</xlink:simple>
naleznete mnoho užitečných informací o XML.
```

XPointer potom přidává ještě:

```
...
  <chapter id=,,kap1‘‘>
  ...
  </chapter>
...
<xlink:simple href=,,dokument.xml#kap1‘‘>odkaz</xlink:simple>
```

Příklad XML dokumentu se svojí DTD gramatikou

Nyní předvedu styl, který vám transformuje informace obsažené v XML dokumentu do HTML stránky a patřičně ji upraví:

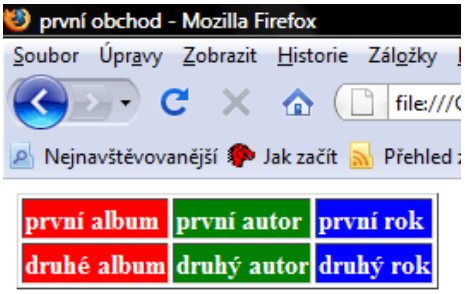
```
<?xml version=,,1.0‘‘?>
<!-- deklarace gramatiky, aby se vedelo,
ktere tagy souvisi s kteryma -->
<!DOCTYPE shop [
<!ELEMENT shop   (name, music+)>
<!ELEMENT name   (#PCDATA)>
<!ELEMENT music  (album, author, year)>
<!ELEMENT album  (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT year   (#PCDATA)>
```

```
]>
<!-- zapis tagu do struktury -->
<shop>
<name>první obchod</name>
<music>
  <album>první album</album>
  <author>první autor</author>
  <year>první rok</year>
</music>
<music>
  <album>druhé album</album>
  <author>druhý autor</author>
  <year>druhý rok</year>
</music>
</shop>
```

Příklad XSL stylu komunikující s XML

```
<?xml version=,,1.0‘‘ encoding=,,utf-8‘‘?>
<xsl:stylesheet
xmlns:xsl=,,http://www.w3.org/1999/XSL/Transform‘‘ version=,,1.0‘‘>
<xsl:template match=,,shop‘‘>
    <html>
    <head><title><xsl:value-of select=,,name‘‘/></title></head>
    <body>
          <xsl:call-template name=,,newmusic‘‘/>
    </body>
    </html>
</xsl:template>
<xsl:template name=,,newmusic‘‘>
    <table border=,,1‘‘>
        <xsl:for-each select=,,music‘‘>
            <tr>
            <td style=,,color: white; background-color:red;‘‘>
                <xsl:value-of select=,,album‘‘/>
            </td>
            <td style=,,color: white; background-color:green;‘‘>
                <xsl:value-of select=,,author‘‘/>
            </td>
            <td style=,,color: white; background-color:blue;‘‘>
                <xsl:value-of select=,,year‘‘/>
            </td>
            </tr>
        </xsl:for-each>
    </table>
</xsl:template>
</xsl:stylesheet>
```

Výsledek vidíte na obrázku 4.2.



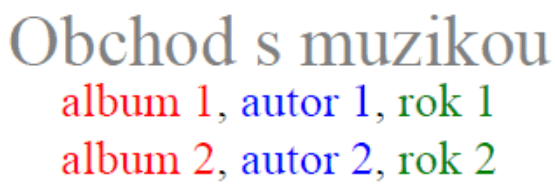
Obrázek 4.2: Výsledek přeložení do HTML

4.2.2 FO

Převést XML do HTML je, jak jste viděli, velice jednoduché. Nicméně šlo o převod do elektronické podoby pro Web. Nyní potřebujeme XML dokument převést do formátů PDF, PostScript nebo RTF, které jsou určené pro tisk nebo vytvořit náhledy pro prezentace. Pro takový tisk nám slouží `formátovací objekty` (FO). FO nám poskytují mechanismy, které dokáží měnit rozměry stránek, způsob zarovnávání, velikosti, barvy písem apod.

Celý princip je velice jednoduchý. Pomocí XSLT a FO vytvoříme XSL styl, který nám spolu s XML dokumentem při překladu pomocí některého XSLT procesoru vytvoří soubor s příponou `.fo`. Tento soubor lze následně pomocí FO procesoru převést na některý formát pro tisk, jako je PDF, PS, RTF atd. Samozřejmě, že lze vytvořit FO soubory rovnou a značky psát přímo do jeho kódu, ale hlavní nevýhoda FO je jeho složitost. Obecně se překlad dělá pomocí některého XSLT procesoru, proto také není potřeba a prakticky ani důvod specifikaci gramatiky FO zjednodušovat.

Jak víme, pro úspěšné propojení FO objektů s XSL stylem musíme nejprve do XSL zahrnout jmenný prostor. Toho dosáhneme připojením FO prostoru, které jsem už ukázal v předchozí kapitole (4.1). Nyní můžeme ukázat jednoduchý kód demonstrující použití XSLT + FO. Posléze ukáži kód vytvořený přímo ve FO (4.2.2), určený k přímému překladu do tiskového formátu, který byl vygenerován XSLT procesorem z předchozího XML dokumentu a následujícího XSL stylu (4.2.2).



Obrázek 4.3: Výsledek překladu z FO do PDF

Příklad XSL-FO

```
<?xml version=,,1.0‘‘ encoding=,,utf-8‘‘ ?>
<xsl:stylesheet xmlns:xsl=,,http://www.w3.org/1999/XSL/Transform‘‘
                xmlns:fo=,,http://www.w3.org/1999/XSL/Format‘‘
                version=,,1.0‘‘>
  <xsl:template match=,,shop‘‘>
    <fo:root>
      <fo:layout-master-set>
        <fo:simple-page-master margin-bottom=,,1.5cm‘‘
                               margin-left=,,2cm‘‘
                               margin-right=,,2cm‘‘
                               margin-top=,,1.5cm‘‘
                               master-name=,,HEAD_BODY‘‘>
          <fo:region-body />
        </fo:simple-page-master>
      </fo:layout-master-set>
      <fo:page-sequence master-reference=,,HEAD_BODY‘‘>
        <fo:flow flow-name=,,xsl-region-body‘‘
                 font-family=,,Times‘‘
                 font-size=,,13pt‘‘ line-height=,,120%‘‘>
          <xsl:apply-templates select=,,name‘‘/>
          <xsl:call-template name=,,mymusic‘‘/>
        </fo:flow>
      </fo:page-sequence>
    </fo:root>
  </xsl:template>

  <xsl:template match=,,name‘‘>
    <fo:block font-size=,,150%‘‘ color=,,grey‘‘ text-align=,,center‘‘>
      <xsl:value-of select=,,.‘‘ />
      <xsl:text />
    </fo:block>
  </xsl:template>

