VYSOKÁ ŠKOLA EKONOMICKÁ V PRAZE

Fakulta informatiky a statistiky
Katedra informačního a znalostního inženýrství

Diplomová práce

Převod formátovacích objektů do formátu WordML

Autor: Jiří Pachman

Vedoucí práce: Ing. Jiří Kosek

únor 2004 – srpen 2004

Anotace

Diplomová práce zkoumá možnosti převodu formátovacích objektů do formátu WordML. Jelikož jsou oba formáty založeny na XML, tak v úvodu práce píši právě o XML. V další části práce popisuji formátovací objekty, formát WordML a porovnávám je. V další části píši o možnostech implementace převodu formátovacích objektů do formátu WordML a v poslední části se snažím tento převod uskutečnit s využitím XSLT.

Poděkování

Rád bych poděkoval Ing. Jiřímu Koskovi za vedení a cenné rady při tvorbě diplomové práce.

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a použil pouze literaturu uvedenou v přiloženém seznamu. Nemám námitek proti půjčení práce se souhlasem katedry ani proti zveřejnění práce nebo její části.

V Kladně dne 5.srpna 2004

Jiří Pachman

Obsah

1 Úvod

V této práci zkoumám možnosti převodu formátovacích objektů do formátu WordML. Zjednodušeně řečeno, formátovací objekty umožňují popsat obsah a strukturu dokumentu. Jsou součástí XSL (Extensible Stylesheet Languague), který umožňuje transformovat jeden XML dokument na jiný. Naproti tomu formát WordML je formát založený na XML, který umí číst textový editor Microsoft Word 2003.

Důvodem převodu formátovacích objektů do formátu WordML je, že k převodu formátovacích objektů na dokument (např. ve formátu PDF) je potřeba procesor formátovacích objektů, který nemusí být dostupný všem uživatelům. Hodně uživatelů naopak používá Microsoft Word a tak by jistě převod formátovacích objektů do Wordu přivítali.

Jelikož jsou tedy oba formáty založeny na XML, tak hned v následující kapitole nastíním základní rysy tohoto jazyka. V dalších třech kapitolách vás postupně seznámím s formátovacími objekty, s formátem WordML a s jejich společnými a rozdílnými rysy. V další kapitole potom zkoumám možnosti implementace převodu formátovacích objektů do formátu WordML. Nakonec následuje ukázka převodu části formátovacích objektů do formátu WordML pomocí jazyka XSLT.

2 XML

XML [10] znamená Extensible markup languague (rozšiřitelný značkovací jazyk) a umožňuje nám vytvářet strukturované dokumenty velmi flexibilním způsobem. Podobá se jazyku HTML, jelikož oba jazyky jsou odvozeny od SGML (Standard Generalized Markup Languague). Narozdíl od HTML se však XML pokouší oddělit obsah dokumentu od jeho reprezentace.

Jak tedy vypadá struktura XML dokumentu? Podobně jako HTML dokument se XML dokument skládá ze značek a ostatních dat. Tyto značky si uživatel sám nadefinuje ve schématu (XML schéma [3], DTD, Relax NG), značky říkají co za data se v nich vyskytuje, nikoliv jak se mají zobrazit. Zobrazení značek se dá dále nadefinovat ve stylu k danému XML dokumentu v jazyce XSL (Extensible Style languague) [1,6] nebo pomocí kaskádových stylů [2].

Značky tedy dávají dokumentu strukturu, zatímco ostatní data představují jeho obsah. Dokument XML je správně strukturovaný, jestliže splňuje následující požadavky: pro každý element existuje počáteční i koncová značka, dokument XML obsahuje kořenový element dokumentu, v němž jsou všechny ostatní značky vloženy a počáteční a koncové značky vložených elementů se nesmějí překrývat. Příklad 1 ukazuje správně strukturovaný dokument XML.

Příklad 1

```
<?xml version="1.0" encoding="windows-1250"?>
<!DOCTYPE adresar SYSTEM "adresar.dtd">
<?xml-stylesheet type="text/xsl" href="adresar.xsl"?>
<adresar>
 <firma>
  <ICO>4758691125</ICO>
  <nazev>Energie - stavební a báňská a.s.</nazev>
  <sidlo>
   <ulice>Vašíčkova 3801</ulice>
   <PSC>27204</PSC>
   <mesto>Kladno</mesto>
  </sidlo>
 </firma>
</adresar>
```

Podívejme se na tento příklad trochu podrobněji. První řádek nám říká, že se jedná o dokument XML verze 1 a je použito kódování windows-1250. Druhý řádek odkazuje na soubor, ve kterém je uloženo schéma tohoto dokumentu. Třetí řádek říká, že k tomuto dokumentu existuje styl v souboru adresar.xsl, který říká, jak se má daný soubor zobrazit. Následuje kořenový element dokumentu a dále další vnořené elementy. Dokument končí koncovou značkou kořenového elementu.

Jestliže dokument XML splňuje řečená pravidla, pak je správně strukturovaný. Další omezení lze obsahu dokumentu XML určit pomocí některého schématu. Schéma obsahuje informace o struktuře dokumentu. Definuje elementy, které lze v dokumentu použít, říká, které elementy mohou obsahovat další elementy, jaký je počet a pořadí elementů, stanovuje atributy, které mohou elementy mít a volitelně také určuje hodnoty dostupných atributů. Toto všechno dovoluje nadefinovat DTD. XML

schémata navíc dovolují nadefinovat datový typ obsahu elementu. Dokument XML je tedy platný, jestliže je správně strukturovaný a zároveň splňuje požadavky schématu.

Jak už jsem uvedl výše, způsob zobrazení XML dokumentu závisí na definici stylu pro jeho zobrazení. Nejlépe se pro napsání takového stylu hodí jazyk XSL. Tento jazyk vytvořilo konsorcium W3C (World Wide Web Consortium). Podstatou tohoto jazyka je přesné určení formátu dokumentu. XSL se skládá z XSLT (XSL Transformation) [6] a z XSL-FO (formátovací objekty) [5]. XSLT umožňuje převádět jeden dokument XML na jiný dokument XML (např. na HTML správně strukturované jako XML). XSL-FO definuje značky, které říkají jak má být dokument na výstupu přesně zobrazen a slouží k převodu do formátů jako je PDF či PostScript. K transformaci XML dokumentu pomocí XSLT stylu je potřeba XSLT procesor (např. Saxon, Xalan). Tento procesor převede tedy XML dokument na jiný XML dokument, například i na dokument obsahující formátovací objekty. K převodu tohoto dokumentu do zobrazitelného formátu (např. PDF) je pak zapotřebí FO procesor. Žádný z těchto procesorů však zatím neumí interpretovat všechny formátovací objekty, jelikož ty jsou velmi složité. Jako příklad FO procesorů mohu uvést XEP a FOP.

K čemu je XML vlastně dobré? Za prvé se díky jeho strukturovanosti rozšiřují možnosti vyhledávání dokumentů. Za druhé, jestliže potřebujeme jeden dokument zároveň zpřístupnit na Internetu a zároveň ho například vytisknout, nemusíme ho psát dvakrát (jednou v HTML a jednou ve Wordu), nýbrž stačí mít jeden XML dokument a k němu dva styly (jeden pro převod do HTML a druhý pro převod do tiskové podoby). A za třetí XML slouží jako universální formát výměny dat.

3 Formátovací objekty

Jak už bylo řečeno výše, formátovací objekty (XSL-FO) jsou druhou částí specifikace XSL, která byla vytvořena konsorciem W3C [1]. XSL-FO dokument je XML dokument, který popisuje, jak má dokument vypadat na výstupu.

Formátovací model XSL-FO je založen na pravoúhelných čtyřúhelnících, říkejme jim oblasti. Každá oblast může obsahovat text, prázdný prostor, obrázek nebo jiné formátovací objekty. Každá takováto oblast má svoje hranice a vnitřní i vnější okraje na všech čtyřech stranách. Když je potom dokument obsahující formátovací objekty zpracován formátovacím programem, tak je dokument rozdělen na stránky. Každá stránka se pak skládá z několika oblastí.

V dalších částech této kapitoly popíšu strukturu dokumentu XSL-FO a způsob použití jednotlivých formátovacích objektů.

3.1 Struktura dokumentu XSL-FO

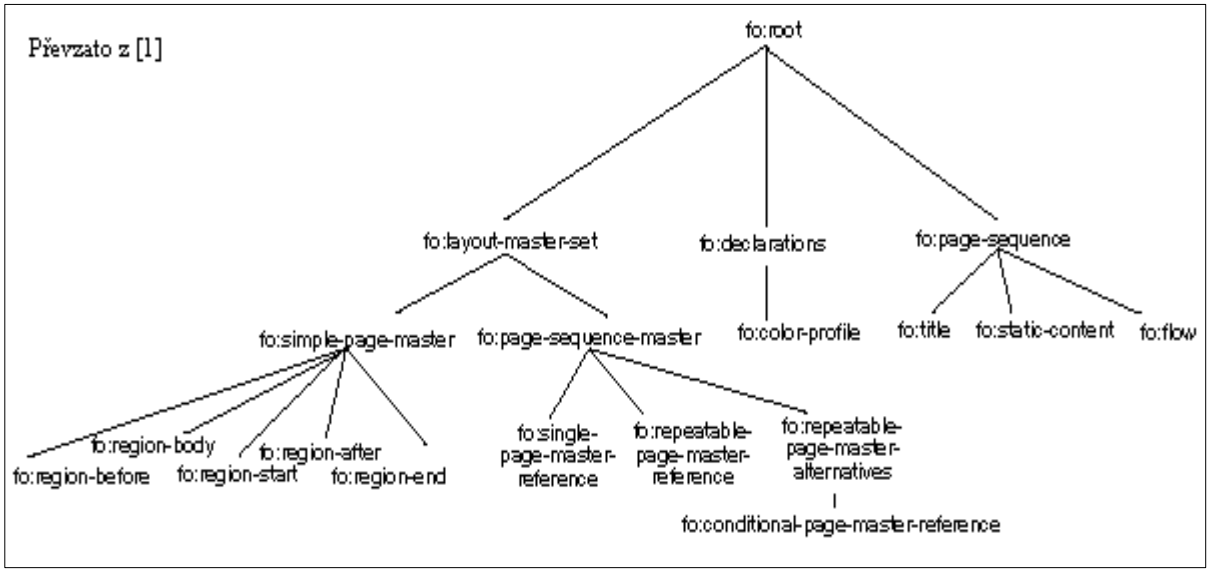
Celý dokument je obalen elementem *fo:root*. Tento element pak může mít tři potomky a to *fo:layout-master-set*, *fo:declarations* a *fo:page-sequence*. Základní strukturu XSL-FO dokumentu ukazuje názorně obrázek 1.

Element *fo:layout-master-set* obsahuje ve svých potomcích *fo:simple-page-master* jednu nebo více předloh, které říkají, jaké je rozvržení stránky nebo stránek. Dále může obsahovat element *fo:page-sequence-master*, který obsahuje další elementy, které pomocí odkazů na *fo:simple-page-master* umožňují definovat sekvence stránek v dokumentu.

Element *fo:page-sequence* obsahuje elementy *fo:title* (nepovinný), *fo:static-content* a *fo:flow*, které již obsahují samotný obsah stránek. Tyto stránky jsou rozvržené podle odkazu na *fo:page-sequence-master* nebo na *fo:simple-page-master*.

Element *fo:declarations* se používá pro seskupení globálních deklarací stylu. Může obsahovat potomky z jiného jmenného prostoru než XSL a jeden nebo více elementů *fo:color-profile*, které definují barevný profil ICC. Barevný profil je identifikován pomocí atributu *src*, který obsahuje URI (Uniform Resource Identifier) barevného profilu a v dokumentu je na něj dále odkazováno jeho jménem, které je zadáno v atributu *color-profile-name*.

Obrázek 1



3.2 Rozvržení stránky

Jak už jsem uvedl výše, rozvržení stránky definuje element *fo:simple-page-master*. Stránka se skládá z okrajů, do kterých nikdy nezasahuje text a dále z těla stránky. Tělo stránky má pak pět částí, které jsou definovány dalšími elementy, jak je vidět z obrázku 1.

Příklad 2 ukazuje, jak může vypadat popis rozvržení stránky a obrázek 2 ukazuje, jak bude vypadat stránka takto rozvržená. Co tedy znamenají použité atributy? Atribut *master-name* je pojmenování rozvržení stránky, na které se odvoláváme v atributu *master-reference* elementu *fo:page-sequence*. Atributy *page-width* a *page-height* určují šířku a výšku stránky a atributy *margin-top*, *margin-bottom*, *margin-left* a *margin-right* elementu *fo:simple-page-master* určují velikost horního, dolního, levého a pravého okraje stránky, do kterých už nezasahuje text. Atribut *extent* určuje výšku regionů before a after nebo šířku regionů start a end. Atributy *margin-top* a *margin-bottom* elementu *fo:region-body* říkají, jak daleko od okrajů stránky začíná normální text těla dokumentu. Z toho vyplývá, že region body překrývá ostatní regiony (start, end, before, after) a aby se nepřekrývaly i textem je třeba nastavit atributy *margin* elementu *fo:region-body* větší nebo rovny atributu *extent* ostatních regionů.

Příklad 2

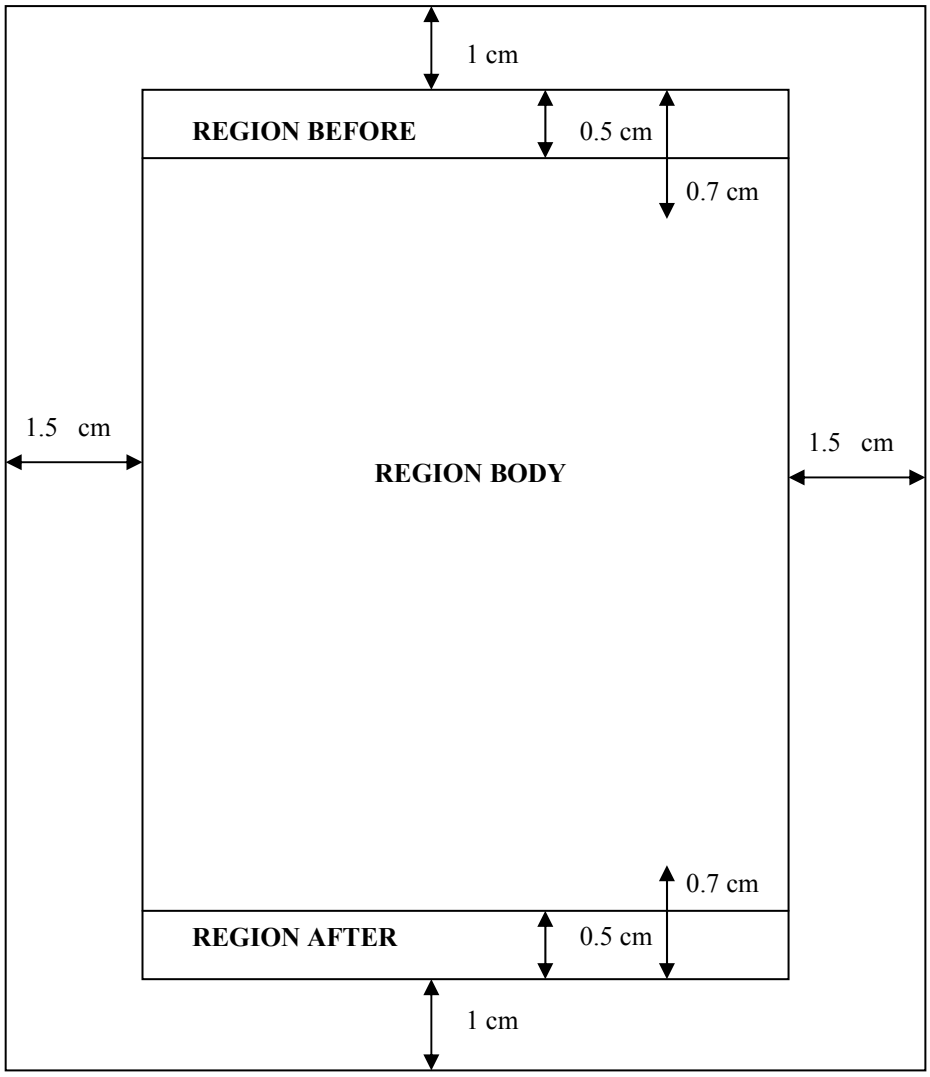
```
<fo:layout-master-set>
  <fo:simple-page-master master-name="A4"
    page-width="210mm"  page-height="297mm"
    margin-top="1cm"  margin-bottom="1cm"
    margin-left="1.5cm" margin-right="1.5cm">
   <fo:region-body margin-top="0.7cm"
                margin-bottom="0.7cm"/>
   <fo:region-before extent="0.5cm"/>
   <fo:region-after  extent="0.5cm"/>
  </fo:simple-page-master>
</fo:layout-master-set>
```

Element *fo:simple-page-master* může ještě obsahovat atributy *writing-mode* a *reference-orientation*. Atributem *writing-mode* můžeme nastavit směr toku textu. Atribut může nabývat hodnot lr-tb (zleva-doprava, shora-dolů), rl-tb (zprava-doleva, shora dolů) nebo tb-rl (text v řádce shora-dolů, řádky zprava-doleva). Atribut *reference-orientation* udává ve stupních rotaci dané oblasti vzhledem k oblasti, která je jejím rodičem.