  <xsl:template name=,,mymusic‘‘>
    <xsl:for-each select=,,music‘‘>
      <fo:block text-align=,,center‘‘>
        <fo:inline color=,,red‘‘><xsl:value-of select=,,album‘‘ />
        </fo:inline><xsl:text>, </xsl:text>
        <fo:inline color=,,blue‘‘><xsl:value-of select=,,author‘‘ />
        </fo:inline> <xsl:text>, </xsl:text>
        <fo:inline color=,,green‘‘><xsl:value-of select=,,year‘‘ />
        </fo:inline>
      </fo:block>
    </xsl:for-each>
  </xsl:template>
</xsl:stylesheet>
```

Příklad FO

Vygenerovaný FO soubor vypadá následovně:

```
<?xml version=,,1.0‘‘?>
<fo:root xmlns:fo=,,http://www.w3.org/1999/XSL/Format‘‘>
  <fo:layout-master-set>
    <fo:simple-page-master margin-bottom=,,1.5cm‘‘
                           margin-left=,,2cm‘‘
                           margin-right=,,2cm‘‘
                           margin-top=,,1.5cm‘‘
                           master-name=,,HEAD_BODY‘‘>
      <fo:region-body/>
    </fo:simple-page-master>
  </fo:layout-master-set>
  <fo:page-sequence master-reference=,,HEAD_BODY‘‘>
    <fo:flow flow-name=,,xsl-region-body‘‘
             font-family=,,Times‘‘
             font-size=,,13pt‘‘ line-height=,,120%‘‘>
      <fo:block font-size=,,150%‘‘ color=,,grey‘‘ text-align=,,center‘‘>
      Obchod s˜muzikou
      </fo:block>
      <fo:block text-align=,,center‘‘>
        <fo:inline color=,,red‘‘>album 1</fo:inline>,
        <fo:inline color=,,blue‘‘>autor 1</fo:inline>,
        <fo:inline color=,,green‘‘>rok 1</fo:inline>
      </fo:block>
      <fo:block text-align=,,center‘‘>
        <fo:inline color=,,red‘‘>album 2</fo:inline>,
        <fo:inline color=,,blue‘‘>autor 2</fo:inline>,
        <fo:inline color=,,green‘‘>rok 2</fo:inline>
      </fo:block>
    </fo:flow>
  </fo:page-sequence>
</fo:root>
```

Tento FO dokument lze posléze přeložit některým FO procesorem do výstupních formátů jako jsou PDF, PostScript, RTF, TEX nebo ODC. Všechny zmíněné příklady jsou přeložitelné a naleznete je i na přiloženém CD. Výsledek přeložení je vidět na obrázku 4.3.

4.3 Transformační nástroje

Transformačními nástroji rozumíme aplikace, které dokáží převádět určitý typ formátu na jiný. Například mezi XML + XSL na FO, XML + XSL na HTML, FO na PDF atp. Takových programů existuje v dnešní době celé spektrum. Procesory lze rozdělit na dva druhy. Jeden se zabývá přímou XSLT tranformací, druhý FO tranformací. Zde shrnu charakteristiky pár neužívanějších, které jsou spjaty s tvorbou dokumentů.

4.3.1 XSLT procesory

- saxon - v dnešní době se jedná zatím o nejmocnějnější transformační nástroj. Mezi výhody programu patří to, že je napsán Javě, čili jeho spuštění nezávisí na operačním systému, podporuje i XSLT 2.0 a XPATH 2.0 a neustále se vyvíjí.
- xsltproc - nenabízí tak širokou podporu jako Saxon. Je ale jednodušší a v některých případech rychlejší než Saxon. Původně byl vyvinut pro Gnome, ale později se stal standardem na unixových systémech. Program je napsán v C a je dostupný se zdrojovými kódy, takže je pro něj možné vytvářet vlastní rozšíření. Součástí balíčku je i knihovna `iconv`, která umožňuje používat velké množství kódování.

4.3.2 FO procesory

- FOP - FO procesor od firmy Apache (http://xml.apache.org/dist/fop/). Je zadarmo, ale neobsahuje podporu českých znaků. Tu lze nalézt na stránkách Jiřího Koska (http://www.kosek.cz/sw/fop/index.html.).
- XEP - jedná se o komerční FO procesor od firmy RenderX. Firma jej rozděluje na dvě části, jednou je placená verze, druhou verze „personal“, která umožnuje použít program zdarma, ale při překladu budete mít na každé stránce malé logo firmy. Novější verze už podporují české znaky.

4.4 Jak transformovat

Samotná transformace je pro správný zápis relativně jednoduchý proces. Záleží jen na tom, pro který typ překladače se rozhodnete. Odlišnosti jsou časové, peněžní a implementační. Každý z překladačů má svůj styl zápisu. Pro bakalářskou práci jsem se rozhodl používat xsltproc a FOP nebo XEP, proto zde uvedu příklad pro zápis v těchto programech. Každý z programů, i ty, které zde nepopisuji, mají svoji vlastní nápovědu, která ukazuje, s jakými parametry lze překladač spustit. Nápověda ke každému programu lze vyvolat příkazem:

`nazev_programu --help` nebo `nazev_programu -h`

Zápis transformace pomocí příkazové řádky:

- xsltproc:
  ```
  xsltproc -o vystup_ve_fo.fo -s xslstyl.xsl vstup_v_xml.xml
  xsltproc -o vystup_v_html.html -s xslstyl.xsl vstup_v_xml.xml
  ```
- FOP:
  ```
  fop vstup_ve_fo.fo vystup_v_pdf.pdf
  ```
- XEP:
  ```
  xep vstup_ve_fo.fo vystup_v_pdf.pdf
  ```

Pokud používáme prostředí xslt2 [14] od Jana Pavloviče, může vypadat zápis takto:

- `xslt -t foppdf -s myxsl.xsl workflow-lncs.xml`
- `xslt -s xsl2html.xsl xml2html.xml -o vystup_v_html.html`

Kapitola 5

Implementace XSLT stylu

5.1 Stav bibliografie v DocBooku

Pro systém DocBook byla vytvořena vlastní bibliografie. Když jsem tuto původní bibliografii zkoušel přeložit, zjistil jsem, že citace nejsou formátovány podle určité normy. Jednotlivým elementům a atributům zde chyběla něco jako jednotnost. Ve výsledku bibliografie vypadala jako částečně stará norma IEEE někdy z roku 2001 a částečně uplně něco jiného. Připadalo mi a stále připadá, že tato bibliografie je zde zahrnuta jenom proto, aby zde vůbec nějaká byla. Jinak řečeno, jejich kvalita byla hodně špatná. Tento problém byl vyřešen až s příchodem stylu upravující bibliografii podle normy ISO690.

5.2 Menší komplikace

Původně měla bakalářská práce zpracovat implementaci bibliografických citací pro normu ČSN ISO690. Jenomže po přezkoumání celkového pohledu na stav těchto citací pro DocBook bylo zjištěno, že tuhle normu už někdo vytvořil v rámci jiné bakalářské práce. ISO690 zpracovala ve své bakalářské práci Jana Dvořáková.

Její dílo bylo a je velice povedenou prací. Správně formátuje veškeré typy bibliografických citací. Předpokládám, že se inspirovala prací pana Boldiše a jeho pravidel citací normy ISO690 [7, 6], kde vysvětluje jak, co a kde se má správně citovat podle této normy, anebo využila šablony `czechiso.bst` pro normu ISO690 v bibTEXu. Ať tak, či tak, její práce je vydařená.

Pro svoji bakalářskou práci jsem si tedy zvolil úpravu citací podle normy LNCS, kterou trochu více osvětlím v kapitole 5.3.

XSLT je pro začátečníky docela problém, proto také na přiloženém CD uvádím xsl styl a jeho makefile, pomocí kterého se dá formátovaní, jednak podle ISO690, tak podle LNCS, provést (v souboru myxsl.xsl místo LNCS zapíšete ISO690 a formátování se už se provede podle příslušného stylu).

5.3 Norma LNCS

LNCS je zkratkou pro Lecture Notes in Computer Science. Tedy jedná se o sérii prestižní počítačové literatury, která je publikováná pod nakladatelstvím Springer, kde vychází sborníky lepších konferencí. Springer vytvořil vlastní šablonu v LATEXu. Je tedy možné své články, konference atd. publikovat právě v normě LNCS, ale jak už jsem zmínil, pouze pro

LaTeX. Nicméně mít na sborníku nálepku nakladatelství Springer, je o něco složitější problém. Pro mnoho autorů svých publikací je zisk této nálepky pouze sen, protože Springer samotný si sám vybírá, které publikace jsou takové pocty hodny (vybírá si podle kvality ty nejlepší).

5.4 Návrh

Začal jsem tím, že jsem si vytvořil databázi veškerých typů bibliografických citací ve formátu BibTeX. Vyhledal jsem si, které parametry může a musí obsahovat ten, či onen typ publikace, jako je Article, Book, Inbook atp. Jak už jsem napsal výše, nakladatelství Springer si vytvořilo vlastní šablonu (`splncs.bst`) v LaTeXu pro formátování jejich citací, která je volně dostupná na jejich oficiálních webových stránkách [3]. Šablonu jsem použil pro překlad databáze citací, kterou jsem vytvořil. Získal jsem tím výsledný dokument, který ukazuje, jak jsou citace zformátovány, a které parametry jednotlivých typů literatur byly použity. Tento dokument pak významně pomůže v implementaci šablony.

Bylo potřeba nejdříve si namyslet, jak bude značen výsledný XML dokument. Na ukázku jsem přiložil rozvržení pro typy článků (Article). Rozvržení bylo potřeba udělat proto, abych věděl, jak budu dále postupovat, resp. odkud vůbec začít. XSLT a jazyk XPATH má určitou hierarchii, která se musí dodržovat a vytvořit si počáteční systém usnadní a urychlí celkovou práci. Implementaci jsem se rozhodl udělat podle šablony normy ISO690.