V elementu *fo:region-body* můžeme ještě použít atributy *column-count* a *column-gap*, ve kterých můžete určit počet sloupců a velikost mezery mezi sloupci.

Ve všech elementech regionů pak můžeme použít ještě atributy týkající se ohraničení, pozadí a vnitřních okrajů, které popíšu později. Dále můžeme použít atributy *writing-mode* a *reference-orientation* popsané výše a nakonec atribut *display-align*, který určuje zarovnání vzhledem k rodičovské oblasti (můžete použít hodnoty before, center, after).

Obrázek 2



3.3 Stanovení sekvence stránek

Rozvržení stránky můžeme pomocí elementu *fo:simple-page-master* stanovit vícekrát. Pořadí a omezení použití těchto rozvržení potom můžeme stanovit pomocí elementu *fo:page-sequence-master*. Tento element pojmenujeme pomocí atributu *master-name*, na který se pak odvoláváme v atributu *master-reference* elementu *fo:page-sequence*. Element se skládá z jednoho nebo více elementů *fo:single-page-master-reference*, *fo:repeatable-page-master-reference* nebo *fo:repeatable-page-master-alternatives*.

Element *fo:single-page-master-reference* specifikuje část sekvence stránek, kterou tvoří jediná stránka. V atributu *master-reference* se odkazujeme na rozvržení stránky, které má být použito.

Element *fo:repeatable-page-master-reference* specifikuje část sekvence stránek, kterou tvoří několik stránek stejného rozvržení. Na toto rozvržení se odkazujeme pomocí atributu *master-reference*, atribut *maximum-repeats* pak stanoví maximální počet opakování stránek s tímto rozvržením.

Element *fo:repeatable-page-master-alternatives* specifikuje část sekvence stránek, kterou tvoří několik stránek s různým rozvržením. Které rozvržení bude pro stránku použito se stanoví pomocí elementů *fo:conditional-page-master-reference* a jejich atributů *master-reference*, *page-position*, *odd-or-even*, *blank-or-not-blank*. Pravidlo je takové, že se použije rozvržení dané atributem *master-reference*, jestliže jsou splněny podmínky dané ostatními atributy.

Atribut *page-position* může nabývat hodnot first, last, rest, any s významem první stránka, poslední stránka, jiná stránka než první nebo poslední, jakákoliv stránka.

Atribut *odd-or-even* může nabývat hodnot odd, even, any s významem sudá stránka, lichá stránka, jakákoliv stránka.

A nakonec atribut *blank-or-not-blank* může nabývat hodnot blank, not-blank s významem prázdná stránka (neobsahuje žádné oblasti elementu *fo:flow*), neprázdná stránka.

3.4 Obsah stránek

Samotný obsah stránek je umístěn v potomcích elementu *fo:page-sequence*. Jaké bude rozvržení sekvence stránek určujeme atributem *master-reference* odkazujícím na nějaký *master-name*. Dále můžeme pomocí atributu *initial-page-number* určit počáteční číslo této sekvence stránek a to buď určením daného čísla nebo použitím jednoho z klíčových slov auto, auto-odd, auto-even, která říkají, začni číslem 1 nebo pokračuj z předchozích stránek a to číslem následujícím, následujícím sudým nebo následujícím lichým. Atributem *force-page-count* pak určujeme, zda je nutné, aby sekvence obsahovala lichý či sudý počet stránek nebo aby končila lichou či sudou stránkou a to hodnotami even, odd, end-on-even, end-on-odd. Další hodnotou tohoto atributu je hodnota auto, která poslední stránku udělá opačnou, než je první stránka (první sudá, poslední lichá a naopak). Poslední hodnotou je *no-force*, která znamená nechat počet stran takový jaký je. Ještě můžeme použít další atributy stanovující

formát číslování a to atributy format, *letter-value*, *country*, *languague*, *grouping-separator* a *grouping-size*.

Element *fo:page-sequence* může obsahovat elementy *fo:title*, *fo:static-content* a *fo:flow*. Element *fo:title* může obsahovat titulek sekvence stránek, který se může skládat z textu a dalších in-line elementů. Titulek může být použit uživatelským programem k identifikaci dokumentu.

Obsah elementu *fo:static-content* se opakuje na každé stránce dané sekvence stránek, pro kterou existuje daný region. Může tak obsahovat například čísla stránek. Atribut *flow-name* specifikuje jméno regionu, kde se má obsah zobrazit. Jména regionů se zadávají ve tvaru xsl-region-before (after, start, end). Potomky elementu mohou být všechny blokové elementy, což jsou *fo:block*, *fo:block-container*, *fo:table-and-caption*, *fo:table* a *fo:list-block*.

A nakonec element *fo:flow* může obsahovat všechny blokové elementy, jejichž obsah se rozdělí do stránek. Stejně jako v předchozím elementu musíme definovat v atributu *flow-name* jméno regionu.

3.5 Odstavce, nadpisy, titulky

Odstavce, nadpisy, titulky, názvy kapitol, popisky tabulek a další podobné součásti textu se dají vytvořit pomocí elementu *fo:block*. Použití tohoto elementu i s jeho nejčastějšími atributy ukazuje příklad 3. Příklad ukazuje vytvoření odstavce, který následuje 1 cm po předchozím a po jehož skončení končí i obsah dané stránky. Odstavec je psán tučným modrým písmem Arial velikosti 12. V odstavci je povoleno dělení slov, rozdělené slovo však musí mít alespoň 3 písmena na začátku nového řádku a alespoň 3 písmena na konci starého řádku. Rozdělená slova se však musí vyskytovat na stejné stránce a těsně za sebou se může opakovat nejvýše jeden řádek s děleným slovem. Za každý řádek je přidána mezera velikosti 6 bodů. Odstavec je zarovnán do bloku, jeho poslední řádek doleva, přičemž první řádek odstavce je odsazen o 0,5 cm. Odstavec může začínat na konci stránky nebo končit na začátku stránky pouze v případě, že se tam vejdou alespoň 3 řádky.

Příklad 3

```
<fo:block space-before="1cm" break-after="page" font-family="arial"
          font-size="12pt" font-weight="bold" color="blue"
          hyphenate="true" hyphenation-push-character-count="3"
          hyphenation-remain-character-count="3"
          hyphenation-keep="page" hyphenation-ladder-count="1"
          text-depth="6pt" text-align="justify"
          text-align-last="left" text-indent="0.5cm"
          orphans="3" widows="3">
Zde je text odstavce.
</fo:block>
```

Element *fo:block* může tedy obsahovat text, ale také další blokové elementy (například tabulky) nebo další in-line elementy (určující např. odlišně napsané slovo v odstavci). Kromě atributů prezentovaných v příkladu 3, může tento element obsahovat spoustu dalších atributů, z nichž některé budou popsány dále.

3.6 Seznamy

K vytváření seznamů se používá element *fo:list-block*. Každý seznam se skládá z několika položek, které jsou zde zastoupeny elementem *fo:list-item*. Každá položka v seznamu sestává z odrážky a z textu. Odrážka se vytvoří elementem *fo:list-item-label* a text elementem *fo:list-item-body*. Oba elementy mohou dále obsahovat blokové elementy.

Příklad 4 ukazuje vytvoření seznamu s 1 položkou. Odrážkou je pomlčka a je odsazena o 1 cm od okraje stránky, vzdálenost mezi začátkem odrážky a začátkem textu jsou 2 cm, vzdálenost mezi koncem odrážky a začátkem textu je 0,5 cm a text je odsazen od pravého okraje stránky o 1 cm. Odsazení odrážky od pravého okraje stránky se automaticky vypočítá pomocí funkce label-end(), což

Příklad 4

```
<fo:list-block provisional-distance-between-starts="2cm"
                provisional-label-separation="0.5cm">
  <fo:list-item>
    <fo:list-item-label start-indent="1cm" end-indent="label-end()">
      <fo:block>&#x2013;</fo:block>
    </fo:list-item-label>
    <fo:list-item-body start-indent="body-start()" end-indent="1cm">
      <fo:block>Zde je text první položky v seznamu</fo:block>
    </fo:list-item-body>
  </fo:list-item>
</fo:list-block>
```

je šířka celé oblasti mínus 2 cm mínus 1 cm plus 0,5 cm. Odsazení textu od levého okraje stránky se automaticky vypočítá pomocí funkce body-start(), což je 1 cm +2 cm.

3.7 Tabulky

K vytváření tabulek s popiskem se používá element *fo:table-and-caption*, k vytvoření tabulky bez popisku můžeme použít pouze element *fo:table*. Element *fo:table-and-caption* tedy obsahuje popisek v elementu *fo:table-caption* a vlastní tělo tabulky v elementu *fo:table*. Element *fo:table* může obsahovat další čtyři elementy. Prvním z nich je *fo:table-column*, který se používá pro definici vlastností sloupců tabulky jako je například jejich šířka. Druhým z nich je *fo:table-header*, který obsahuje hlavičku tabulky. Podobně element *fo:table-footer* obsahuje patičku tabulky. Element *fo:table-body* potom obsahuje všechny buňky tabulky mezi její hlavičkou a patičkou. Všechny tyto elementy kromě elementu *fo:table-column* se skládají ze řádků tabulky *fo:table-row* a řádky se skládají z jednotlivých buněk tabulky *fo:table-cell*, přičemž element *fo:table-row* může být vynechán, jestliže oddíl tabulky obsahuje pouze jeden řádek.

V příkladu 5 je ukázka jednoduché tabulky. Tabulka se skládá z popisku, který je umístěn nad tabulkou a zarovnán doleva a ze samotné tabulky, za kterou je vložen konec stránky. Samotná tabulka je 120 mm široká, text je v každé buňce vycentrován, přičemž celá tabulka je ohraničená černou souvislou čárou širokou půl bodu. Text v buňce je vzdálen od její hranice na všech stranách o 2 body.

Jestliže se tabulka nevejde na stránku, bude na každé stránce s kusem tabulky zobrazena i hlavička, patička však bude pouze na poslední stránce. Tabulka se skládá ze 2 sloupců širokých 60 mm. Tabulka dále obsahuje 4 řádky. První z nich tvoří hlavičku (s modrým pozadím), druhé dva tělo tabulky a poslední tvoří patičku tabulky (se žlutým pozadím), v níž jsou oba sloupce tabulky sloučeny dohromady.

Příklad 5

```
<fo:table-and-caption caption-side="top" break-after="page">
  <fo:table-caption>
    <fo:block text-align="left">Tabulka 1</fo:block>
  </fo:table-caption>
  <fo:table table-omit-header-at-break="false"
            table-omit-footer-at-break="true" width="120mm"
            text-align="center" border-style="solid"
            border-width="0.5pt"  border-color="black"
            border-collapse="separate" padding="2pt">
    <fo:table-column column-width="50%"
                     number-columns-repeated="2"/>
    <fo:table-header background-color="blue">
      <fo:table-cell><fo:block>Anglicky</fo:block></fo:table-cell>
      <fo:table-cell><fo:block>Česky</fo:block></fo:table-cell>
    </fo:table-header>
    <fo:table-footer background-color="yellow">
      <fo:table-cell number-columns-spanned="2">
        <fo:block>Autorem tabulky je ...</fo:block>
      </fo:table-cell>
    </fo:table-footer>
    <fo:table-body>
      <fo:table-row>
        <fo:table-cell><fo:block>black</fo:block></fo:table-cell>
        <fo:table-cell><fo:block>černá</fo:block></fo:table-cell>
      </fo:table-row>
      <fo:table-row>
        <fo:table-cell><fo:block>white</fo:block></fo:table-cell>
        <fo:table-cell><fo:block>bílá</fo:block></fo:table-cell>
      </fo:table-row>
    </fo:table-body>
  </fo:table>
</fo:table-and-caption>
```

3.8 Poznámky pod čarou

Poznámka pod čarou se vkládá do dokumentu pomocí elementu *fo:footnote*. Každá poznámka se skládá z odkazu na poznámku a z textu poznámky. Odkaz na poznámku vytvoříme pomocí elementu *fo:inline*, přičemž text tohoto elementu bude přidán do textu odstavce. Samotný text poznámky vytvoříte pomocí elementu *fo:footnote-body*, který může obsahovat jakýkoliv blokový element a jehož obsah se poté objeví v dolní části stránky (region-after).

Použití poznámky pod čarou ilustruje příklad 6. Poznámka se skládá z odkazu [1] v textu na stránce a dále z textu poznámky pod čarou v zápatí stránky.

Příklad 6

```
<fo:footnote>
  <fo:inline font-size=”smaller” vertical-align=”super”>
    1
  </fo:inline>
  <fo:footnote-body>
    <fo:block>
      <fo:inline font-size=”smaller” vertical-align=”super”>
        1
      </fo:inline>
      Zde je text poznámky pod čarou…
    </fo:block>
  </fo:footnote-body>
</fo:footnote>
```

3.9 Obrázky

Obrázek se dá do dokumentu vložit pomocí elementu *fo:external-graphic* nebo pomocí elementu *fo:instream-foreign-object*. Element *fo:external-graphics* se používá ke vkládání obrázků, jejichž data jsou uložena vně dokumentu (např. GIF nebo JPG), zatímco element *fo:instream-foreign-object* se používá ke vkládání obrázků, jejichž data jsou součástí dokumentu, nejčastěji jako podstrom se syntaxí XML (např. SVG).

Vložení obrázku ukazuje příklad 7. Do dokumentu je tak vložen obrázek ze souboru obrazek.gif, jehož typ je dán atributem *content-type*. Obrázek je vertikálně vycentrován, přičemž výška a šířka oblasti, do které je obrázek vložen, je dána jeho skutečnou výškou a šířkou. Nicméně je zadána jeho požadovaná výška i šířka 2 cm krát 2 cm, takže skutečné rozměry obrázku musí být upraveny. Hodnota uniform atributu *scaling* určuje, že musí být zachován poměr výšky obrázku ku jeho šířce. Poslední atribut říká, že úprava šířky a výšky obrázku se má odehrávat pouze v celých číslech.

Příklad 7

```
<fo:block>
  <fo:external-graphic content-type=”content-type:image/gif”
                       src=”url(‘obrazek.gif’)”
                       display-align=”center”
                       height=”auto” width=”auto”
                       content-height=”2cm” content-width=”2cm”
                       scaling=”uniform”
                       scaling-method=”integer-pixels”/>
</fo:block>
```

3.10 Vodící linky, citace stránek

Vodící linky jsou používané nejčastěji při generování obsahu dokumentu jako oddělovače názvu kapitoly a čísla stránky, na kterém se daná kapitola nachází. Dají se vytvořit pomocí elementu *fo:leader*. Citace čísla stránky se do dokumentu vloží pomocí elementu *fo:page-number-citation* a

jeho atributu *ref-id*, který se odkazuje na atribut *id* jakéhokoliv blokového nebo in-line elementu nacházejícího se v toku stránek.

Příklad 8 ukazuje použití vodících linek pro generování obsahu dokumentu a obrázek 3 ukazuje, jak to bude vypadat na výstupu. Element *fo:block*, který obsahuje vodící linku je zarovnán do bloku a nejprve obsahuje název firmy, poté vodící linku složenou z černých teček a odsazenou zleva i zprava o dva milimetry a nakonec číslo stránky, na které se o firmě píše.

Atribut *leader-pattern* může ještě nabývat hodnoty space, rule nebo use-content. Hodnota space znamená vložení mezer, hodnota rule znamená vložení čáry určené atributy *rule-style* (styl čáry) a *rule-thickness* (tloušťka čáry) a hodnota use-content znamená vložení obsahu definovaného potomky elementu *fo:leader* (potomky může být text nebo in-line elementy).

Element *fo:leader* může obsahovat kromě atributu *leader-pattern* ještě další speciální atributy. Například atribut *leader-length* určuje délku vodící linky a atribut *leader-pattern-width* určuje šířku každé části vodící linky.

Příklad 8

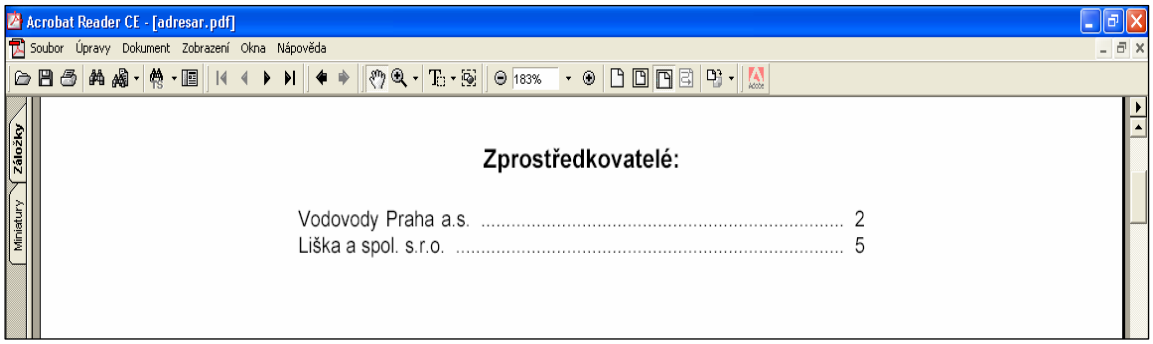
```
<fo:block font-size="12pt" space-after="0.5cm" font-weight="bold"
          text-align="center">
  Zprostředkovatelé:
</fo:block>
<fo:block space-after="1cm">
  <fo:block font-size="10pt" start-indent="3cm" end-indent="3cm"
            text-align-last="justify">
    Vodovody Praha a.s.
    <fo:leader color="black" space-start="2mm" space-end="2mm"
               leader-pattern="dots"/>
    <fo:page-number-citation ref-id="N11"/>
  </fo:block>
  <fo:block font-size="10pt" start-indent="3cm" end-indent="3cm"
            text-align-last="justify">
    Liška a spol. s.r.o.
    <fo:leader color="black" space-start="2mm" space-end="2mm"
               leader-pattern="dots"/>
    <fo:page-number-citation ref-id="N21"/>
  </fo:block>
</fo:block>
```

3.11 Odkazy, dynamické efekty

Odkazy na jiný dokument nebo odkazy na místo v aktuálním dokumentu se vytvoří pomocí elementu *fo:basic-link*. Pomocí atributu *external-destination* se určí adresa URI, na které je umístěn dokument, na který odkazujeme nebo pomocí atributu *internal-destination* určíme id místa v aktuálním dokumentu.