```
<biblioentry role=,,article‘‘>
   <biblioset relation=,,article‘‘>
      <author>
         <surname>Dijkstra</surname>
         <firstname>E.W.</firstname>
      </author>
   <title>The Humble Programmer</title>
   </biblioset>
   <biblioset relation=,,journal‘‘>
      <title>Communications of the ACM</title>
         <pubdate>
            <month>October</month>
            <year>1972</year>
         </pubdate>
      <volumenum>15</volumenum>
      <issuenum>10</issuenum>
      <pagenums>859-866</pagenums>
   </biblioset>
</biblioentry>
```

5.5 Implementace

Abych mohl začít implementovat, musel jsem XSLT, XPATH, a v neposlední řadě šablonu ISO690, důkladně nastudovat a pochopit. V minulosti jsem o XSLT a XPATH vůbec nic nevěděl, ale nastudování nebylo zas tak těžké. Problém se ovšem začal rýsovat v přehlednosti a celkové představivosti. XPATH je totiž založen na práci s informacemi, které jsou uloženy na určitých „cestách“ a které symbolizují značky XML.

XSL je procedurálním jazykem. A proto každý „template“ je tvořen vlastní procedurou a uvnitř každé této procedury jsou umístěny prvky, které volají další procedury, které jsou umístěny někde v souboru. Bylo velice důležité si správně rozvrhnout, do kterých míst a ke kterým procedurám se přidají jiné procedury. Usnadnilo to rychlost orientace v kódu. Dalším důležitým aspektem bylo správné pojmenování pro volání jednotlivých procedur. Rozhodně se mi nevyplatilo počáteční pojmenování např. proměnných typu `var 1`, `var 2` apod. Tyto názvy jsem později zavrhl a všechny přejmenoval přesně podle příslušnosti k danému typu citace.

Po nastudování šablony ISO690 jsem zjistil, že použitelná je jen implementace autora a názvu publikace. Vše ostatní jsem takřka smazal a musel naimplementovat znovu. Bylo to dáno především tím, že mnoho pravidel bylo pro moji tvorbu nepotřebných, zbytečných nebo navíc.

Výsledek naimplementování předchozí ukázky článku by pak vypadal takto:

Dijkstra, E.W.: The Humble Programmer. Communications of the ACM. **15** (10) (October 1972) 859-866

Citovaná položka pak odpovídá překladu z bibTEXu pomocí šablony `splncs.bst`.
Jak nainstalovat a spustit LNCS styl je popsán v dodatku B.

Kromě samotné šablony jsem modifikoval i soubory `docbook.xsl`, `biblio.xsl` a `en.xml`. Je to proto, že jsem zamýšlel a spěl k tomu, aby mohla být šablona, stejně jako ISO690, zavedena jako standard do xsl stylů DocBooku. Autoři, podílející se na stylech, by pak o šabloně věděli a směli ji vlastnoručně upravovat a jistým směrem se podílet na jejím vývoji do budoucna. Kvůli tomu je možná její instalace komplikovanější.

Modifikace `en.xml` znamenala přidání kontextu, který ovlivňoval anglickou variantu LNCS. Změny v tomto dokumentu byly provedeny proto, že v budoucnosti by mohla existovat LNCS norma přeložená do češtiny. Například *Technical Report* by mohl být přidán do souboru `cs.xml` jako *Technická zpráva*. Při změně atributu `lang` na `cs` v elementu `bibliography` by se pak po překladu všechny tyto jazykové kontexty měnily na výstupu v závislosti na daném jazyce. Další modifikace byly provedeny v souboru `biblio.xsl`. Zde se musela provést menší úprava, která by dokázala přepnout formátování citací právě do mého stylu s LNCS a zároveň neovlivnila už existující „přepínač“ na normu ISO690. Posledním modifikovaným externím souborem je hlavní „rozcestník“, styl `docbook.xsl`. Zde se musel přidat příkaz, který načte moji modifikaci LNCS ze souboru.

Původně jsem zamýšlel vytvořit dvě verze šablony. Jedna, která by byla zahrnuta mezi standardy, a o které jsem hovořil výše. Druhá, která by byla méně složitá, co se týče její instalace, a která by se dala použít „jednorázově“. To znamená její překlad by byl pouze s XML dokumentem a jedním XSL stylem rovnou provádějící formátování bez toho, aby se prováděla modifikace tří dalších XSL stylů. Toho se mi z nedostatku času nepodařilo docílit a je vytvořena jen verze modifikující i XSL styly DocBooku.