Další atributy, které může tento element obsahovat, jsou například *indicate-destination*, který určuje zda se cílové místo odkazu má nějak zvýraznit (hodnota true) nebo *show-destination*, který určuje, zda se má dokument otevřít v novém okně (hodnota new) nebo má nahradit stávající dokument

Obrázek 3



(hodnota replace). Atribut *destination-placement-offset* potom určuje, jak daleko má být cíl odkazu od horního okraje stránky.

Další dynamické efekty se dají tvořit pomocí elementu *fo:multi-case* a jeho potomků. Tímto elementem se zde však nebudu zabývat, protože je určen především pro interaktivní webové stránky.

3.12 Plovoucí objekty, živé záhlaví

Plovoucí objekty jsou objekty (např. obrázky, tabulky), jejichž umístění na stránce nepotřebujeme přesně určit, stačí nám jejich přibližné umístění v okolí určitého odstavce textu. Takovéto objekty se definují pomocí elementu *fo:float*. Tento element může mít atributy *float* a *clear*. Atribut *float* říká, v jaké části stránky se má plovoucí objekt zobrazit (hodnoty before, start, end, left, right, none) a atribut *clear* říká, zda je povoleno obtékání plovoucího objektu textem. Jeho hodnoty (start, end, left, right, both, none) zakazují obtékání na určené straně, přičemž hodnota none znamená žádný zákaz.

Příkladem pojmu živé záhlaví je záhlaví, které obsahuje aktuální název kapitoly. Takové záhlaví se dá vytvořit pomocí kombinace elementů *fo:marker* a *fo:retrieve-marker*. Element *fo:marker* se používá v toku textu k obalení například právě názvu kapitoly a element *fo:retrieve-marker* se používá při definování záhlaví či jiného pevného obsahu stránky ke zkopírování obsahu elementu *fo:marker* z aktuální stránky. Element *fo:marker* obsahuje atribut *marker-class-name*, na nějž se pak odkazuje atribut *retrieve-class-name* elementu *fo:retrieve-marker*. Element *fo:retrieve-marker* může ještě obsahovat atribut *retrieve-boundary*, který omezuje použití určitého elementu *fo:marker* na stránku, na sekvenci stránek nebo na celý dokument (hodnoty page, page-sequence, dokument) nebo atribut *retrieve-position*, který určuje preferenci toho, jaký element *fo:marker* bude použit (hodnoty first-starting-within-page, first-including-carryover, last-starting-within-page, last-ending-within-page).

3.13 Formátovací vlastnosti

Existuje velké množství formátovacích vlastností a nemá cenu je zde všechny detailně popisovat. Ty, které využiji v této práci, popíšu při srovnání formátovacích objektů s formátem WordML. Ty nejdůležitější již také byly částečně popsány v příkladech. Existují však určitá pravidla, která se musí dodržet při zpracování dokumentu s formátovacími objekty z nichž některé zde nastíním.

Většina vlastností se dá použít u více formátovacích objektů. Jestliže hodnota vlastnosti není u některého objektu určena, doplní se buď z jeho rodičů nebo se doplní její výchozí hodnota. Jestliže je hodnota zadána jako funkce nebo výraz nebo relativní či procentuální hodnota, musí se nejprve přepočítat na absolutní hodnotu.

Jako absolutní jednotky můžeme u formátovacích vlastností použít centimetry (cm), milimetry (mm), palce (1 in = 2,54 cm), body (1 pt = 1/72 in) a pc (1 pc = 12 pt). Jako relativní jednotku můžeme využít em, kde 1 em je rovna aktuální velikosti písma.

Dále můžeme některé vlastnosti zadat v rozmezí jako minimální, maximální a optimální hodnotu. Uděláme to tak, že k názvu vlastnosti připojíme tečkou klíčová slova minimum, maximum nebo optimum (např. *space-before.optimum*="2cm").

Jako funkce můžeme používat například minimum (min), maximum (max), absolutní hodnota (abs) a další. Můžeme používat též základní aritmetické funkce. Dále též můžeme používat některé speciální funkce k výpočtu nějaké hodnoty (např. body-start() popsaná v části 3.6).

4 Formát WordML

Formát WordML [8] je formátem XML, ve kterém se dá uložit dokument napsaný v textovém editoru Microsoft Word 2003. Tento formát používá několik jmenných prostorů, z nichž nejdůležitější je jmenný prostor s názvem „XML Dokument 2003“, který definuje základní elementy a atributy používané k popisu dokumentů. Tento jmenný prostor používá prefix *w*. Další dva jmenné prostory, které pravděpodobně budu v této práci využívat mají názvy „Annotation Markup Languague“ a „Auxiliary XML Dokument 2002“. První z nich používá prefix *aml* a je důležitý, protože se jeho pomocí definují mimo jiné záložky v dokumentu (místa v dokumentu, na které může vést odkaz). Druhý z nich používá prefix *w10* a slouží k upřesnění pozice a dalších vlastností obrázků. Poslední čtyři jmenné prostory v této práci pravděpodobně nevyužiji. Tyto jmenné prostory mají názvy „Schema Library“, „Auxiliary XML Dokument 2003“, „Vector Markup Languague“ a „Common Properties“. První z nich používá prefix *sl* a slouží k odkazům na schémata používaných v dokumentu. Druhý používá prefix *wx* a slouží jako pomůcka při transformaci dokumentu do formátu HTML. Třetí jmenný prostor s prefixem *v* slouží k uchování vektorové grafiky a nakonec čtvrtý jmenný prostor s prefixem *o* slouží k uchování obecných a statistických vlastností dokumentu. Tabulka 1 ukazuje URI všech těchto jmenných prostorů.

Tabulka 1

Prefix jmenného prostoru	URI
w	http://schemas.microsoft.com/office/word/2003/wordml
aml	http://schemas.microsoft.com/aml/2001/core
w10	urn:schemas-microsoft-com:office:word
sl	http://schemas.microsoft.com/schemaLibrary/2003/core
wx	http://schemas.microsoft.com/office/word/2003/auxHint
v	urn:schemas-microsoft-com:office:vml
o	urn:schemas-microsoft-com:office:office

V dalších částech této kapitoly stejně tak jako v předchozí kapitole nejprve popíšu strukturu dokumentu ve formátu WordML a poté použití jednotlivých elementů a atributů důležitých pro formátování dokumentu.

4.1 Struktura dokumentu ve formátu WordML

Formát WordML je založený na elementech. To znamená, že pro většinu vlastností a nastavení dokumentu jsou používány elementy. Atributy jsou používány primárně pouze pro odkazy a identifikaci.

Celý dokument je obalen elementem *w:wordDocument*. Tento element může dále obsahovat elementy[1] *w:fonts*, *w:lists*, *w:styles*, *w:bgPict*, *w:docPr* a *w:body*. Element *w:fonts* obsahuje definici

[1] A další elementy, které však nepovažuji za důležité a nebudu se jimi proto zabývat.

fontů písmen používaných v dokumentu, element *w:lists* obsahuje definici seznamů používaných v dokumentu, element *w:styles* obsahuje definici stylů používaných pro formátování dokumentu, element *w:bgPict* definuje barvu či obrázek, který je na pozadí celého dokumentu, element *w:docPr* obsahuje spoustu vlastností dokumentu (např. v jakém přiblížení se dokument zobrazí) a nakonec element *w:body* obaluje celý obsah dokumentu.

Element *w:body* se skládá z odstavců (element *w:p*), z tabulek (element *w:tbl*), z in-line definicí stylů, fontů a seznamů (element *w:cfChunk* a jeho potomci *w:styles*, *w:fonts* a *w:lists*) a z definice rozvržení stránek vztahující se k předchozímu textu (element *w:sectPr*).

Odstavec se skládá z definice jeho vlastností (element *w:pPr*), z elementu *w:r* obalujícího data dokumentu jako jsou text, obrázky a další, dále z odkazů dovnitř dokumentu nebo na jiný dokument (element *w:hlink*) nebo též z elementu *w:fldSimple*, který obsahuje za běhu počítané vlastnosti, jako jsou například čísla stránek.

Element *w:r* tedy obsahuje text v elementu *w:t* a další elementy reprezentující data dokumentu. Dále obsahuje element *w:rPr*, který definuje vlastnosti daného textu, jako je například font, velikost a barva písma.

4.2 Rozvržení stránek

Jak bylo řečeno výše, rozvržení stránek a jejich další společné vlastnosti se definují pomocí elementu *w:sectPr*. Takto definované vlastnosti se vztahují vždy k předchozímu obsahu dokumentu, dokument tak může obsahovat několik sekcí s různě rozvrženými stránkami. Jestliže však dokument obsahuje více sekcí, musí být tyto sekce definovány v elementu *w:pPr*, až na poslední, která je definována jako potomek elementu *w:body*.

Prvními vlastnostmi, které se zde mohou definovat jsou výška a šířka stránky (element *w:pgSz*), dále pak velikost okrajů stránky, velikost záhlaví a zápatí a velikost okraje stránky ztracená při vázání listů (element *w:pgMar*). Příklad 9 ukazuje použití těchto elementů. Všechny hodnoty jsou uvedeny v jednotkách twips, přičemž 1 in má 1440 twips a 1 in má 2,54 cm. Šířka stránky je tedy nastavena na 21 cm (11906 twips) a její výška na 29,7 cm (16838 twips). Všechny okraje na stránce jsou nastaveny na 2,5 cm (1417 twips) a velikost záhlaví a zápatí je 1,25 cm (708 twips), přičemž je to část za okrajem stránky (horní a dolní okraj stránky je tedy ve skutečnosti pouze 1,25 cm). Atribut *w:gutter* určuje, jaký okraj se má přidat k levé části stránky z důvodu ztráty místa při vázání.

Příklad 9

```
<w:sectPr>
  <w:pgSz w:w=”11906” w:h=”16838”/>
  <w:pgMar w:top=”1417” w:right=”1417” w:bottom=”1417”
           w:left=”1417” w:header=”708” w:footer=”708”
           w:gutter=”0”/>
</w:sectPr>
```

Dále se zde definuje samotný obsah záhlaví a zápatí a to pomocí elementů *w:hdr* a *w:ftr*. Oba elementy obsahují atribut *w:type*, který říká na jaké straně se záhlaví či zápatí vyskytuje. Hodnota even ho zobrazí na lichých stránkách, hodnota odd na sudých stránkách a hodnota first pouze na první stránce této sekce. Elementy dále mohou obsahovat stejné elementy jako element *w:body*, ve kterých je definován obsah záhlaví a zápatí.[2]

Také zde můžeme definovat vlastnosti poznámek pod čarou a to pomocí elementu *w:footnotePr*. Dá se zde určit umístění poznámek pod čarou (element *w:pos*), formát automatického číslování (element *w:numFmt*), počáteční značku (element *w:numStart*) a místo, kde se má číslování restartovat (element *w:numRestart*). Všechny tyto elementy obsahují atribut *w:val*, pomocí kterého se určí daná vlastnost. Poznámky pod čarou mohou být umístěny například v dolní části stránky nebo na konci dokumentu (hodnoty page-bottom, doc-end). Formát automatického číslování může být například číselný, malá písmena, velká písmena a další (hodnoty decimal, lower-letter, upper-letter). Číslování se může restartovat v každé sekci nebo na každé stránce (hodnoty each-sect, each-page).

Pomocí elementu *w:type* se dá nastavit, kde má tato sekce začínat, zda na další straně, v dalším sloupci, tam, kde skončila předchozí, na sudé straně nebo na liché straně. To se určí pomocí atributu *w:val* elementu *w:type* a jeho hodnot next-page, next-column, continuous, even-page, odd-page.

Dále se dá pomocí elementu *w:pgBorders* nastavit orámování stránek v této sekci. Tento element může mít atribut *w:display*, který určuje na jakých stránkách má být orámování zobrazeno (hodnoty all-pages, first-page, not-first-page) a atribut *w:z-order*, který specifikuje relativní umístění orámování k ostatnímu obsahu stránky (hodnoty front, back). Element dále obsahuje potomky *w:top*, *w:bottom*, *w:left* a *w:right*, kteří pomocí svých atributů určují vzhled ohraničení na dané straně stránky. Jsou to atributy *w:val* určující styl ohraničení (např. čára, dvojitá čára, tečky), *w:color* určující barvu ohraničení, *w:sz* určující šířku čáry v bodech a další méně podstatné atributy.

Pomocí elementu *w:pgNumType* a jeho atributu *w:fmt* se dá stanovit formát číslování stránek (např. decimal nebo upper-roman). Dále pomocí atributu *w:start* se dá stanovit počáteční číslo stránky v této sekci, jestliže není zadáno, pokračuje se v číslování z předchozí sekce.

Dále zde můžeme definovat počet a šířku sloupců a mezery mezi nimi. K tomu slouží element *w:cols*. Jeho atribut *w:equalWidth* určuje, zda jsou všechny sloupce stejně široké (hodnoty on, off), atribut *w:sep* určuje, zda je mezi sloupci zobrazena čára (též hodnoty on, off), atribut *w:num* určuje počet sloupců a atribut *w:space* určuje velikost mezery mezi sloupci v jednotkách twips. Jestliže všechny sloupce nejsou stejně široké je potřeba každý sloupec definovat v elementu *w:col*. Jeho atribut *w:w* pak určuje jeho šířku v jednotkách twips a atribut *w:space* mezeru mezi tímto sloupcem a sloupcem následujícím (též v twipsech).

[2] Při použití jiného záhlaví na sudých a lichých stránkách je potřeba do elementu *w:docPr* vložit prázdný element *w:evenAndOddHeaders* a při použití jiného záhlaví na titulní straně je potřeba na konec elementu *w:sectPr* vložit prázdný element *w:titlePg*.

A nakonec pomocí elementu *w:textFlow* můžeme definovat směr toku textu. V jeho atributu *w:val* můžeme nastavit hodnoty lr-tb, tb-rl, bt-lr, lr-tb-v nebo tb-rl-v (hodnota lr-tb například znamená zleva doprava a shora dolů).

4.3 Další vlastnosti dokumentu

V elementu *w:docPr* se dá nastavit mnoho obecných vlastností platných pro celý dokument. Uvedu zde pouze pár příkladů, jelikož většinu těchto vlastností v této práci nevyužiji.

Tak například se zde dá nastavit pomocí elementu *w:view* mód zobrazení dokumentu ve Wordu. Atribut *w:val* může například nabývat hodnoty print, normal nebo web, které znamenají rozložení při tisku, normální rozložení nebo zobrazení jako webová stránka.

Dále se zde dá pomocí elementu *w:zoom* nastavit zvětšení či zmenšení stránky na obrazovce. Pomocí atributu *w:percent* se dá nastavit hodnota v procentech (10% až 500%) a pomocí atributu *w.val* se dá nastavit, jak se má stránka zobrazit (např. full-page, best-fit, text-fit).

Pomocí elementu *w:mirrorMargins* se dá určit, zda levá a pravá stránka mají mít zrcadlové okraje (tzn. na pravé stránce se prohodí levý a pravý okraj). Určí se tak pomocí atributu *w:val* a jeho hodnoty on nebo off.

Pomocí elementu *w:gutterAtTop* můžeme okraj přidaný ke stránce z důvodu ztráty místa při vázání umístit k hornímu okraji stránky místo k levému okraji. Opět pomocí atributu *w:val* a jeho hodnoty on.

Dále zde také můžeme nastavit vlastnosti týkající se dělení slov v dokumentu. Pomocí elementu *w:autoHyphenation* můžeme nastavením atributu *w:val* na hodnotu on zapnout automatické dělení slov. Pomocí elementu *w:consecutiveHyphenLimit* a jeho atributu *w:val* určujeme maximální počet řádků za sebou s děleným slovem. Další element *w:hyphenationZone* stanovuje maximální vzdálenost od pravého okraje stránky v jednotkách twips, kde se má uplatnit dělení slov. A nakonec element *w:doNotHyphenCaps* říká, aby se nedělila slova psaná pouze velkými písmeny.