Kapitola 6

Nástroje

6.1 RefDB

6.1.1 Proč RefDB?

Bibliografické citace jako takové jsou obvykle velice zdlouhavé a náročné při jejich zápisu a celkovém vyhledávání. Například lidé, pracující na jakékoli vědecké pozici, musí přečíst, zhodnotit nebo zpracovat mnoho knih. Nemohou ovšem stále dokola přepisovat, od koho daná publikace vzešla, z jakého je roku, či jaké má kniha ISBN číslo. Nebo také naopak. Pokud bude někdo chtít získat z dané publikace citovanou literaturu, aby si třeba ověřil určitá fakta, musí v této publikaci být citována takovým způsobem, a v takovém formátu, aby usnadnila její následné dohledání. Proto se hodně lidí snaží přijít s nástroji, které by správu citací jakýmkoli způsobem usnadnili. Právě snaha přispět nebo vytvořit koncepci pro „globální“ zpracování citací byla podmínkou pro vytvoření referenční databáze. Lidé, kteří aplikaci rozumí, dokáží vytvořit databáze, které jsou pak volně dostupné buď na různých serverech, podporujících toto zpracování přímo, nebo je jen tak „souborově“ nasdílet kdekoli se dá. Uživatelé pak tyto databáze mohou využívat pro citování knih, článků, časopisů nebo jiných odborných či neodborných textů tak, že je nemusí složitě přepisovat do různých formátů (například pro bibTEX, docbook xsl, apod.).

6.1.2 Co je to RefDB?

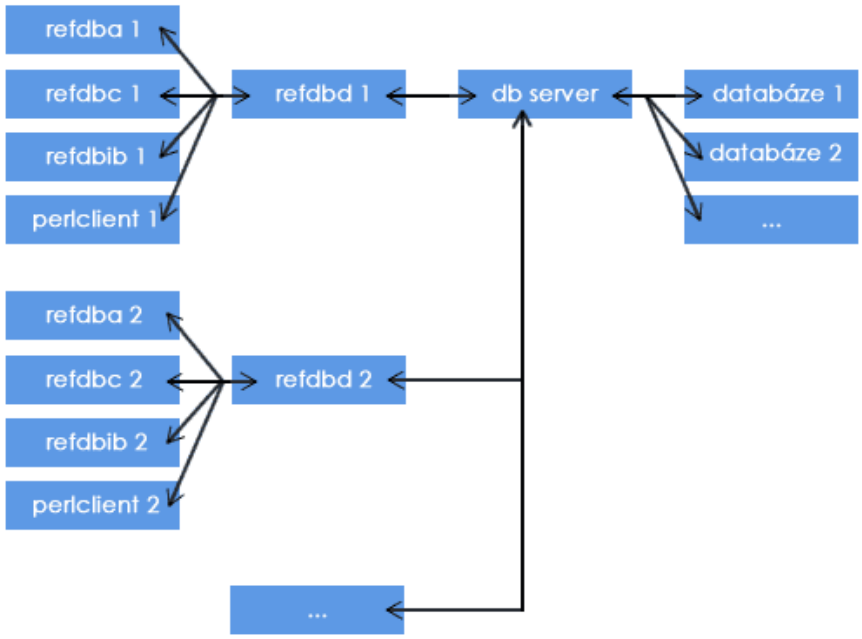
RefDB [8, 18] je referenční databáze typu klient-server, využívající ke své práci externích databází, jako jsou MySQL, PostgreSQL, SQLite nebo SQLite3 pro ukládání různých dat. Ukládat „různá“ data lze jen v rámci možností. Implicitně je referenční databáze spjata s bibliografickými citacemi. Jinak řečeno, je vytvořena právě pro správu bibliografických citací. Taková databáze pak uživatelům usnadňuje přístup k jednotlivým bibliografiím. Například pokud chtějí citovat ve své publikaci dílo někoho jiného, stačí spustit klienta, napsat příslušný příkaz pro nalezení díla a vše potřebné se vám zobrazí. Autorem této aplikace je Markus Hoenicka.

6.1.3 Jak RefDB funguje?

Externí databáze je pro RefDB nutná. Ta musí být správně nastavená. MySQL a PostgreSQL jsou plnohodnotnými databázemi a je potřeba u nich nastavit správná administrátorská práva pro přístup a manipulaci s obsahem. U SQLite a SQLite3 je nastavení trochu

jiné a jednodušší. Jde totiž o určitou lokální databázi a práva zde nejsou zas tak potřeba. SQLite rozhodně doporučuji začátečníkům. Jeho instalace a použití je výrazně jednodušší.

RefDB se skládá ze čtyř hlavních částí: refdbd, refdba, refdbc, refdbib. Schéma vidíte na obrázku (6.1). Práva a nastavení z externích databází pak využije server `refdbd` k přístupu k vytvořeným databázím a jejich obsahu. Pracuje něco jako „vnější povrch“ externí databáze, který ji obalí, získá a pak využije obsah této databáze k potřebným účelům. Je důležité podotknout, že do externí databáze nezasahujeme přímo. S jejími informacemi zacházíme právě prostřednictvím servru `refdbd`. Tyto databáze může měnit systémový klient `refdba` a uživatelský klient `refdbc` pak umožnuje manipulaci s obsahem databází.



Obrázek 6.1: Schéma připojení RefDB k databázi

RefDB se také umí vzdáleně připojovat na jiné servry. Zde oceníte sílu MySQL, či PostgreSQL. Tady už jsou práva pro přístup k nastavení a správu obsahu jistě potřeba. Určitě by se nám nelíbilo, kdyby měl někdo přístup k námi vytvořené databázi. Pokud na vzdáleném servru běží MySQL nebo PostgreSQL, můžeme se k nim jednoduše přes `refdbd` připojit a pomocí klientů tvořit, zpracovávat nebo mazat námi vytvořené data (samozřejmě tím myslím správu bibliografických citací).

Do databáze se může přistupovat například přes formát RIS. Je to univerzální formát, velice zjednodušený, pro zápis bibliografických citací. Pro rozlišení citací je stanoveno pravidlo, kdy každá citovaná položka musí začínat tagem TY (definuje typ dokumentu) a končit tagem ER (prázdný tag značící konec citované publikace).

Příklad takové položky může být následující:

```
TY - BOOK
ID - Harmon88Expert
AU - Harmon,P.
```

```
AU - Maus,R.
AU - Morrissey,W.
CY - New York
PY - 1988///
BT - Experts Systems Tools and Applications
PB - John Wiley & Sons
ER -
```

Ovšem formát RIS není jediný, který se dá pro RefDB použít. Existuje celá škála formátů pro zápis bibliografie. I na tyto formáty se nezapomnělo a ve větší míře jsou zde implementované, zahrnuje BibTEX, RISX, ISI, MODS XML, Medline, PubMed, Endnote, MARC, a Copac.

Díky vytvořeným skriptům v jazyce perl, RefDB poskytuje podporu převodů mezi těmito formáty, například bib2ris, med2ris atd. (více na http://refdb.sourceforge.net/doc.html). Výstupní dokumenty mohou pak existovat buď v těchto formátech, nebo ve formě HTML, XHTML, DocBook, či TEI.

Raději si nástroje RefDB shrneme:

- **refdbd** - jedná se o server komunikující mezi klientskou částí referenční databáze a externí databáze, např. SQLite3.
- **refdba** - klient, nebo spíše systémový nástroj pro správu nastavení servru. Umožnuje vytvářet, mazat databáze, nastavovat jim přístupová práva nebo prohlížet různá typy statistik nastavení.
- **refdbc** - klient sloužící k přidávání, editování, mazání nebo prohlížení jednotlivých referenčních položek.
- **refdbib** - nástroj pro tvorbu bibliografie.
- **perl skripty** - rozšíření pro komunikaci s jazykem perl, provádí hlavně transformace mezi různými formáty, **med2ris**, **bib2ris**, **marc2ris** apod.

RefDB funguje hlavně pro Linux (proto také nepřikládám žádný obrázek, jedná se prakticky o program ovládaný přes konzoli), ale je vytvořena i varianta podporující Windows přes emulátor Linuxu Cygwin.

6.2 BibteXML Converter

6.2.1 Proč BibteXML?

BibteXML (Obrázek 6.2) je velice důležitým nástrojem pro tvorbu bibliografie. A to zejména snahou spojit dva největší „bibliografické konkurenty“, kteří se na poli zpracování textu, alias bibliografických citací, nachází - BibTEXa DocBook. Každý vytváří jinou hiearchchii zápisu. Je tedy problém vybrat citace od jednoho nástroje a předávat ho druhému. Přesně k tomuto směřuje program BibteXML [1].

6.2.2 Jak BibteXML funguje?

Program jednoduše převádí citace z bibtexovského formátu do XML, který je ve formě DocBooku. Vše funguje samozřejmě i naopak. Nová verze tohoto programu umí ještě více. Samozřejmostí je překlad i do jiných formátů. Konkretně se jedná o RIS, MODS XML, MARC, Endnote, které můžeme například využít v RefDB. Umí také konvertovat soubory `.bib`, či `.xml` přímo do bibliografie latexu. Připravit je pro překlad pomocí DocBook verze 4.5 nebo 5 nebo vytvořit formátované výstupy v HTML. Nakonec dochází k validaci všech dokumentů. V celém programu je možné měnit kódování jednotlivých výstupních dokumentů. BibteXML dokáže komunikovat s programem JabRef. Tedy umí převzít kódování, ve kterém je dokument uložen, a dále s ním pracovat. S programem je možné pracovat na platformách Linux i Windows.

6.3 JabRef

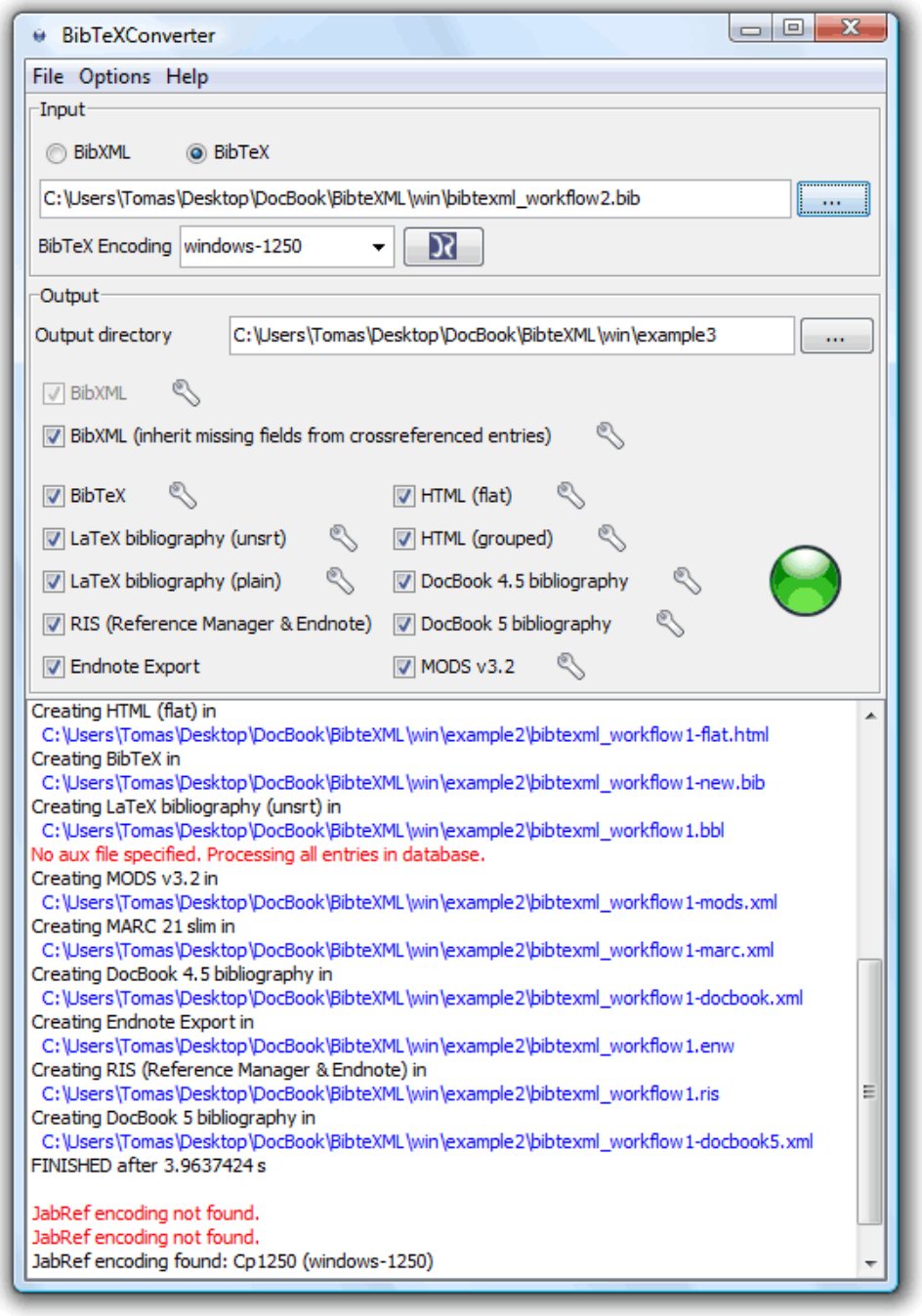
Program JabRef (Obrázek 6.3) [2] byl speciálně vytvořen pro snadnou editaci bibtexovských databází. Jedná se o grafický editor, kde vidíte načtené citace přehledně v tabulkách a za „běhu“ lze tyto tabulky měnit a manipulovat s nimi. JabRef poskytuje řadící a vyhledávací funkcionalitu pro lepší orientaci v obsahu vaší databáze citací. Nejpodstatnější funkcí je přidávání vlastních citací bez toho, abychom znali, které položky jsou potřebné, které volitelné, nebo jaký vložit identifikátor dané citace. Takové identifikátory mohou být automaticky generovány. Program je vytvořen v javě, je tedy nezávislý na platformě a může být spuštěn na jakémkoli systému podporujícím javu.

6.4 JReferences

Tento nástroj [21] byl vytvořen na podobném principu, jako je RefDB. Prakticky má tu samou funkcionalitu, až na malou drobnost. Ke svému chodu nepotřebuje externí databázový systém. Může citace zpracovávat buď ze souboru, nebo z databáze MySQL. V současné době není program aktivně vyvíjen, poslední modifikace byla někdy v roce 2002. Je určen převážně pro systém Linux (zase tedy nepřikládám obrázek, protože ovládání je přes konzoli), ale je možné ho přeložit i na Windows (přes Cygwin). Program je napsán pro javu a je nutno provést před spuštěním manuální nastavení cest, což je celkově dost nepříjemné a krkolomné pro koncového uživatele.

6.5 Zhodnocení nástrojů

Všechny zmíněné nástroje jsou prakticky více než potřebné pro lidi, kteří s bibliografickými citacemi nějakým způsobem manipulují. Bibliografické citace jsou obyčejné informativní texty. Tyto texty je potřeba zviditelnit tak, aby se s nimi lépe pracovalo a aby se umožnilo například pozdějšímu dohledání dané publikace, což zmíněné nástroje umožnily. Jejich převody mezi formáty nebo zobrazování v určité čitelné formě je asi jedinou možností, co s nástroji, resp. s bibliografickými citacemi jako texty, dělat.