Nakonec zde můžeme pomocí elementu *w:footnotePr* stanovit určité vlastnosti poznámek pod čarou, avšak jiné než uvedené v předchozí části. Tento element má až tři potomky *w:footnote*, jejichž atribut *w:type* říká, jaké části poznámky pod čarou se definice týká. Hodnota separator říká, že se jedná o oddělovač poznámky od textu, hodnota continuation-separator říká, že se jedná o oddělovač pokračování poznámky z předchozí stránky od textu a hodnota continuation-notice říká, že se jedná o odkaz na pokračování poznámky. Tento element může obsahovat ještě atribut *w:suppressRef*, jehož hodnota on potlačuje automatické vložení znaku odkazu poznámky pod čarou. Element *w:footnote* může dále obsahovat stejné elementy jako element *w:body*, ve kterých jsou definovány dané oddělovače.

4.4 Vlastnosti odstavců

Pro každý odstavec můžeme určit řadu vlastností pomocí elementu *w:pPr*, který je potomkem elementu *w:p*. Opět neuvádím všechny vlastnosti, ale jen ty, které pravděpodobně využiji.

První skupinou vlastností, která se zde dá nastavit, je skupina vlastností týkající se zalomení stránky. Elementem *w:keepNext* můžeme požadovat, aby mezi tímto a dalším odstavcem nebyl konec stránky. Pomocí elementu *w:keepLines* dále můžeme požadovat, aby celý obsah tohoto odstavce byl na jedné stránce. Další element *w:pageBreakBefore* umístí tento odstavec na nové stránce. A nakonec element *w:widowControl* stanoví pravidlo, které zakazuje umístit samotný první řádek odstavce na konec stránky nebo samotný poslední řádek odstavce na začátek stránky. Všechny tyto elementy se nastavují pomocí atributu w:val a hodnot on, off.

Pomocí elementu *w:pBdr* se dají nastavit ohraničení odstavce. Element obsahuje potomky *w:left*, *w:right*, *w:top* a *w:bottom*, pomocí kterých se nastaví ohraničení na všech stranách odstavce, dále potomka *w:between*, který určí hranici mezi odstavci a nakonec *w:bar*, který stanoví stejné hranice na všech čtyřech stranách odstavce. Hranice se určí pomocí atributů *w:val*, *w:color* a *w:sz*, které byly popsány v části 4.2 v odstavci popisujícím ohraničení stránky.

Elementem *w:shd* můžeme nastavit stínování odstavce. Atributem *w:val* můžeme nastavit styl stínování (různé vzory na pozadí odstavce), atribut *w:color* určí barvu vybraného stylu stínování a atribut *w:fill* určí barvu výplně pozadí.

Další element *w:bidi* stanovuje směr toku textu pro tento odstavec zprava doleva a to nastavením jeho atributu *w:val* na hodnotu on.

Pomocí elementu *w:spacing* a jeho atributů se dají stanovit velikosti mezer mezi odstavci a mezi řádky. K určení mezery před odstavcem se dají využít atributy *w:before* (hodnota v twipsech) nebo *w:before-lines* (hodnota v řádcích). K určení mezery za odstavcem se pak dají využít atributy *w:after* nebo *w:after-lines*. Mezera mezi řádky se určí pomocí atributu *w:line* (hodnota v twipsech) a interpretace této hodnoty se určí atributem *w:line-rule* (hodnoty exact, at-least).

Dále můžeme pomocí elementu *w:ind* nastavit odsazení jednotlivých řádků odstavce. Velikost odsazení (nebo předsazení při záporné hodnotě) od levého okraje stránky můžeme určit pomocí atributu *w:left* (v twipsech) nebo pomocí atributu *w:left-chars* (v počtu znaků). Obdobně můžeme určit odsazení zprava. Dále můžeme pomocí atributů *w:hanging* nebo *w:hanging-chars* určit, o kolik je první řádek předsazen oproti ostatním řádkům odstavce. A nakonec můžeme pomocí atributů *w:first-line* nebo *w:first-line-chars* určit odsazení prvního řádku oproti ostatním.

Pomocí elementu *w:jc* a jeho atributu *w:val* můžeme nastavit zarovnání daného odstavce. Hodnoty left, right, center a both popořadě říkají zarovnat vlevo, vpravo, na střed, do bloku. Přípustné jsou ještě další hodnoty, pravděpodobně pro zarovnání textu s jiným směrem toku než je zvykem u nás.

Jestliže se jedná o odstavec textu v buňce tabulky, v textovém poli nebo v textovém rámu, potom je možno určit též směr toku textu pomocí elementu *w:textDirection* a jeho atributu *w:val*.

Dále se zde dají také definovat vlastnosti seznamu, tabulátory a vodící linky pomocí elementů *w:listrPr* a *w:tabs*. Tyto definice budou popsány dále v kapitolách o tvorbě seznamů a o tvorbě tabulátorů a vodících linek.

Definice vlastností odstavce může též být definována pomocí stylu. Vlastnosti odstavce pak mohou obsahovat pouze element *w:pStyle*, který se pomocí atributu *w:val* odkazuje na daný styl. Styl se může definovat pomocí elementu *w:style*, který je potomkem elementu *w:styles*[3]. Element *w:style* může obsahovat atributy *w:type*, *w:styleId* a *w:default*. Atribut *w:type* určuje druh definovaného stylu (hodnoty paragraph, character, table, list), atribut *w:styleId* dává stylu jméno používané k odkazu na tento styl a atribut *w:default* určuje, jestli je daný styl výchozí pro daný druh stylu. Potomky tohoto elementu jsou pak hlavně definice vlastností odstavců (element *w:pPr*), vlastností textu (element *w:rPr*), vlastností tabulek (element *w:tblPr*) a dalších vlastností. Element dále může mít potomka *w:basedOn*, který říká, že tento styl je založen na jiném stylu a dále má vlastnosti, které ho od toho stylu odlišují. Posledním zajímavým potomkem je element *w:next*, který určuje název stylu odstavce, který se má aplikovat na následující odstavec.

4.5 Vlastnosti textu

Vlastnosti textu v odstavci definujeme pomocí elementu *w:rPr*, který je potomkem elementu *w:r*. Stejně jako u vlastností odstavce můžeme vlastnosti textu definovat buď přímo nebo odkazem na nějaký styl. Na styl se zde můžeme odkázat pomocí elementu *w:rStyle*, přičemž se však uplatní pouze formátovací vlastnosti stylu určené pro text, nikoliv vlastnosti určené pro odstavec.

První vlastností, která se zde dá nastavit, je písmo. Můžeme ho nastavit pomocí elementu *w:rFonts* a jeho atributů *w:ascii*, *w:h-ansi*, *w:fareast*, *w:cs* a *w:hint*. Prvními čtyřmi atributy můžeme nastavit název fontu použitého pro daný typ fontu a atribut *w:hint* říká, jaký typ fontu by se měl použít (hodnoty default, fareast, cs).

Dále můžeme pomocí elementu *w:sz* určit velikost písma pro první tři typy fontů a pomocí elementu *w:sz-cs* velikost písma pro typ fontu „comlex script“. Velikost písma se stanoví pomocí atributu *w:val* a to v polovinách bodů (což je 1/144 in).

Dále samozřejmě můžeme stanovit barvu písma a to pomocí elementu *w:color* a jeho atributu *w:val*. Barva se zadává jako hexadecimální hodnota ve formátu RRGGBB.

Všechny následující elementy používají atribut *w:val* k zapnutí (on) či vypnutí (off) dané vlastnosti textu. Element *w:b* nastavuje tučné písmo a element *w:i* nastavuje kurzívu. Element *w:caps* formátuje všechna malá písmena jako velká a element *w:smallCaps* formátuje všechna malá písmena jako velká a navíc zmenší jejich velikost. Dále element *w:strike* nakreslí čáru skrz text a element *w:dstrike* nakreslí dvojitou čáru skrz text. Element *w:shadow* zobrazí ke každému písmenu stín, element *w:emboss* zobrazí text jako by vyvstával ze stránky a nakonec element *w:imprint* zobrazí text jako by byl vtlačen do stránky. Elementem *w:rtl* se dá ještě nastavit směr toku textu na zprava doleva.

[3] Jak bylo řečeno výše, tento element může být potomkem kořenového elementu *w:wordDocument* nebo elementu *w:cfChunk*, který je využíván pro in-line definici stylů v těle dokumentu.

Následující skupina elementů manipuluje s velikostí písma nebo mezer mezi písmeny a to pomocí atributu *w:val*. Elementem *w:spacing* můžeme nastavit hodnotu v twipsech, která říká, o kolik jsou zvětšeny nebo zmenšeny mezery mezi písmeny oproti normálu. Další element *w:w* pak umožňuje stanovit hodnotu v procentech o kterou je text horizontálně zúžen nebo rozšířen. A nakonec element *w:kern* stanoví nejmenší velikost písma v polovinách bodů, pro kterou je ještě automaticky uplatňován kerning (úprava vzdálenosti mezi písmeny, aby všechna byla vizuálně stejně od sebe vzdálená).

Následující dva elementy upravují vertikální polohu textu též pomocí atributu *w:val*. Element *w:position* určuje hodnotu v polovinách bodů, o kterou má být text posunut vertikálně nahoru nebo dolů z vzhledem k normální pozici. Naproti tomu element *w:vertAlign* sice též posunuje text nahoru nebo dolů (podle hodnot superscript nebo subscript), ale zároveň zmenší velikost písma. To znamená, že výsledkem je horní nebo dolní index, který se zpátky vrátí nastavením hodnoty na baseline.

Další element *w:u* nastavuje podtržení písma. Tento element má dva atributy. První atribut *w:val* nastavuje styl podtržení (např. čára, pouze slova, dvojitá čára) a druhý atribut *w:color* nastavuje barvu podtržení.

Dalšími dvěmi elementy *w:bdr* a *w:shd* můžeme nastavit ohraničení a stínování (pozadí) textu. Element *w:bdr* nemá žádné potomky, ale jinak má stejné atributy jako element *w:pBdr*, který byl popsán v části 4.4. Element *w:shd* byl také již popsán tamtéž. Elementem *w:effect* ještě můžeme nastavit další animovaný efekt textu (např. blikání, blikající rámeček či blikající pozadí).

Posledními dvěmi elementy, které se zde dají použít, jsou elementy *w:lang* a *w:hyphen*. Prvním z nich se dá nastavit jazyk daného textu pomocí atributů *w:val*, *w:fareast* nebo *w:bidi*. Jaký použijeme atribut záleží na tom, jestli se jedná o jazyk latinský, asijský nebo o „complex script“. Druhým se dá nastavit styl dělení slov. Pomocí atributu *w:val* můžeme určit dělící znak a pomocí atributu w:rule můžeme nastavit pravidlo pro dělení slov (např. hodnoty none, normal, add-before, delete-before).

4.6 Tabulátory, vodící linky

Tabulátory se používají k posunutí textu na určité místo v řádku. To znamená, že po jeho použití se místo vyplní mezerami (nebo definovanými znaky) a text pokračuje až za nimi. Tabulátory se definují ve vlastnostech odstavce pomocí elementu *w:tabs*.

Jednotlivý tabulátor se potom definuje pomocí elementu *w:tab* a jeho atributů *w:val*, *w:leader* a *w:pos*. První atribut slouží k nastavení zarovnání textu napsaného u zarážky a může nabývat například hodnot left, right, center nebo decimal. Druhý atribut slouží k nastavení znaků, které vyplní prostor před tabulátorem. Mohou to být čára, tečky, pomlčky nebo mezery. Poslední atribut nastavuje pozici zarážky v twipsech, místo před ní se tedy vyplní nastavenými znaky a za ní normálně pokračuje text.

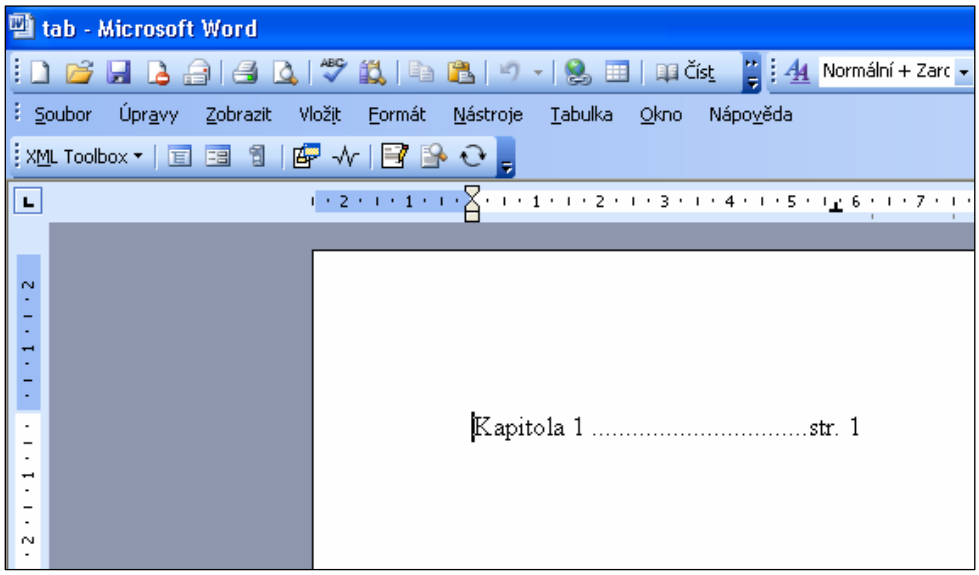
Tabulátor se používá na úrovni elementu *w:r* a to jednoduše zápisem prázdného elementu *w:tab*. Je-li definováno více tabulátorů, použije se nejbližší z nich.

Použití tabulátorů ukazuje příklad 10. Příklad není třeba komentovat, výsledek je zobrazen na obrázku 4.

Příklad 10

```
<w:p>
  <w:pPr>
    <w:jc val=”both”/>
    <w:tabs>
      <w:tab w:val=”center” w:leader=”dot” w:pos=”3200”/>
    </w:tabs>
  </w:pPr>
  <w:r>
    <w:t>Kapitola 1</w:t>
    <w:tab/>
    <w:t>str. 1</w:t>
  </w:r>
</w:p>
```

Obrázek 4



4.7 Seznamy

Každý seznam použitý v dokumentu se musí nejprve definovat v elementu *w:lists*, který je potomkem kořenového elementu nebo elementu *w:cfChunk*. Samotná definice seznamu se provádí pomocí elementu *w:listDef*. Tento element má povinný atribut *w:listDefId*, který obsahuje číslo, které tuto definici jednoznačně identifikuje. Jeho nejdůležitějším potomkem je potom element *w:lvl*, ve kterém se definují jednotlivé vlastnosti daného stupně seznamu. Stupeň seznamu se nastaví pomocí jeho atributu *w:ilvl*.

Definici jednoduchého jednostupňového seznamu a jeho použití v dokumentu ukazuje příklad 11. Element *w:lvlText* obsahuje definici odrážky, v našem případě je to znak *. Element *w:lvlJc* definuje zarovnání odrážky (je důležité u číslovaného seznamu). Elementy *w:pPr* a *w:rPr* obsahují odstavcové a textové vlastnosti odrážky až na definovaný tabulátor. Ten určuje, jak daleko od levého okraje

stránky začíná textová položka seznamu. Naproti tomu element *w:ind* určuje odsazení odrážky od levého okraje stránky. Element *w:color* nastavuje barvu odrážky na červenou.

Jak se definice tohoto seznamu používá v těle dokumentu? Tak, že pro každou položku seznamu vytvoříme samostatný odstavec a do jeho vlastností vložíme element *w:listPr*. Tento element má dva potomky *w:ilvl* a *w:ilfo*. První z nich určuje stupeň seznamu pro tento odstavec a druhý se odkazuje na definici seznamu. Neodkazuje se přímo na atribut *w:listDefId* elementu *w:listDef*, nýbrž se odkazuje na atribut *w:ilfo* elementu *w:list*, jehož potomek *w:ilst* se pak již odkazuje na daný *w:listDefId*. Tento zprostředkující element *w:list* se používá, jelikož může pomocí elementu *w:lvlOverride* pozměnit nějaké vlastnosti definice seznamu, aniž se musí psát celá definice znovu.

Příklad 11

```
<w:lists>
 <w:listDef w:listDefId="1">
  <w:lvl w:ilvl="0">
   <w:lvlText w:val="*"/>
   <w:lvlJc w:val="left"/>
   <w:pPr>
    <w:tabs>
     <w:tab w:val="list" w:pos="1080"/>
    </w:tabs>
    <w:ind w:left="540"/>
   </w:pPr>
   <w:rPr>
    <w:color w:val="ff0000"/>
   </w:rPr>
  </w:lvl>
 </w:listDef>
 <w:list w:ilfo="1">
  <w:ilst w:val="1"/>
 </w:list>
</w:lists>
<w:body>
 <w:p>
  <w:pPr>
   <w:listPr>
    <w:ilvl w:val="0"/>
    <w:ilfo w:val="1"/>
   </w:listPr>
  </w:pPr>
  <w:r>
   <w:t>První položka seznamu</w:t>
  </w:r>
 </w:p>
</w:body>
```

Pomocí elementu *w:lvl* můžeme ještě kromě výše uvedených vlastností stanovit další vlastnosti týkající se automatického číslování seznamu. Například pomocí elementu *w:nfc* určíme číslo formátu seznamu, který chceme použít (např. 0 jsou čísla a 1 jsou římská čísla). Pomocí elementu *w:start* pak určíme, kolikátým prvkem seznamu chceme začít číslování. Tvar čísla určíme pomocí elementu *w:lvlText* ve tvaru „%level", což značí dané číslo (odrážku) a za tím jakýkoliv text, který se má spolu

s číslem zobrazit (např. tečka nebo závorka). Dále můžeme pomocí elementu *w:lvlRestart* určit, kolikátým prvkem seznamu se má číslování restartovat. Nakonec též můžeme jako odrážku určit obrázek pomocí elementu *w:lvlPicBulletId*, který svým atributem odkazuje na číslo obrázku definovaného na úrovni elementu *w:lists* elementem *w:listPicBullet*.