Obrázek 6.2: BibtexXML Converter

Obrázek 6.3: JabRef

Kapitola 7

Závěr

Cílem této práce bylo popsat systém DocBook a jeho nástroje a navrhnout a implementovat šablonu pro formátování bibliografických citací. Formátovací šablona je přenosná mezi jednotlivými operačními systémy. Odzkoušeno na Ubuntu 8.04 a Windows Vista. Většina aplikací je vytvořena v javě, a pokud některé nejsou, tak jsou zaručeně implementovány, jak pro Linux, tak pro Windows. Existují ovšem i nástroje pro práci s DocBookem, vytvořené jenom pro systém Linux, jako například jReferences. Ale takové nástroje po bližším prozkoumání zastavily svůj vývoj před několika lety.

LNCS styl v XSL formátuje bibliografii přesně podle šablony vydané volně nakladatelstvím Springer, autorem LNCS. Jak už jsem popsal v implementaci, šablona byla vytvořena speciálně pro zahrnutí do DocBook XSL standardu. Druhá varianta šablony, která měla sloužit jako jednorázový překlad bez modifikace jiných souborů, se už nepodařila z časových důvodů dokončit.

Samotné programování mi přineslo hodně zkušeností s XSLT, jazykem XPATH a práci s XML, protože až do implementace šablony mi například XSLT a XPATH vůbec nic neříkalo. A o to víc mě seznámení s XSL lákalo a nakonec i hodně bavilo. Cíle této práce bylo tedy dosaženo, i když ne tak jednoduše, jak se mi na první pohled zdálo, protože celková studie a tvorba byla náročná z různých hledisek.

Studium tvorby dokumentace jiným způsobem, než například pomocí LaTeXu nebo MS Office, či OpenOffice, bylo velice zajímavé. Ačkoli je DocBook už nějaký ten pátek na světě, stále se o něm ví málo a jeho vývoj také nejde kdovíjak dopředu. Dle mého názoru je to způsobeno tím, že už na světě existují dokumentační nástroje, které jsou jednodušší, jak nastavením, tak ovládáním.

To, že práce nešla zrovna od ruky, bylo způsobeno tím, že systém DocBook je velice „rozkouskovaný“ a nejednotný, i když existuje prostředí `xslt2` od Jana Pavloviče. Je o něm velice málo vědět a na jeho vývoji se prakticky moc autorů, včetně pana Pavloviče, nepodílí (poslední revize byla 6. 6. 2006). To je dle mého úsudku škoda. Protože tento balík by mohl mít v budoucnu místo jako nástroj pro tvorbu bakalářských, diplomových, disertačních a jiných prací v rámci Fakulty Informačních Technologií.

Literatura

[1] BibTeX as XML markup.
URL http://bibtexml.sourceforge.net/

[2] JabRef: Reference Manager.
URL http://jabref.sourceforge.net/

[3] Lecture Notes In Computer Science.
URL http://www.springer.com/computer/lncs?SGWID=0-164-7-72376-0

[4] Berglund, A.: *Extensible Stylesheet Language (XSL) Version 1.1 W3C Working Draft*. W3C Prosinec 2006.
URL http://www.w3.org/TR/xsl

[5] Bob Stayton: *DocBook XSL: The Complete Guide*. Sagehill Enterprises, Čtvrté vydání, Září 2007.
URL http://www.sagehill.net/docbookxsl/

[6] Boldiš, P.: *Bibliografické citace dokumentů podle ČSN ISO 690 a ČSN ISO 690-2: Část 2 Modely a příklady citací u jednotlivých typů dokumentů*. Verze 3.0 (2004). ©1999 - 2004, poslední aktualizace 11. 11. 2004.
URL http://www.boldis.cz/citace/

[7] Boldiš, P.: *Bibliografické citace dokumentů podle ČSN ISO 690 a ČSN ISO 690-2: Část 1 Citace: metodika a obecná pravidla*. Verze 3.3. ©1999 - 2004, poslední aktualizace 11.11. 2004.
URL http://www.boldis.cz/citace/

[8] Hoenicka, M.: RefDB Documentation.
URL http://refdb.sourceforge.net/doc.html

[9] Kosek, J.: *XML: eXtensible Markup Language* [Online]. 1999.
URL http://www.kosek.cz/clanky/xml/xml-uvod.html

[10] Kosek, J.: *XML pro každého: podrobný průvodce*. Grada Publishing, 2000, ISBN 80-7169-860-1.

[11] Kosek, J.: *DocBook: Kapitola 9. Instalace* [Online]. 2007.
URL http://www.kosek.cz/xml/db/inst.html

[12] Kosek, J.: *DocBook: Stručný úvod do tvorby dokumentů* [Online]. 2007.
URL http://www.kosek.cz/xml/db/

[13] Norman Walsh a Leonard Muellner: *DocBook: The Definitive Guide*. O'Reilly, Říjen 2006, ISBN 156592-580-7.
URL http://www.docbook.org/tdg/en/html/docbook.html

[14] Pavlovič, J.: *Návod k modulu xslt2* [Online]. Červen 2006.
URL http://www.fi.muni.cz/~xpavlov/xml/

[15] Wikipedia: *Document Type Definition (CZ)* [Online]. Březen 2009.
URL http://cs.wikipedia.org/wiki/Document_Type_Definition

[16] Wikipedia: *Document Type Definition (EN)* [Online]. Květen 2009.
URL http://en.wikipedia.org/wiki/Document_Type_Definition

[17] Wikipedia: *Markup Language* [Online]. Květen 2009.
URL http://en.wikipedia.org/wiki/Markup_language

[18] Wikipedia: *RefDB* [Online]. Duben 2009.
URL http://en.wikipedia.org/wiki/Refdb

[19] Wikipedia: *SGML* [Online]. Duben 2009.
URL http://en.wikipedia.org/wiki/SGML

[20] Wikipedia: *XML* [Online]. Květen 2009.
URL http://en.wikipedia.org/wiki/Xml

[21] Willighagen, E. L.: JReferences.
URL http://jreferences.sourceforge.net/

Dodatek A

Obsah CD