4.8 Tabulky

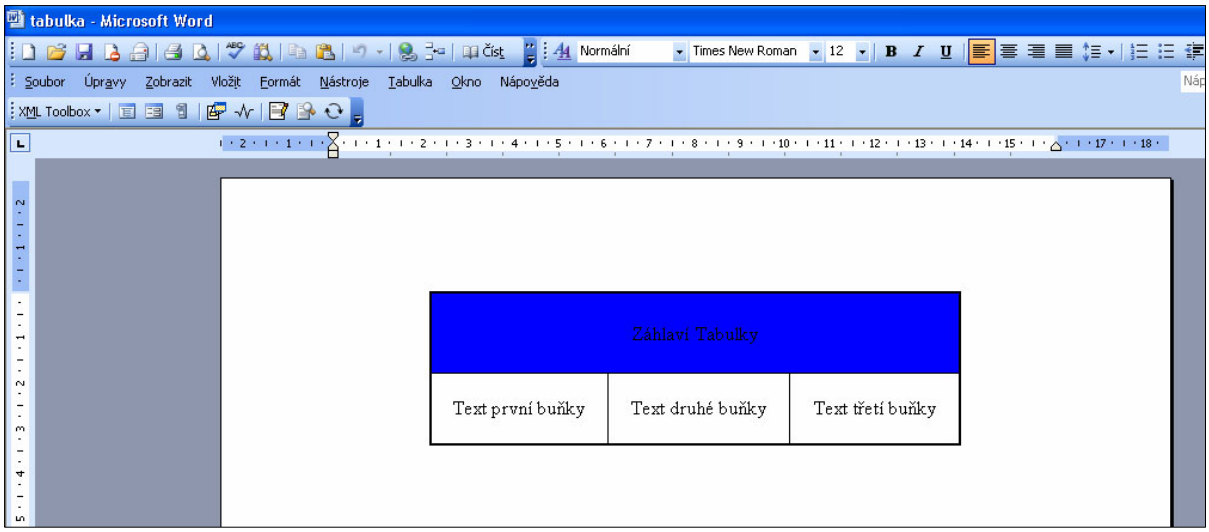
Tabulku můžeme vytvořit pomocí elementu *w:tbl*, který je potomkem elementu *w:body*. Tento element může mít až tři potomky. Prvním potomkem je element *w:tblPr*, pomocí kterého určujeme různé vlastnosti celé tabulky. Pomocí druhého potomka *w:tblGrid* definujeme počet a šířku sloupců v tabulce. Poslední potomek *w:tr* obsahuje řádek tabulky.

Každý řádek tabulky dále může obsahovat definici vlastností řádku tabulky (element *w:trPr*), výjimky vlastností celé tabulky platné pouze pro tento řádek (element *w:tblPrEx*) a nakonec samozřejmě jednotlivé buňky tabulky (element *w:tc*).

Každá buňka tabulky může dále obsahovat definici jejích vlastností (element *w:tcPr*) a samozřejmě též samotný obsah buňky, který je tvořen odstavcem.

Jelikož vlastností tabulek existuje celkem hodně, nebudu je všechny popisovat, nýbrž ukážu jejich použití na příkladu 12. Tabulka má definováno několik vlastností. Jsou to šířka tabulky 6000 twipsů, zarovnání na střed, ohraničení tabulky nahoře, vlevo, dole, vpravo, vnitřní oddělovací čáry horizontální a vertikální jako černá čára tlustá okolo tabulky 15 bodů a uvnitř tabulky 5 bodů, dále je toto rozložení tabulky určeno jako fixní a nakonec jsou definovány vnitřní okraje jednotlivých buněk tabulky z každé strany 288 twipsů. Dále jsou definovány sloupce tabulky, v našem případě tři, z nichž každý je široký 2000 twipsů. Následuje definice jednotlivých řádků tabulky. První z nich je definován jako hlavička tabulky a jeho výška je nastavena na 720 twipsů. Následují jednotlivé buňky tabulky. Element *w:hmerge* s hodnotou restart začíná od této buňky slučovat buňky tabulky horizontálně a to

Obrázek 5



všechny ty, které ve svých vlastnostech obsahují také tento element. Element *w:shd* nastavuje pozadí buňky na modrou barvu a element *w:vAlign* nastavuje vertikální zarovnání textu v buňce. Samotný obsah buňky musí být dále definován jako odstavec. Musíme vždy definovat všechny buňky, které chceme zobrazit, i ty které slučujeme. Tabulka má dále ještě jeden řádek s nesloučenými buňkami. Výsledné zobrazení tabulky ukazuje obrázek 5.

Příklad 12

```
<w:tbl>
 <w:tblPr>
  <w:tblW w:w="6000" w:type="dxa"/><w:jc w:val="center"/>
  <w:tblBorders>
   <w:top w:val="single" w:color="000000" w:sz="15"/>
   <w:left w:val="single" w:color="000000" w:sz="15"/>
   <w:bottom w:val="single" w:color="000000" w:sz="15"/>
   <w:right w:val="single" w:color="000000" w:sz="15"/>
   <w:insideH w:val="single" w:color="000000" w:sz="5"/>
   <w:insideV w:val="single" w:color="000000" w:sz="5"/>
  </w:tblBorders>
  <w:tblLayout w:type="Fixed"/>
  <w:tblCellMar>
   <w:top w:w="288" w:type="dxa"/><w:left w:w="288" w:type="dxa"/>
   <w:bottom w:w="288" w:type="dxa"/><w:right w:w="288" w:type="dxa"/>
  </w:tblCellMar>
 </w:tblPr>
 <w:tblGrid>
  <w:gridCol w:w="2000"/><w:gridCol w:w="2000"/>
   <w:gridCol w:w="2000"/>
 </w:tblGrid>
 <w:tr>
  <w:trPr>
   <w:trHeight w:val="720" w:h-rule="exact"/><w:tblHeader w:val="on"/>
  </w:trPr>
  <w:tc>
   <w:tcPr>
    <w:hmerge w:val="restart"/><w:vAlign w:val="center"/>
    <w:shd w:val="clear" w:fill="0000ff"/>
   </w:tcPr>
    <w:p><w:pPr><w:jc w:val="center"/></w:pPr>
     <w:r><w:rPr><w:color w:val="000000"/></w:rPr>
      <w:t>Záhlaví Tabulky</w:t></w:r></w:p>
  </w:tc>
  <w:tc>
   <w:tcPr><w:hmerge/></w:tcPr><w:p></w:p>
  </w:tc>
  <w:tc>
   <w:tcPr><w:hmerge/></w:tcPr><w:p></w:p>
  </w:tc>
 </w:tr>
 <w:tr>
  <w:trPr>
   <w:trHeight w:val="600" w:h-rule="exact"/>
  </w:trPr>
  <w:tc><w:tcPr><w:vAlign w:val="center"/></w:tcPr>
   <w:p><w:r><w:t>Text první buňky</w:t></w:r></w:p>
  </w:tc>
  <w:tc><w:tcPr><w:vAlign w:val="center"/></w:tcPr>
   <w:p><w:r><w:t>Text druhé buňky</w:t></w:r></w:p>
  </w:tc>
  <w:tc><w:tcPr><w:vAlign w:val="center"/></w:tcPr>
   <w:p><w:r><w:t>Text třetí buňky</w:t></w:r></w:p>
  </w:tc>
 </w:tr>
</w:tbl>
```

4.9 Obrázky

Obrázky se vkládají do dokumentu na úrovni elementu *w:r* a to pomocí elementu *w:pict*. Tento element dále obsahuje definici rámu okolo obrázku ve formátu VML (Vector Markup Languague), což je jazyk na bázi XML uchovávající vektorovou grafiku. Dále v elementu *w:binData* následují data obrázku zakódovaná v base64. Jestliže nechceme, aby byl obrázek uložen současně s dokumentem, můžeme použít místo elementu *w:binData* element *v:imageData*. V tomto elementu použijeme atribut src k určení cesty k souboru s obrázkem.

Příklad 13 ukazuje vložení obrázku do dokumentu právě pomocí elementu *v:imageData*. Nejprve se tedy musí definovat rám pomocí elementu *v:shape*. V jeho atributu style jsou určeny jeho vlastnosti oddělené středníkem. V našem případě má rám 150% své základní šířky i výšky, horizontálně je umístěn ve středu stránky, vertikálně je umístěn 0,5 cm pod předcházejícím odstavcem a text musí být vzdálen od rámu minimálně 2,85 bodů nahoře i dole. Element *v:imagedata* vloží do výše definovaného rámu obrázek obr.jpg. Poslední element *w10:wrap* určuje typ obtékání obrázku, v našem případě nahoře a dole.

Příklad 13

```
<w:p>
 <w:r>
  <w:pict>
   <v:shape id="1"  style="margin-top:0.5cm;
          position:absolute;width:150%;height:150%;
          mso-wrap-distance-top:2.85pt;
          mso-wrap-distance-bottom:2.85pt;
          mso-position-horizontal:center;
          mso-position-horizontal-relative:page;
          mso-position-vertical:absolute;
          mso-position-vertical-relative:text">
    <v:imagedata src="obr.jpg" />
    <w10:wrap type="topAndBottom"/>
   </v:shape>
  </w:pict>
 </w:r>
</w:p>
```

4.10 Odkazy

Odkazy na jiný dokument nebo odkazy na jiné místo v aktuálním dokumentu můžeme vytvořit pomocí elementu *w:hlink*, který je potomkem elementu *w:p*. Tento element tedy obaluje text (elementy *w:r* a *w:t*), který je odkazem. Atribut *w:dest* určuje URL cílového dokumentu, který se má otevřít. Pomocí atributu *w:bookmark* můžeme odkázat na místo v aktuálním dokumentu. Tento atribut odkazuje na název záložky vytvořené na nějakém místě dokumentu. Záložka se vytvoří pomocí elementu *aml:annotation* a obaluje cílový odstavec. Záložka obsahuje atributy *aml:id*, *w:type* a *w:name*. První se používá k identifikaci dvojice počátek a konec záložky. Druhý označuje, zda se

jedná o začátek záložky (hodnota Word.Bookmark.Start) nebo konec záložky (hodnota Word.Bookmark.End). Poslední se používá k pojmenování záložky používaného při odkazování na ní.

4.11 Poznámky pod čarou

Jak už bylo řečeno výše, některé vlastnosti poznámek pod čarou se definují v elementu *w:sectPr* a některé v elementu *w:docPr*. Jak se ale dá poznámka pod čarou vložit do dokumentu? Tak, že se pro ní v odstavci vytvoří samostatný element *w:r*, který obsahuje element *w:footnote*. Tento element dále obsahuje odstavec se dvěmi elementy *w:r*. První element *w:r* obsahuje prázdný element *w:footnoteRef*, který vloží do dokumentu odkaz na poznámku. Druhý element *w:r* pak již obsahuje samotný text poznámky pod čarou. Odstavec i text poznámky pod čarou může samozřejmě být formátován pomocí elementů *w:pPr* a *w:rPr*.

4.12 Dynamické vkládání textu

Dynamickým vkládáním textu rozumím vkládání textu do dokumentu, který je generován samostatně podle kontextu. Například to jsou čísla stránek, záhlaví obsahující název kapitoly nebo generovaný obsah dokumentu.

Veškerý takto dynamicky generovaný text se vkládá do dokumentu pomocí elementu *w:fldChar*, který je potomkem elementu *w:r*. Tento element označuje textové pole, jehož začátek je označen pomocí atributu *w:fldCharType* a jeho hodnoty begin a konec je označen jeho hodnotou end. Mezi počátek a konec pole pak můžeme vložit dynamický text pomocí elementu *w:instrText*. Obsahem tohoto elementu je instrukce, která říká jaký text se má zobrazit.

Existuje celá řada takových instrukcí z nichž nejvýznamnější jsou instrukce PAGE, PAGEREF, STYLEREF a REF. Instrukce PAGE zobrazí aktuální číslo stránky, zatímco instrukce PAGEREF následovaná názvem záložky zobrazí číslo stránky, na kterém se nachází text obalený danou záložkou. Instrukce STYLEREF následovaná názvem stylu zobrazí nejbližší text zobrazený daným stylem. Tato instrukce se tak dá použít například pro zobrazení aktuální kapitoly v záhlaví. Poslední instrukce REF následovaná názvem záložky zobrazí text obalený danou záložkou.

Příklad 14

```
<w:sectPr>
 <w:ftr w:type="odd">
  <w:p>
   <w:r>
    <w:fldChar w:fldCharType="begin"/>
    <w:instrText>PAGE</w:instrText>
    <w:fldChar w:fldCharType="end"/>
   </w:r>
  </w:p>
 </w:ftr>
</w:sectPr>
```

Příklad 14 ukazuje, jak se dá v zápatí stránky zobrazit číslo stránky pomocí instrukce PAGE použité při definování zápatí.

5 Srovnání formátovacích objektů s formátem WordML

V této kapitole bych chtěl nastínit hlavní rozdíly mezi oběma formáty a upozornit na hlavní problémy při převodu jednoho formátu do druhého. Nejprve porovnám základní strukturu dokumentu a definici rozvržení stránek v obou formátech. Poté rozdělím formátovací vlastnosti na odstavcové a textové (stejně jako jsem o nich psal v částech 4.4 a 4.5) a porovnám tyto vlastnosti v obou formátech. Nakonec budu porovnávat mapování jednotlivých struktur (elementů) formátovacích objektů do formátu WordML.

Na začátku této kapitoly bych chtěl upozornit na hlavní rozdíl mezi oběma formáty. Tím je, že formátovací objekty nepoužívají mnoho elementů, používají hlavně atributy k definování všech možných vlastností. Naproti tomu formát WordML používá elementy téměř ke všemu, k definování struktur dokumentu i k definování vlastností. Už tato rozdílná struktura může způsobit potíže při převodu jednoho formátu do druhého.

5.1 Porovnání základní struktury dokumentu

Jak už bylo řečeno výše, dokument s formátovacími objekty nejprve definuje všechny možné rozvržení stránek a až poté následuje samotný obsah dokumentu, odkazující se na tato rozvržení. Naproti tomu formát WordML definuje rozvržení stránek až za vlastním obsahem dokumentu a to tak, že se každá definice vztahuje právě k předešlému obsahu. To by však neměl být při převodu zásadní problém. Podívejme se však na vlastní strukturu definice rozvržení stránek.

Při definici *fo:simple-page-master* se na sebe přesně mapují výška a šířka stránky a levý i pravý okraj stránky. Tedy až na jednotky, jejichž převod se bude muset zajistit[4]. Úplně přesně si též odpovídají atribut *fo:writing-mode* a element *w:textFlow*, které určují směr toku textu. Horní a dolní okraj na stránce definovaný pomocí formátovacích objektů však neodpovídá hornímu a dolnímu okraji formátu WordML. Formát WordML totiž do těchto okrajů zahrnuje i záhlaví a zápatí. Platí tudíž, že *w:top* = *margin-top* + *extent* (*fo:region-before*) a *w:header* = *margin-top*. Pro dolní okraj i zápatí platí obdobný vzorec.

Formátovací objekty umožňují definovat pouze počet sloupců a velikost mezi nimi, kdežto formát WordML umožňuje definovat i další vlastnosti sloupců. Nicméně atributy *column-count* a *column-gap* elementu *fo:region-body* se přesně shodují s atributy *w:num* a *w:space* elementu *w:cols*.

Formát WordML však neumožňuje definovat regiony start a end, dále neumožňuje definovat vzdálenost textu od záhlaví a zápatí. Naopak formátovací objekty neumožňují definovat obdobu atributu *w:gutter*.

[4] O jednotkách se dále nebudu zmiňovat, platí ale, že většina hodnot používajících jednotky se bude muset převádět.

Formát WordML umožňuje v sekci definující rozvržení stránek ještě definovat obsah záhlaví a zápatí. Formátovací objekty je umožňují definovat až v obsahu dokumentu v elementu *fo:static-content*. To by však při převodu neměl být zásadní problém. Problémem však může být, že formát WordML umožňuje v atributu *w:type* určit, na jakých stránkách se má dané záhlaví objevit. Formátovací objekty to sice také umožňují, ale dosti složitým způsobem. Umožňují to pomocí stanovení sekvence stránek elementem *fo:repeatable-page-master-alternatives*, který použije rozvržení stránky s různě pojmenovanými regiony záhlaví či zápatí. Stanovení sekvence stránek však znamená při převodu zásadní problém. Jelikož předem není známo, jestli text spadne na lichou nebo sudou stránku, nemůžeme tyto alternativy převést do formátu WordML.

Dále se v této sekci dá definovat počáteční číslo stránky a formát čísel stránky. Atribut *initial-page-number* téměř přesně odpovídá atributu *w:start* elementu *w:pgNumType* a atribut *format* se dá přibližně namapovat na atribut *w:fmt* téhož elementu po úpravě jeho hodnot. Co se však opět převést nedá, je atribut *force-page-count*, jelikož říká, kde má končit daná sekvence stránek.

Na závěr tedy můžeme říci, že obsah elementu *fo:simple-page-master* se dá víceméně převést do elementu *w:sectPr*, element *fo:page-sequence-master* se však převést nedá.

Existuje ještě jeden rozpor týkající se struktury dokumentu. Tímto rozporem je, že Word používá styly k definování vlastností části dokumentu. Tyto styly se mohou definovat všechny najednou na začátku dokumentu, což by ale samozřejmě byl problém, jelikož formátovací objekty žádné styly nepoužívají. Další možností je definovat styly za běhu tvorby dokumentu v samotném těle dokumentu pomocí elementu *w:cfChunk*. To už by nemusel být tak velký problém. Nejjednodušším řešením by však bylo nepoužívat styly vůbec a definovat pouze holé vlastnosti odstavce a textu. To by však zase uživatelům znesnadnilo změnu formátování většího dokumentu.

5.2 Mapování vlastností formátovacích objektů do vlastností odstavců

V této části se pokusím najít odpovídající vlastnosti formátovacích objektů k vlastnostem odstavce, které umožňuje definovat formát WordML.

Začněme vlastnostmi týkajícími se zákazu zalomení stránky. Elementu *w:keepNext* odpovídá atribut *keep-with-next*. Atribut *keep-with-next* však může nabývat číselných hodnot, které říkají, jak moc úsilí se má věnovat tomu, aby se obsah tohoto odstavce a následujícího odstavce objevil na stejné stránce. Kromě toho může nabývat hodnoty always znamenající maximální úsilí a auto znamenající žádné úsilí. Jelikož element *w:keepNext* je pouze dvouhodnotový, mělo by se jakékoliv nenulové číslo i hodnota always převést na hodnotu on, všechny ostatní hodnoty na hodnotu off. Problémem však je, že formátovací objekty umožňují definovat i atribut *keep-with-previous*, který požaduje umístění aktuálního odstavce s předcházejícím na stejné stránce. To znamená, že se pro každý zpracovávaný odstavec musíme podívat i na následující odstavec a jestliže ten obsahuje *keep-with-previous*, musíme pro aktuální odstavec vložit vlastnost *w:keepNext*. Elementu *w:keepLines* odpovídá atribut *keep-together* přesně opět až na hodnoty.

Další vlastnosti přikazují určité zalomení. Elementu *w:pageBreakBefore* odpovídá atribut *break-before* s hodnotou page. Tento atribut však může ještě nabývat dalších hodnot, které umožňují též vložit zalomení sloupce, zalomení na lichou stránku nebo na sudou stránku. Toto formát WordML neumožňuje definovat přímo ve vlastnostech odstavce, nýbrž pouze pomocí elementu *w:type*, který je potomkem elementu *w:sectPr*. To ale znamená problém, protože se zde musí určit též celé rozvržení stránky a to může způsobit zbytečné opakování informací o tomto rozvržení. Navíc se toto zalomení týká celé sekce, která je ohraničena pouze jiným předchozím elementem *w:sectPr*, což je jiné ohraničení, než ve formátovacích objektech. A ještě navíc formátovací objekty mohou používat i atribut *break-after*, který by spíše odpovídal elementu *w:type*.

Další vlastností je určení minimálního počtu osamocených řádků odstavce na konci nebo na začátku stránky. Formátovací objekty pro to používají atributy *orphans* a *widows* s číselnými hodnotami, kdežto formát WordML pro to používá pouze jeden element *w:widowControl*, který při hodnotě on odpovídá tomu, že oba atributy jsou nastaveny na 1. Z toho vyplývá, že by se element *w:widowControl* měl zapnout ve všech případech kromě toho, kdy jsou oba výše uvedené atributy nastaveny na 0.

Dalšími vlastnostmi jsou vlastnosti určující ohraničení odstavce. Formátovací objekty k tomu používají atributy ve tvaru border-strana-vlastnost, kde strany mohou být top, bottom, right, left, before, after, start a end a vlastnost může být color, width nebo style. Těmto atributům odpovídá element *w:pBdr*, kde strana ohraničení je určena jeho potomky *w:top*, *w:bottom*, *w:right* a *w:left* a vlastnost je určena atributy těchto potomků *w:color*, *w:sz* a *w:val*. První čtyři strany se mapují přesně, u dalších čtyřech stranách záleží na směru toku textu. Vlastnosti se mapují též přesně, ale mají jiné hodnoty. Barva si odpovídá nejpřesněji, jelikož ji oba formáty definují stejně pomocí hexadecimálního čísla ve tvaru RRGGBB. Převést na tento tvar se tedy musí pouze předdefinované barvy a funkce rgb(). Šířka ohraničení se ve formátovacích objektech zadává číselně s jednotkami nebo pomocí tří klíčových slov thin, medium nebo thick, jejichž šířku můžeme napevno definovat. Musí se tedy pouze převést všechna čísla na hodnotu v bodech. Styl ohraničení se v obou formátech stanoví pomocí klíčových slov, přičemž formát WordML umožňuje těchto stylů stanovit opravdu hodně. Nevím však, jestli pro všechny styly z formátovacích objektů existují odpovídající styly formátu WordML. Formátovací objekty ještě umožňují určit velikost mezery mezi textem a ohraničením pomocí atributu ve tvaru padding-strana. Tomuto atributu přesně odpovídá další atribut *w:space*. Ve formátovacích objektech ještě můžeme určit zkráceně ohraničení ve tvaru border-vlastnost, což znamená ohraničení na všech čtyřech stranách. Tomu odpovídá element *w:bar*, který je potomkem elementu *w:pBdr*. Nakonec ještě zmíním, že pro ohraničení celé stránky Word používá zvláštní element *w:pgBorders*, který však má stejnou strukturu (až na pár vyjímek) jako element *w:pBdr*.

Další skupinou vlastností jsou vlastnosti týkající se pozadí. Jedinou vlastností, která se zde dá převést je *background-color*, která se přesně mapuje na element *w:shd* a jeho atribut *w:fill*. Opět existuje ve Wordu zvláštní element pro pozadí celého dokumentu *w:bgPict*, který může obsahovat

opět buď barvu nebo i obrázek. Formátovací objekty definují obrázek pozadí pomocí atributu *background-image*, jehož hodnotou je adresa URI.

Dalšími vlastnostmi jsou mezery mezi odstavci a mezi řádky v odstavci. Mezery mezi řádky v odstavci se dají ve formátovacích objektech určit atributem *line-height* a to i v rozmezí. Hodnotou může být normal (mezera mezi řádky je ponechána na formátovači), dále číslo s jednotkami, číslo bez jednotek nebo procento (poslední dvě nastaví mezeru mezi řádky na násobek či procento velikosti použitého fontu). Tento element se mapuje na atribut *w:line* elementu *w:spacing*, přičemž se všechny jeho možné hodnoty musí převést na twipsy. Jestliže je určeno pouze minimum, nastaví se hodnota dalšího atributu *w:line-rule* na at-least, jinak na exact. Ostatní atributy upravující velikost mezery mezi řádky se do Wordu nedají převést. K určení mezery mezi odstavci používají formátovací objekty atributy *space-before* a *space-after* (může být určeno rozmezí) nebo jim odpovídající atributy *margin-top* a *margin-bottom*. Tyto atributy se přesně mapují do atributů *w:before* a *w:after* elementu *w:spacing* až na jednotky.

Složitější na převod bude odsazení jednotlivých řádků odstavce. Formátovací objekty k tomu používají atributy *start-indent*, *end-indent*, *text-indent* a *last-line-end-indent*. První dva atributu určují odsazení (kladná hodnota) či předsazení (záporná hodnota) všech řádků odstavce zleva a zprava. Třetí atribut určuje odsazení či předsazení prvního řádku odstavce zleva a poslední atribut odsazení či předsazení posledního řádku odstavce zprava. Celkové odsazení první řádky se získá součtem *start-indent* + *text-indent*. Z toho vyplývá, že atributy *start-indent* a *end-indent* odpovídají atributům *w:left* a *w:right* elementu *w:ind* (za předpokladu toku textu zprava doleva). Dále kladná hodnota atributu *text-indent* odpovídá atributu *w:first-line* a naopak záporná hodnota tohoto atributu odpovídá atributu *w:hanging*. Pro atribut *last-line-end-indent* neexistuje ve Wordu žádný podobný.

Poslední vlastností je zarovnání textu v odstavci. Formátovací objekty pro to používají atributy *text-align* a *text-align-last*. První určí zarovnání všech řádků odstavce kromě posledního a druhý určí zarovnání posledního řádku odstavce. Hodnoty jsou obvyklé left, right, center, justify, ale také start a end a inside a outside. Poslední dvě hodnoty znamenají zarovnání k vnitřnímu nebo vnějšímu okraji listu knihy. Atributu *text-align* odpovídá atribut *w:val* elementu *w:jc* s tím, že místo hodnoty justify nabývá hodnoty both, hodnoty start a end se převedou podle směru toku textu a pro hodnoty inside a outside neexistuje alternativa. Zarovnání poslední řádky se nedá do Wordu převést, jelikož je upravuje automaticky.

5.3 Mapování vlastností formátovacích objektů do vlastností textu

V této části se pokusím najít odpovídající vlastnosti formátovacích objektů k vlastnostem textu, které umožňuje definovat formát WordML.

Začněme nastavením fontu. Formátovací objekty nastavují font pomocí atributu *font-family*, jehož hodnotou je seznam názvů fontů nebo typů fontů seřazených dle priority. Název fontu tedy odpovídá určitému fontu, který se ve Wordu zadává pomocí elementu *w:rFonts* některým z jeho atributů. Typ

fontu může nabývat hodnot serif, sans-serif, cursive, fantasy nebo monospace. Převod tedy nebude jednoduchý, jelikož Word nepodporuje prioritu ani určení typu fontu.

Pokračujme stanovením velikosti písma. Pomocí formátovacích objektů můžeme nastavit velikost písma atributem *font-size*. Tento atribut může nabývat absolutní hodnoty (klíčová slova xx-small až xx-large), relativní hodnoty (larger, smaller) nebo procenta vztahující se k rodičům a nakonec též absolutní hodnoty s jednotkami. Tento atribut se mapuje na atribut *w:val* elementu *w:sz*, musíme však všechny hodnoty převést na poloviny bodů.

Barva písma se dále ve formátovacích objektech stanoví pomocí atributu *color*. Může nabývat hodnot ve tvaru #RRGGBB nebo 16 předdefinovaných hodnot (např. black nebo blue). Skoro přesně tomuto atributu odpovídá atribut *w:val* elementu *w:color*. Musíme akorát odstranit znak # a převést názvy předdefinovaných barev.

Ve formátovacích objektech existuje atribut *font-weight*, kterým nastavujeme tloušťku fontu. Může nabývat hodnot normal, bold, lighter, bolder a dále hodnot 100, 200, ... , 900. Hodnota normal odpovídá číslu 400, hodnota bold číslu 700. Hodnoty lighter a bolder jsou relativní hodnoty vztahující se ke svým předchůdcům. Jelikož Word dovoluje pouze zapnout či vypnou tučné písmo pomocí elementu *w:b*, všechny hodnoty 400 a nižší by měly vypnout tento element, ostatní hodnoty naopak zapnout.

Kromě tloušťky textu můžeme ještě stanovit horizontální velikost písma. Formátovací objekty pro to používají atribut *font-stretch*. Atribut může nabývat hodnot několika klíčových slov, která jsou seřazená právě podle horizontální velikosti písma, přičemž hodnota normal znamená neupravená hodnota. Dále též může nabývat dvou klíčových slov znamenajících relativní hodnotu k dříve použité horizontální velikosti písma. Tento atribut se mapuje na element *w:w*, jehož hodnotou jsou však procenta (1 % až 600 %).

Ve formátovacích objektech dále můžeme pomocí atributu *font-style* určit podobu daného fontu. Atribut může nabývat hodnot normal, italic, oblique nebo backslant. Word dovoluje pouze zapnout či vypnout psaní kurzívou elementem *w:i*. Myslím, že hodnota normal atributu *font-style* by měla kurzívu vypnout, ostatní hodnoty naopak zapnout.

Další vlastností je úprava písmen textu. Formátovací objekty k tomu používají atributy *font-variant* a *text-transform*. První atribut může nabývat hodnot normal nebo small-caps, přičemž hodnota small-caps znamená nahrazení malých písmen trochu zmenšenými velkými písmeny. To znamená, že se tento atribut přesně mapuje na element *w:smallCaps*. Druhý atribut může nabývat hodnot capitalize, uppercase nebo lowercase, přičemž první hodnota nahradí každé první písmeno slova velkým písmenem, druhá hodnota převede všechna písmena na velká a třetí hodnota naopak na malá písmena. Do Wordu se dá tedy převést pouze hodnota uppercase tohoto atributu, která se přímo mapuje na zapnutý element *w:caps*. Ostatní hodnoty by se musely převést změnou převáděného textu.

Dalšími vlastnostmi jsou různé dekorace textu. Formátovací objekty pro to používají atribut *text-decoration*. Tento atribut může nabývat hodnot underline, overline, line-through nebo blink, které

znamenají podtržený text, čára nad textem, přeškrtnutý text a blikající text. Barva dekorace se určí pomocí atributu *color*, jestliže je v potomcích změněna barva, barva dekorace by měla zůstat stejná. Zda se dekorace týká i mezer mezi slovy se dá dále stanovit pomocí atributu *score-spaces*. Jeho hodnota true znamená dekoraci i přes mezery, hodnota false dekoraci pouze slov. Atribut *text-decoration* s hodnotou line-through se mapuje na zapnutý element *w:strike* avšak s tím, že se nedá určit barva přeškrtnutí (ta je stejná, jako barva textu) ani zda se má přeškrtnout i mezera mezi slovy (ta je standardně přeškrtnuta). Hodnota underline tohoto elementu se mapuje na element *w:u*. Jestliže hodnota atributu *score-spaces* je true, hodnota atributu *w:val* elementu *w:u* bude single, jinak tato hodnota bude words. Barva podtržení se určí jak bylo popsáno výše a mapuje se do atributu *w:color* elementu *w:u*. Ostatní dekorace textu nelze převést.

Formátovací objekty mohou dále pomocí atributu *text-shadow* určit stínování jednotlivých písmen, a to jeho velikost i barvu. Nicméně Word umožňuje pouze zapnutí či vypnutí stínování elementem *w:shadow*. Jestliže je tedy určeno jakékoliv stínování pomocí formátovacích objektů, mělo by se ve Wordu zapnout (bude mít však předem určenou velikost a stejnou barvu jako text).

Další vlastností je mezera mezi písmeny. Ve formátovacích objektech ji můžeme upravit proti normálu použitím atributu *letter-spacing*. Atribut může nabývat kladných i záporných hodnot a přesně až na jednotky mu odpovídá element *w:spacing*. Ve formátovacích objektech též můžeme upravovat mezeru mezi slovy a to pomocí atributu *word-spacing*, nicméně pro tento atribut neexistuje ve Wordu alternativa.

Další vlastností je vertikální posun textu vzhledem k normálu. Formátovací objekty pro to definovaly několik atributů, z nichž nejpoužívanější je shrnující atribut *vertical-align*. Tento atribut může nabývat několika hodnot sestávajících z klíčových slov, dále též procentuální hodnoty (vztahující se k výšce řádku) a nakonec též absolutní hodnoty posunu. Procentuální a absolutní hodnoty a dále též hodnoty sub a super odpovídají víceméně elementu *w:position*, který určuje vertikální posun v polovinách bodů. Pro ostatní hodnoty neexistuje ve Wordu alternativa.

Posledními vlastnostmi jsou ohraničení a stínování, které se však mapují úplně stejně jako v odstavci až na to, že ohraničení zde může být buď na všech stranách nebo nikde a nedá se zde nastavit velikost mezery mezi textem a ohraničením..

5.4 Mapování odstavců

Nejzákladnější částí dokumentu, kterou potřebujeme převést, jsou odstavce. Formátovací objekty pro ně používají element *fo:block*. Součástí tohoto elementu jsou atributy určující nejen vlastnosti daného odstavce, ale i vlastnosti daného textu v odstavci. To znamená, že pro každý element *fo:block* musíme vytvořit k němu odpovídající element *w:p*. Dále musíme pro vlastnosti týkající se odstavce vytvořit element *w:pPr* a převést do něj tyto vlastnosti. Dále musíme vytvořit element *w:r* a jestliže element *fo:block* obsahuje i vlastnosti textu musíme též vytvořit element w:rPr a převést do něj tyto vlastnosti.

Další převod závisí na obsahu elementu *fo:block*. Jestliže například obsahuje text, vytvoříme pro něj obalovací element *w:t*, jestliže obsahuje obrázek, vytvoříme naopak obalovací element pro obrázek. Jestliže však uvnitř elementu *fo:block* narazíme na jiný element *fo:block* nebo na element *fo:inline* a tento element obsahuje pouze vlastnosti textu musíme vytvořit nové elementy *w:r* a *w:rPr*. Problémem však bude, jestliže tyto vnořené elementy budou obsahovat i vlastnosti odstavců. Jestliže před takovým vnořeným elementem není žádný text, můžeme vlastnosti odstavců přidat do vlastností aktuálního odstavce. V ostatních případech budeme muset pravděpodobně ukončit aktuální odstavec a vytvořit nový odstavec a tyto vlastnosti vložit do něj.