```
LNCS template/docbook-xsl-1.74.0/
                                common/en.txt
                                fo/biblio.xsl
                                  /biblio-lncs.xsl
                                  /docbook.xsl
Examples/
        /2html/
              /html_html.html
              /html_xml.xml
              /html_xsl.xsl
        /fo2pdf/
               /fo_fo.fo
               /fo_pdf.pdf
               /fo_xml.xml
               /fo_xsl.xsl
        /deklaration.xml
        /grammar.dtd
Run LNCS with xslt2/
                    /fop/
                        /Makefile
                        /myxsl.xsl
                        /workflow-lncs.xml
                        /fop-lncs.pdf
                    /xep
                        /Makefile
                        /myxsl.xsl
                        /workflow-lncs.xml
                        /xep-lncs.pdf
                    /workflow-lncs.xml
Run LNCS with xsltproc/
                   /fop/
                        /Makefile
                        /myxsl.xsl
                        /workflow-lncs.xml
                        /fop-lncs.pdf
```

```
                    /xep
                        /Makefile
                        /myxsl.xsl
                        /workflow-lncs.xml
                        /xep-lncs.pdf
                    /workflow-lncs.xml
Utilities/
          /xslt2-2.6.5 (win32-linux)/
                                    /docbook-xsl.zip
                                    /Settings XEP FOP for XSLT2.zip
                                    /xslt2-2.6.5-install.jar
          /xsltproc (win32)/
                           /iconv-1.9.2.win32.zip
                           /libxml2-2.7.2+.win32.zip
                           /libxmlsec-1.2.11+.win32.zip
                           /libxslt-1.1.24.win32.zip
          /XEP 4.14 (win32-linux)/
                                  /setup-4.14-20081212-personal.jar
                                  /readme.txt
          /FOP 0.95 (win32-linux)/
                                  /fop-0.95-bin.zip
          /docbook-xsl-1.74.0.zip
INSTALL.txt
settings.txt
BP.zip
bp.pdf
```

Dodatek B

Instalace stylu a programů

Instalace LNCS stylu není těžká. V kořenové složce na instalačním CD je už v souboru `install.txt` napsán návod, jak styl nainstalovat a zprovoznit, ale ještě ho zde popíši. U prostředí XSLT2 ukáži, jak jej lze nainstalovat jak na Windows (Windows XP, Vista), tak na Linuxu (Ubuntu 8.04). U instalace samostatných komponent pro překlad jsem si vybral operační systém Linux (instalace a zprovoznění je zde mnohem jednodušší).

B.1 Docbook-xsl-1.74.0

Vše záleží na tom, jakou verzi DocBook XSL vlastníte. Styl byl vytvořen pro verzi 1.74.0. Ve složce `LNCS template/docbook-xsl-1.74.0` jsou obaženy složky se stylem a konfiguračními soubory. Složky lze rovnou překopírovat do složky s vašimi docbook-xsl styly.

Pokud tedy pracujete na Linuxu, zřejmě máte naistalovány, nebo nainstalujete xsl styly z balíčků. Přepněte se jako superuser pomocí příkazu `su`. Poté do míst, kde jsou balíčky nainstalovány, nakopírujte ze složky `LNCS template` její obsah. Stáhnutý balíček s docbook-xsl má cestu s největší pravěpodobností takovou:

`/usr/share/xml/docbook/stylesheet/nwalsh/`

Na Windows najděte složku, ve které máte xsl styly a do ní překopírujte obsah složky docbook-xsl-1.74.0.

Jestliže nemáte naistalovány žádné styly, na instalačním CD je taktéž přiložena celá verze xsl stylů 1.74.0, která je obsažena ve kořenové složce v `docbook-xsl-1.74.0`. V této složce je i zahrnut LNCS styl. Je tedy možné vzít celou složku a zkopírovat ji do míst, kde potřebujete xsl styly.

B.2 Jiné verze docbook-xsl

Víceméně by nemělo kolidovat, pokud budete mít nižší, či vyšší verzi stylů. Pokud by se tak přeci jen stalo, například proto, že se při přechodu na vyšší verzi do těchto souborů zasahovalo, tak v souboru `settings.txt` na CD je popsán postup, které soubory poměnit a přidat tak, abyste neměli žádné problémy. Aktuální verzi stylů najdete na adrese http://sourceforge.net/projects/docbook/.

B.3 Instalace xslt2

Balíček `xslt2-2.6.5` [14] od Jana Pavloviče je velice užitečný nástroj napsán v jazyce `Perl`, který slouží pro tvorbu dokumentů v DocBooku. Jeho instalace je velice snadná. Ve vašem počítači potřebujete mít nainstalováno prostředí Java Runtime Environment (`http://www.java.com/en/download/manual.jsp`), systém LaTeX(podporovaná je například distribuce Texlive dostupná na `http://ftp.cvut.cz/tex-archive/systems/texlive/Images/` a některou verzi jazyka Perl (`http://www.perl.org/get.html`). Následně stačí být ve složce, kde máte uložený instalační balíček xslt2 a napsat z příkazové řádky:

```
java -jar xslt2-2.6.5-install.jar
```

Zobrazí se vám průvodce instalací a vy vše nainstalujte.

Po instalaci je potřeba nastavit systémové cesty. Uvedené cesty jsou příkladné a mohou se měnit v závislosti na tom, kam jste si co zkopírovali a nainstalovali.

- V Linuxu - pokud neměníte instalační cesty, tak se instalace xslt2 automaticky volí do složky `opt`, stačí napsat `export PATH=/opt/xslt2-2.6.5/run:$PATH`
- Ve Windows - jděte do `Ovládací panely/Systém/Upřesnit nastavení systému/proměnné prostředí` a přidejte do `Path` cestu, kde máte nainstalovaný xslt2, například `c:/xslt2 složka/run`.

Pokud máte nainstalován Texlive v Linuxu, při instalaci xslt2 se vám automaticky vyhledá složka s jeho umístěním, ale ve Windows si složku s LaTeXem musíte najít sami. Je to zřejmě menší chyba, kterou autor nezamýšlel.

B.4 Instalace FOP a XEP

B.4.1 FOP

FOP je program zadarmo vyvíjený firmou Apache. Na CD je program ve verzi 0.95. Přiložil jsem zde binární verzi pro Windows i Linux. Pro Linux lze také FOP stáhnout ve formě balíčku (v Ubuntu 8.04 se v repositářích nachází verze 0.94). Pro zprovoznění stačí nakopírovat obsah adresáře s FOPem kdekoli na disk a nastavit systémovou cestu právě na tento adresář.

B.4.2 FOP pro xslt2

Xslt2 má už při své instalaci nainstalovanou podporu pro FOP, ale není náležitě nastaven. Důležité je mít správně nastavené cesty. Ve složce `Utilities/xslt2-2.6.5/Settings XEP FOP for XSLT2` je složka `fop`. Tu překopírujte do adresáře `soft` ve složce s nainstalovaným xslt2. Dále musíme nastavit přístup k typům písem. Ve složce `fop/config` je soubor `userconfig.xml` a nastavte tyto tři položky podle svého umístění:

- Pro Windows:

```
<!ENTITY fonts.dir ,,c:\windows\fonts\‘‘>
<!ENTITY metrics.dir ,,cesta k˜xslt2-2.6.5\soft\fop\fonts‘‘>
<!ENTITY hyph.dir ,,cesta k˜xslt2-2.6.5\soft\fop\hyph‘‘>
```

- Pro Linux:

```
<!ENTITY fonts.dir ,,/usr/share/fonts/truetype/‘‘>
<!ENTITY metrics.dir ,,/opt/xslt2-2.6.5/soft/fop/fonts/‘‘>
<!ENTITY hyph.dir ,,/opt/xslt2-2.6.5/soft/fop/hyph‘‘>
```

Pokud nemáte na Linuxu ve složce `truetype` potřebné fonty, ve složce s FOPem je adresář s fonty. Můžete si tyto fonty následně zkopírovat nebo nastavit přímo cestu k těmto fontům.

B.4.3 XEP

XEP je komerční placený program a k jeho používání zadarmo musíte mít licenci, kterou obdržíte na stránkách výrobce (http://www.renderx.com/download/index.html). Tato licence vám zajistí bezproblémový překlad s tím rozdílem, že ve spodu každé stránky se vám zobrazí malé logo firmy. S licencí se tedy jedná víceméně o verzi „personal“, umožňující využívat program zadarmo. Na CD jsem přiložil program ve verzi 4.14. Pro instalaci musíte mít nainstalovanou javu. Vlastní instalaci pak spustíte příkazem:

```
java -jar setup-4.14-20081212-personal.jar
```

Abyste mohli XEP náležitě používat, je nutné zase nastavit systémové cesty, které budou obdobné jako při instalaci prostředí xslt2.

B.4.4 XEP pro xslt2

Z CD v adresáři `Utilities/xslt2-2.6.5/Settings XEP FOP for XSLT2` stačí překopírovat složku `Tools` a `xep` do `cesta k xslt2-2.6.5/soft`. Protože se jedná o komerční program, potřebujete si zajistit zdarma licenci na stránkách výrobce.

B.5 Instalace xsltproc

- Windows - na CD je ve složce `Utilities/xsltproc` program `xsltproc`. Program je binární distribucí a je určen jen na Windows, Kdekoli na disku jej můžete rozbalit. Poté stačí nastavit systémovou cestu na toto umístění a můžete jej normálně používat pomocí příkazů, které jsem napsal výše (více o instalaci pro Windows je na [11]).
- Linux - zde je situace o něco jednodušší. Xsltproc by měl být už předinstalovaný v systému. Ovšem pokud není, zaručeně pro něj existuje balíček, který se dá stáhnout z repositářů. Na Ubuntu 8.04 je balíček v repositářích. Stačí stáhnout a nainstalovat. Systémové cesty už jsou nastaveny automaticky.

B.6 Spuštění stylu

Na CD jsou dvě složky:

- `Run LNCS with xslt2` - spuštění přes prostředí xslt2. Uvnitř se nachází složky FOP a XEP. Každá z nich má svůj Makefile. Pokud tedy máte jeden z překladačů FOP nebo XEP, můžete LNCS styl přeložit přes ně. Makefile provádí následující překlad:

```
xslt -t foppdf -s myxsl.xsl workflow-lncs.xml
```

- `Run LNCS with xsltproc` - spuštění pomocí přímých nástrojů pro překlad, jako je `xsltproc` do formátu `.fo` a FOP, nebo XEP do formátu `.pdf`. Makefile tedy provádí:

  ```
  xsltproc --nonet -o lncs-fop.fo myxsl.xsl workflow-lncs.xml
  fop lncs-fop.fo lncs-fop.pdf
  ```

Samozřejmě, jak už jsem psal, vše záleží na tom, jak máte nastavené systémové cesty. Pokud je nebudete mít nastaveny, musíte zavolat jednotlivé příkazy (xslt2, xsltproc, fop, xep) tak, že napíšete na příkazovou řádku celou cestu od kořenového adresáře až k umístění překladačů. Mohlo by to vypadat asi takto:

```
c:/xslt2-2.6.5/run/xslt -t foppdf -s myxsl.xsl workflow-lncs.xml
c:/docbook/xsltproc/xsltproc --nonet -o lncs-fop.fo myxsl.xsl workflow-lncs.xml
c:/docbook/fop/fop lncs-fop.fo lncs-fop.pdf
```

Na Linuxu by byl přístup obdobný.