Existuje ještě jedna výjimka. Tou je, když je potomkem elementu *fo:block* tabulka. V tom případě vůbec nevytvoříme element *w:p*, místo něj vytvoříme rovnou obalovací element tabulky *w:tbl*. Jak se postaráme o převod atributů elementu *fo:block* popíšu v části popisující mapování tabulek.

5.5 Mapování vodících linek

Vodící linky vytvářené ve formátovacích objektech pomocí elementu *fo:leader* nepůjdou pravděpodobně do Wordu převést. Zdánlivě by mohlo vypadat, že se můžou namapovat na tabulátory, ale není to pravda. Délka vodící linky je totiž ve formátovacích objektech určena buď pomocí atributu *leader-length* nebo závisí na zarovnání daného odstavce. Naproti tomu tabulátor ve Wordu znamená nastavení zarážky na určité místo stránky v šířce stránky a linka se vytvoří od místa, kde právě skončil text až k dané zarážce. A to činí při převodu nepřekonatelný problém, jelikož nemůžeme nijak zjistit, kam danou zarážku nastavit (ledaže bychom to nějak složitě vypočítali).

5.6 Mapování seznamů

Jak již bylo řečeno výše, formátovací objekty používají pro vytvoření seznamu obalovací element *fo:list-block*, který se skládá z několika položek, z nichž každá se skládá z odrážky a z textu. Jak víme, Word požaduje každý seznam nejprve definovat v elementu *w:lists*. Mimo jiné se tam definuje i podoba odrážky nebo tvar číslování. U odrážky, pokud je v seznamu všude stejná, není žádný problém. Problém však nastane jestliže se jedná o číslovaný seznam nebo pokud je odrážka různá. Pokud se jedná o číslovaný seznam, můžeme ještě otestovat o jaký typ číslování se jedná a podle toho nastavit v elementu w:nfc kód daného typu číslování. Pokud je však odrážka různá na stejném stupni seznamu, tak to ve Wordu nelze nastavit (ledaže bychom pro každý řádek seznamu vytvořili novou definici nebo změnu původní definice seznamu).

Jak se na sebe tedy budou mapovat dané seznamy v obou formátech? Pro každý element *fo:list-block*, který bude obsahem následujícího odstavce, musíme vytvořit před tímto odstavcem element *w:cfChunks* a v něm definici daného seznamu v elementu *w:lists*. V této definici se obsah elementu *fo:list-item-label* vloží do atributu *w:val* elementu *w:lvlText* (pokud není zjištěno, že se jedná o číslovaný seznam). Pokud jsou definovány některé textové či odstavcové vlastnosti odrážky, vloží se dále do elementů *w:rPr* a *w:pPr*. Do odstavcových vlastností se zvlášť ještě musí převést vzdálenosti

odrážky a textu od levého okraje stránky. Vzdálenost odrážky se určí pomocí elementu *w:ind* a odpovídá mu hodnota atributu *start-indent* elementu *fo:list-item-label*. Vzdálenost textu se určí pomocí definovaného tabulátoru a jeho atributu *w:pos*, který odpovídá hodnotě *start-indent* (*fo:list-item-label*) + *provisional-distance-between-starts* (*fo:list-block*).

Dále musíme vytvořit samotný odstavec pro každý element *fo:list-item*, do jehož odstavcových vlastností zadáme odkaz na danou definici seznamu. Jestliže existují nějaké textové vlastnosti textu seznamu, vytvoříme též element *w:rPr* a dané vlastnosti do něj vložíme. Nakonec převedeme textový obsah elementu *fo:list-item-body* do elementu *w:t*.

Na závěr bych shrnul, že při převodu seznamu mohou nastat problémy s číslovanými a s vícestupňovými seznamy, neměly by však znamenat nepřekonatelný problém.

5.7 Mapování tabulek

Formátovací objekty umožňují definici tabulky i s popiskem, formát WordML pouze bez popisku. Popisek by se možná mohl vytvořit uměle jako samostatný odstavec. Nicméně převod samotné tabulky začíná mapováním elementu *fo:table* na element *w:tbl*. Většina vlastností elementu *fo:table* se převede do potomků elementu *w:tblPr*. Všechny sloupce definované elementem *fo:table-column* se převedou na elementy *w:gridCol*, které budou potomky elementu *w:tblGrid*. Problémem však je, že Word v tomto elementu umožňuje definovat pouze šířku sloupce, kdežto formátovací objekty i některé jiné vlastnosti. Dále se elementy *fo:table-header* a *fo:table-footer* mapují na první a poslední řádek tabulky (element *w:tr*), všechny ostatní řádky tabulky definované v elementu *fo:table-body* se musí vložit mezi hlavičku a patičku tabulky. Navíc jestliže element *fo:table* obsahuje atribut *table-omit-header-at-break* s hodnotou false, musí být v definici hlavičky tabulky v elementu *w:trPr* zapnut element *w:tblHeader*. Dále element *fo:table-cell* se mapuje na element *w:tc* a obsah elementu *fo:table-cell* se mapuje na odstavec.

To by bylo vše k mapování struktury tabulky, teď se podívejme na mapování vlastností. Začněme vlastnostmi elementu *fo:table*, z nichž se většina převede do elementu *w:tblPr*. Celkem bez problémů by se měly převést obecné vlastnosti ohraničení a pozadí (tak jak už o nich bylo zmíněno) až na to, že pro ohraničení tabulky existují ve formátovacích objektech dva modely určené atributem *border-collapse*. Hodnota separate znamená, že se musí určit ohraničení pro každou buňku zvlášť a mezi toto ohraničení se dá vložit mezera pomocí atributu *border-separation*. Hodnota collapse znamená, že se dají ohraničovat i celé sloupce a řádky tabulky, přičemž ohraničení je vycentrované na mřížce mezi buňkami tabulky. Word umožňuje definovat ohraničení zde na úrovni celé tabulky (levé, pravé, horní, dolní, ostatní vertikální a ostatní horizontální) nebo na úrovni řádků v elementu w:tblPrEx nebo na úrovni buněk v elementu *w:tcPr*. Z úrovně elementu fo:table se tedy ohraničení převede pouze na levé, pravé, horní a dolní ohraničení tabulky. Atribut *border-separation* se dále mapuje na element *w:tblCellSpacing* až na to, že pro atribut *border-separation* existují dvě hodnoty – jedna pro mezeru vertikální a druhá pro mezeru horizontální. Ve Wordu však daná mezera může mít pouze jednu

hodnotu. Dále zde můžeme určit vnitřní mezeru mezi textem a ohraničením pomocí atributů ve tvaru *padding-strana*, kde strany mohou být left, right, top, bottom, start, end, before, after. Tyto atributy se však na úrovni elementu *fo:table* vztahují k mezeře mezi vlastním obsahem tabulky a jejím ohraničením. To znamená, že pro ně na této úrovni ve Wordu neexistuje alternativa. V elementu *fo:table* dále můžeme určit například pomocí atributů *space-before* a *space-after* mezery mezi tabulkou a dalšími nebo předchozími částmi dokumentu. Do Wordu to však půjde převést pouze nepřímo a to vložením prázdného odstavce před nebo za tabulku s danou definovanou mezerou. Stejně tak se budou muset převést další odstavcové vlastnosti, které se zde mohou vyskytnout. Dále můžeme pomocí atributů *height* a *width* určit výšku a šířku tabulky. Word však na této úrovni umožňuje určit pouze šířku. Atribut *width* se tak poměrně přesně mapuje na element *w:tblW*. A nakonec atribut *table-layout* se přesně mapuje na element *w:tblLayout*, určující zda se má rozvržení tabulky přesně dodržet nebo jestli se má upravovat v závislosti na délce textu v buňkách.

Pokračujme vlastnostmi elementu *fo:table-column*. Jediná vlastnost *column-width* se převede na atribut *w:w* elementu *w:gridCol*. Navíc zde však můžeme definovat ještě několik dalších vlastností. Pro vlastnosti pozadí pro sloupec neexistuje ve Wordu alternativa, proto se jejich převodem nebudu zabývat. Vlastnosti ohraničení pro samostatný sloupec se též ve Wordu nedají určit, můžeme však definovat ve vlastnostech celé tabulky všechny vertikální hranice. To znamená, jestliže je určeno stejné ohraničení pro všechny sloupce můžeme toto ohraničení převést na element *w:insideV*, který je potomkem elementu *w:tblBorders*. Jinak toto ohraničení budu ignorovat. Dále atributem *number-columns-repeated* můžeme určit, kolikrát se bude daný sloupec opakovat. To znamená, že musíme pro takovýto sloupec vytvořit tolik odpovídajících elementů *w:gridCol*, kolik udává daný atribut. Dále zde můžeme též určit pomocí atributu *number-columns-spanned*, kolik sloupců má být spojeno dohromady. Tento atribut by se velmi těžko převáděl, proto ho převedu pouze na úrovni buněk.

Jestliže by měl element *fo:table-body* nějaké atributy, všechny se převedou stejně jako atributy elementu *fo:table* do elementu *w:tblPr*.

Pokračujme tedy vlastnostmi elementu *fo:table-row*. Vlastnost *height* tohoto elementu se mapuje na element *w:trHeight*, který je potomkem elementu *w:trPr*. Odstavcové vlastnosti tohoto elementu se nedají do Wordu převést. Vlastnosti pozadí také ne, jelikož ve formátovacích objektech definují pozadí všech buněk v řádku, kdežto na této úrovni ve Wordu definují výplň mezer v řádku mezi jednotlivými buňkami (což je ve formátovacích objektech dáno vlastnostmi pozadí elementu *fo:table*). Převést ale půjdou vlastnosti ohraničení popsaných výše a to do potomka *w:tblBorders* elementu *w:tblPrEx*.

Nakonec se podívejme na vlastnosti elementu *fo:table-cell*. Jednoduché by mělo být mapování obecných vlastností ohraničení, pozadí a vnitřních okrajů, tak jak již bylo popsáno výše, tentokrát na elementy *w:tcBorders*, *w:shd* a *w:tcMar*, které jsou potomky elementu *w:tcPr*. Dále se atribut *display-align*, který určuje vertikální zarovnání textu v buňce, přesně mapuje na element *w:vAlign*. Dále se atribut *width* určující šířku buňky mapuje na element *w:tcW*. Pro atribut *height* však neexistuje na této

úrovni tabulky alternativa. Poslední dva atributy *number-columns-spanned* a *number-rows-spanned* určují počet buněk spojených v horizontálním směru (ve sloupci) a ve vertikálním směru (v řádku). Spojené buňky přitom už nemusí být v definici tabulky uvedeny. Word pro toto používá elementy *w:hmerge* a *w:vmerge*, které jsou potomky elementu *w:tcPr*, přičemž jejich atribut *w:val* může nabývat hodnot restart nebo continue. Dané elementy se používají tak, že se na začátku spojování uvede daný element s hodnotou restart a dále se v buňkách, které chceme spojit s touto buňkou uvedou tyto elementy s hodnotou continue. Znamená to též, že musíme definovat i takto spojené prázdné buňky. Převod těchto atributů bude pravděpodobně dělat velké problémy, hlavně převod atributu *number-rows-spanned*.

5.8 Mapování poznámek pod čarou

Jak již víme, formátovací objekty využívají k vytvoření poznámky pod čarou element *fo:footnote*. Tento element se přímo mapuje na dvojici elementů *w:r* a *w:footnote* (se zapnutým atributem *w:suppressRef*), v jehož rámci ještě vytvoříme odstavec. Element *fo:inline*, který obsahuje odkaz na poznámku se převede do elementů *w:r* a *w:t*, které jsou potomkem odstavce v elementu *w:footnote*. Vlastnosti elementu *fo:inline* se převedou do elementu *w:rPr*. Dále se vlastnosti elementu *fo:block*, který je potomkem elementu *fo:footnote-body*, musí převést na odstavcové a textové vlastnosti tohoto odstavce. Pro obsah tohoto elementu se vytvoří další elementy *w:r* a *w:t*. Vynechá se však element *fo:inline*, který obsahuje odkaz na poznámku pod čarou, ten se vloží až za konec elementu *w:footnote* do elementu *w:t*. Jeho textové vlastnosti se však musí vložit již před element *w:footnote* do elementu *w:rPr*.

Jak víme, ve Wordu se dají definovat některé obecné vlastnosti poznámek pod čarou. Aby si převedené poznámky odpovídali, je třeba do vlastností elementu *w:footnotePr* v *w:sectPr* vložit element *w:pos* s hodnotou page-bottom.

Formátovací objekty dále umožňují definovat pomocí elementu *fo:static-content* a jeho atributu *flow-name* s hodnotou xsl-footnote-separator oddělovač poznámky pod čarou. Ve Wordu se dá definovat totéž, ale pouze pro celý dokument v elementu *w:footnotePr*, který je potomkem elementu *w:docPr*. Jelikož se v obou formátech tento oddělovač definuje jako odstavec, může se tato definice jako odstavec převést.

5.9 Mapování obrázků

Jelikož Word používá ke vkládání obrázků do dokumentu formát VML, bude převod elementů *fo:external-graphic* a *fo:instream-foreign-object* velmi obtížný. Proto se převodem elementu *fo:instream-foreign-object* nebudu vůbec zabývat a z elementu *fo:external-graphic* převedu pouze úplný základ.

Když tedy narazím na element *fo:external-graphic*, tak pro něj vytvořím elementy *w:r*, *w:pict* a *v:shape*. V atributu *style* elementu *v:shape* mohu upravit některé hodnoty podle atributů elementu

fo:external-graphic nebo jeho rodičovského elementu *fo:block*. Atributy týkající se mezer (např. *margin-top*) se převedou na stejnojmenné hodnoty atributu *style*. Atributy *content-width* a *content-height* se převedou na hodnoty height a width, problémem však je že si budou odpovídat pouze absolutní hodnoty (ostatní hodnoty se v obou formátech vztahují k něčemu jinému a nelze je na sebe převést). Atribut *text-align* se dále převede na hodnotu mso-position-horizontal. Pro atribut *src* se vytvoří element *v:imagedata* a jeho hodnota se mapuje na jeho stejnojmenný atribut.

5.10 Mapování odkazů

Mapování odkazů bude jednoduché. Element *fo:basic-link* se přímo mapuje na element *w:hlink*. Dále se atributy *external-destination* a *internal-destination* elementu *fo:basic-link* mapují přímo na atributy *w:dest* a *w:bookmark* elementu *w:hlink*. Ostatní atributy týkající se odkazu nemají ve Wordu alternativu.

Aby fungoval odkaz uvnitř dokumentu, musíme ještě převést každý atribut *id* z formátovacích objektů na element *aml:annotation*, který bude obalovat obsah elementu s atributem *id*. Hodnotu atributu *id* můžeme převzít jako identifikátor i pro formát WordML.

5.11 Čísla stránek, živá záhlaví

Formátovací objekty umožňují vkládat do dokumentu čísla stránek pomocí elementu *fo:page-number*, přičemž formátování těchto čísel se řídí hodnotou několika atributů elementu *fo:page-sequence* (o jejich mapování jsem se zmínil v části 5.1). Samotný element *fo:page-number* se mapuje na instrukci PAGE, tak jak byla popsána v části 4.12.

Formátovací objekty dále umožňují vložit do dokumentu číslo stránky, na které se nachází nějaký text. Umožňují to použitím elementu *fo:page-number-citation* a jeho atributu *ref-id* odkazujícího se na atribut *id*. Tento element se mapuje na instrukci PAGEREF následovanou jménem záložky (obsahem atributu *ref-id*). Jak se vytvoří v dokumentu záložka bylo popsáno v části 5.10.

Živá záhlaví se ve formátovacích objektech tvoří pomocí elementů *fo:marker* a *fo:retrieve-marker*. Ve Wordu se dají živá záhlaví vytvořit pomocí instrukce STYLEREF. Jak ale převedeme výše zmíněné elementy? Pro každý element *fo:marker* s různou hodnotou atributu *marker-class-name* musíme vytvořit styl, který bude použit v odstavci obsahujícím textovou část elementu *fo:marker*. Vlastnosti stylu se dají získat z následujícího elementu *fo:inline* nebo *fo:block*, jelikož bude pravděpodobně obsahovat stejný text jako element *fo:marker*. Styl pojmenujeme hodnotou atributu *marker-class-name*. Element *fo:marker* může obsahovat další elementy, které se potom převedou do formátování daného textu v záhlaví. Pomocí instrukce STYLEREF se však toto formátování nedá převést, dá se převést pouze samotný text. Element *fo:retrieve-marker* se mapuje na samotnou instrukci STYLEREF následovanou názvem stylu daným hodnotou atributu *retrieve-class-name*. Dále atribut *retrieve-position* s hodnotami last-starting-within-page nebo last-ending-within-page můžeme převést do Wordu zapnutím přepínače dané instrukce, který zapíná prohledávání stránky zdola nahoru.

6 Možnosti implementace

V zásadě existují dvě možnosti, jak implementovat převod formátovacích objektů do formátu WordML. První možností je použití stylového jazyka XSLT, který se sice používá hlavně na definici toho, jak se má dokument XML zobrazit na výstupu, ale děje se tak vlastně transformací daného XML dokumentu na jiný XML dokument, který už říká, jak zobrazit elementy a atributy primárního dokumentu. A to je vlastně přesně to, co potřebujeme.

Druhou možností je pak použití libovolného programovacího jazyka, ve kterém se dá použít XML parser [7], který sám zkontroluje správnou strukturovanost XML dokumentu a zpřístupní nám ho. Dalším požadavkem na daný programovací jazyk je existence rozhraní (API), které nám umožňuje dostatečně manipulovat s obsahem daného XML dokumentu. Takovým rozhraním je DOM (Document Object Model) [10], který reprezentuje XML dokument jako stromovou hierarchickou strukturu a obsahuje funkce, které nám umožňují celý strom procházet a manipulovat s jeho uzly.

Oba způsoby implementace mají své výhody i nevýhody. Jedinou nevýhodou jazyka XSLT je, že má slabší vyjadřovací možnosti oproti programovacím jazykům. Z toho vyplývá, že se zde řada věcí bude programovat hůře, co se této práce týče, zmíním například převod jednotek. Jinak má XSLT již samé výhody počínaje tím, že vlastně je přímo stvořeno pro transformace z jednoho XML formátu do jiného. Další výhodou je, že se dá snadno a rychle naučit, což se o programovacím jazyku říct nedá. Hlavní výhodou pak je jazyk XPath, který je součástí XSLT a který pomocí jednoduché syntaxe umožňuje vyhledat jakýkoliv element nebo skupinu elementů v XML dokumentu. Takto nalezený element nebo skupina elementů se dále zpracovává pomocí šablon. Odpovídající šablona se zavolá automaticky pro každý takto nalezený element. Šablony se dají také pojmenovat a mohou se tak používat i jako procedury v programovacím jazyku, dokonce se dají volat rekurzivně a mohou se jim předávat parametry. Jazyk XPath dále umožňuje používat i některé funkce pro práci s čísly, řetězci, nalezenými elementy a logické funkce. XSLT též obsahuje příkazy pro podmínku a pro cyklus, které trochu zmírní nevýhodu oproti programovacímu jazyku. Navíc existují rozšíření jazyka XSLT, ve kterých můžeme volat funkce napsané v programovacím jazyku.

Výhodou programovacích jazyků jsou tedy hlavně jejich silné vyjadřovací prostředky a nevýhodou je potom horší manipulace s XML daty. Co se této nevýhody týká, pro některé programovací jazyky již existují knihovny [9], které umožňují používat přímo jazyk XPath, čímž tato nevýhoda zaniká. Takovými jazyky jsou například Java, C, Perl nebo Python.

Já pro převod formátovacích objektů do formátu WordML využiji XSLT, jelikož si myslím, že jeho výhody převažují nad jeho nevýhodami a dále proto, že neumím žádný vhodný programovací jazyk.

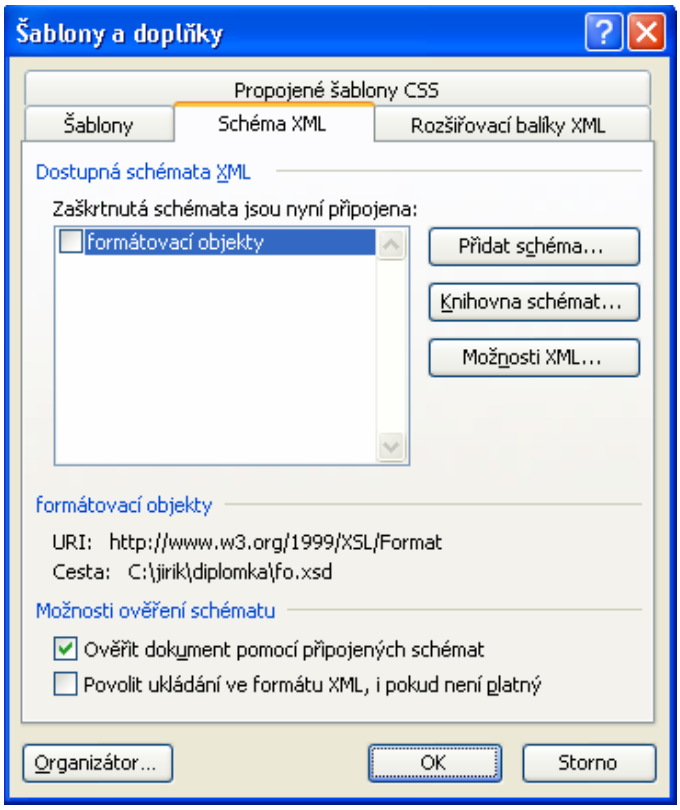
7 Integrace převodu do MS Word 2003

Další velkou výhodou použití XSLT je, že se tento převod dá přímo integrovat do MS Wordu 2003 [4], jelikož ten přímo umožňuje při otevření XML dokumentu použít automaticky styl XSLT a zobrazit výsledek.

A jak to vlastně funguje? Stačí vložit do Wordu XML schéma daného typu XML dokumentů, na které chceme automaticky použít styl XSLT a samozřejmě též styl samotný. Když pak ve Wordu otevíráme XML dokument, Word kontroluje, zda dokument není napsán ve stejném jmenném prostoru jako některé připojené XML schéma, a pokud ano a je k němu přiřazen styl, tak ho automaticky na daný dokument aplikuje a zobrazí pouze výsledek.

XML schéma[5] můžeme vložit do Wordu otevřením menu Nástroje a v něm otevření menu Šablony a doplňky. Zde klikneme na záložku Schéma XML jak je vidět na obrázku 6. Dále klikneme na záložku Přidej schéma a zadáme cestu k místu, kde se nachází. Do kolonky Alias můžeme zadat jméno schématu, tak, jak se bude při použití zobrazovat (jestliže ho nezadáme, bude se zobrazovat URI jmenného prostoru).

Obrázek 6



[5] XML schéma k formátovacím objketům se nachází na přiloženém CD v souboru fo.xsd, styl pro převod v souboru prevod.xsl. XML schéma bylo automaticky vygenerováno z DTD vytvořeném společností RenderX Inc. (www.renderx.com) použitím programu Dtd2Schema, který najdete na stránce www.syntext.com.

Dále musíme k tomuto schématu přiřadit styl XSLT, který se použije na dokumenty napsané podle tohoto schématu. Opět si otevřeme menu, které je vidět na obrázku 6 a klikneme na záložku Knihovna schémat. V kolonce schémata klikneme na dané schéma a dále klikneme na záložku Přidat řešení, kde zadáme cestu k místu, kde se nachází daný styl XSLT. V kolonce Alias opět zadáme pojmenování, které se bude zobrazovat při jeho použití.

Od této chvíle se již při otevření souboru, který používá pro formátování formátovací objekty zobrazí přímo formátovaný obsah dokumentu.

8 Převod formátovacích objektů do formátu WordML

V této kapitole popíšu strukturu XSLT stylu, který převede dokument napsaný pomocí formátovacích objektů do dokumentu ve formátu WordML. Dále zmíním hlavní problémy, které při převodu nastaly a jak jsem je vyřešil.

Předem upozorňuji, že jsem zdaleka nepřevedl všechny formátovací objekty ani jejich vlastnosti a že převod není úplně do detailu přesný[6]. Důvodem je velká složitost a množství formátovacích objektů a jejich vlastností a též částečná nekompatibilita mezi oběma formáty. Nicméně myslím, že to na ukázku možností převodu úplně stačí.

8.1 Šablona odpovídající kořenovému uzlu

Tato šablona vytvoří základní strukturu dokumentu, tak jak byla popsaná ve druhém odstavci části 4.1. Nejprve jsou zde vytvořeny definice seznamů, pokud zpracovávaný dokument nějaký seznam obsahuje.

Dále je pro každý element *fo:marker* s různým atributem *marker-class-name* vytvořen styl, který je dále použit při tvorbě živých záhlaví. Jinak tvorbu stylů nepoužívám, všechny ostatní vlastnosti převádím přímo do elementů *w:pPr* a *w:rPr* v daném odstavci.

Dále jsem definoval dvě vlastnosti dokumentu, které ho přimějí zobrazit v módu „rozložení při tisku“ a ve velikosti 100 %.

Dále je vytvořeno tělo dokumentu elementem *w:body*, ze kterého jsou volány dvě šablony. První zpracovává všechny potomky elementů *fo:flow* a druhá zpracovává rozvržení poslední sekvence stránek (posledního elementu *fo:page-sequence*).

8.2 Šablony odpovídající elementům *fo:block*

Nejčastějším potomkem elementu *fo:flow* je element *fo:block*. Pro něj existují dvě šablony. První přesnější je volaná pro tento element, který je potomkem elementů *fo:flow*, *fo:static-content*, *fo:table-cell*, *fo:table-caption* nebo *fo:list-item-body* (jinými slovy potomkem elementů, které mohou obsahovat pouze blokové elementy). Druhá obecnější je volaná v ostatních případech (tzn. když je potomkem elementů, které mohou obsahovat i inline elementy a text).

Hlavním problémem při převodu tohoto elementu bylo, že ne pro každý element *fo:block* se může vytvořit element *w:p*. Element *w:p* se má vytvořit pouze, když je potomkem aktuálního elementu *fo:block* holý text nebo nějaký inline element. Proto jsem pro vytvoření počátečního a koncového tagu elementu *w:p* použil nestandardní funkci *xsl:text* s hodnotou atributu *disable-output-escaping* nastavenou na yes, která dovolí na výstup zkopírovat doslovně znaky <,>.

[6] V příloze je uveden seznam všech formátovacích objektů a jejich vlastností a též do jaké míry jsou dané vlastnosti či objekty převedeny.

Jak tedy vypadá struktura přesnější šablony? Nejprve je otestováno, jestli je prvním potomkem aktuálního elementu nějaký inline element. Jestliže ano, je vytvořen odstavec pomocí elementu *w:p* a dále jsou volány šablony zpracovávající odstavcové vlastnosti.

Jestliže element obsahuje atribut *id*, je vytvořena záložka. Dále jsou volány šablony pro všechny elementy, které může aktuální element obsahovat (to znamená i šablona pro element *fo:block*).

Následuje test, zda posledním potomkem aktuálního elementu je text a vnořený test, který zkoumá, zda předposledním potomkem je blokový element. Jestliže jsou oba testy splněny, vytvoří se odstavec a volají se šablony pro odstavcové vlastnosti a dále nebo při splnění pouze prvního testu se volají šablony pro vlastnosti textu a vypíše se samotný text. Dále musíme ukončit odstavec, jestliže byl předposledním potomkem blokový element nebo jestliže byl prvním potomkem inline element a posledním nebyl element *fo:block*.

Na tomto místě bych rád zmínil, že s převodem samotného textu byly problémy. Když se totiž text převedl tak, jak byl v původním dokumentu, vznikly občas vícečetné mezery mezi slovy. V XSLT je naštěstí definovaná funkce normalize-space(), která odstraňuje vícečetné mezery mezi slovy, ale bohužel též na začátku a na konci textu. Tím zase mezi některými slovy nebyly mezery vůbec a tak jsem toto musel sám otestovat, aby to fungovalo správně.

Jak se od výše popsané šablony liší obecnější šablona? Za prvé je na jejím začátku volaná šablona se jménem „predchoziodstavec", jejímž úkolem je zpracovat text, který je potomkem stejného elementu jako aktuální element a který se nachází těsně před aktuálním elementem. Šablona tedy nejprve otestuje, zda je předcházející uzel textem, dále vytvoří odstavec a zavolá šablony pro odstavcové vlastnosti (pokud je druhým předchozím uzlem blokový element), dále zavolá šablony pro vlastnosti textu, vypíše daný text a uzavře odstavec pokud je předcházejícím uzlem text nebo inline element.

Za druhé se tato obecnější šablona liší trochu odlišným voláním šablon některý formátovacích vlastností.

8.3 Šablony pro odstavcové vlastnosti a vlastnosti textu

Nejprve pár slov obecně ke všem formátovacím vlastnostem. Při jejich převodu jsem striktně dodržoval dědičnost vlastností z rodičů na potomky. Výchozí hodnoty jsem většinou převádět nemusel, jelikož Word si je sám nastaví skoro vždy tak, že odpovídají specifikaci. Absolutní míry jsem též převedl všechny. Nicméně už nezbyl čas na převod relativních jednotek (pouze u atributů týkajících se šířky buňky, sloupce a tabulky jsem převedl i procenta), na převod rozmezí a z funkcí jsem převedl pouze pár významných.

Jak už jsem uvedl výše, šablony pro odstavcové vlastnosti se volají ze šablon pro element *fo:block* (některé též ze šablony pro tabulku). Jestliže se jedná o dědičné vlastnosti, tak se volá výskyt vlastnosti u nejbližšího předka, jestliže nikoliv, volá se výskyt vlastnosti pouze u aktuálního uzlu.

Existují však určité výjimky. Například atributy *space-before* a *space-after* se musí zavolat tak, aby se mezera vytvořila za odstavcem, za kterým má. Atribut *space-after* může totiž být určen například u elementu *fo:block*, který obsahuje další dva elementy *fo:block* a daná mezera se musí vložit do odstavcových vlastností posledního potomka tohoto elementu *fo:block*. Tyto vlastnosti se též musí při zpracování sčítat tak, aby za každým odstavcem byla odpovídající mezera.

Další výjimkou je atribut *break-after*, který se musí převádět na element *w:pageBreakBefore* následujícího odstavce nebo atributy *border-color*, *border-width* a *padding*, které se volají pouze, když je v aktuálním elementu přítomen element *border-style* (jelikož jeho počáteční hodnotou je žádné ohraničení) a to až ze šablony obsluhující tento element.

Šablony pro vlastnosti textu jsou volány ze šablon většiny formátovacích objektů a pro jejich volání platí stejná pravidla jako u šablon odstavcových vlastností, zde však nejsou žádné výjimky.

Jednotlivé šablony nebudu popisovat, jelikož jsou většinou velmi jednoduché a jak detailně jsou tyto vlastnosti převedeny je uvedeno v příloze.

Nakonec bych ještě uvedl, že většina těchto šablon používá pojmenované šablony, které zajišťují převod jednotek. Jedná se například o šablonu „prevodtwips", která převádí všechny možné absolutní míry na twipsy. Jestliže je název šablony doplněn písmenem „p", znamená to, že šablona používá parametry (například proto, že potřebujeme k převedenému číslu ještě něco přičíst).

8.4 Šablony týkající se rozvržení stránek

Nejprve je volána šablona pro element *fo:page-sequence* a to za prvé z místa, odkud jsou volány šablony pro odstavcové vlastnosti (a to pouze pokud se jedná o poslední odstavec elementu *fo:page-sequence* a existuje následující element *fo:page-sequence*) a za druhé ze šablony kořenového uzlu jako poslední potomek elementu *w:body* (ta se volá vždy).

Tato šablona pak volá další šablony týkající se rozvržení stránek. Za prvé zavolá šablonu elementu *fo:simple-page-master*, jehož atribut *master-name* je roven atributu *master-reference* aktuálního uzlu[7]. Dále zavolá šablonu elementu *fo:static-content* a nakonec ještě šablonu pro atribut *initial-page-number*.

Šablona pro element *fo:simple-page-master* dále volá šablony pro atributy týkající se rozměrů a okrajů stránky. Šablona pro element *fo:static-content* vytvoří záhlaví (jestliže atribut *flow-name* je roven xsl-region-before) nebo zápatí (jestliže atribut *flow-name* je roven xsl-region-after) a z něho volá šablonu pro element *fo:block*.

[7] Toto je jisté zjednodušení, jelikož ve skutečnosti by *master-name* mohl odkazovat též na element *fo:page-sequence-master*.

8.5 Šablony týkající se tabulek

Šablony pro elementy *fo:table-and-caption* a *fo:table* jsou volány ze šablony kořenového elementu a ze šablony elementu *fo:block*.

První zmíněná šablona volá šablony pro elementy *fo:table* a *fo:table-caption*. Jaká z těchto šablon se zavolá dříve, záleží na hodnotě atributu *caption-side* (pro hodnoty top a left se nejprve volá *fo:table-caption*). Ze šablony elementu *fo:table-caption* se volají šablony pro všechny blokové elementy.

Samotná tabulka se začíná vytvářet pomocí šablony elementu *fo:table*. Tato šablona nejprve zavolá pojmenovanou šablonu „predchoziodstavec", která byla popsána v části 8.2. Dále vytvoří prázdný neviditelný odstavec (docíleno nejmenším možným řádkováním), ze kterého jsou volány šablony pro vlastnosti týkající se mezer a zalomení stránky před tabulkou. Poté je vytvořena samotná tabulka elementem *w:tbl*. Dále jsou volány šablony týkající se vlastností celé tabulky. Tyto šablony nejsou nijak složité až na šablonu pro atribut *width* (je společnou šablonou pro šířku tabulky, sloupce a buňky), kterou proto jedinou popíšu později. Poté je volána šablona pro element *fo:table-column* a pak už šablony pro elementy *fo:table-header*, *fo:table-body* a *fo:table-footer*. Za tabulkou je pak ještě vytvořen další neviditelný odstavec, ze kterého jsou volány šablony pro vlastnosti týkající se mezer za tabulkou.

Jak vypadá šablona pro element *fo:table-column*? Tato šablona nejprve vytvoří proměnnou z atributu *number-columns-repeated* (počet sloupců, pro které platí jejich definice) a poté zavolá pojmenovanou šablonu „sloupec" s parametrem rovným výše zmíněné proměnné. Tato šablona pak vytvoří sloupec a zavolá šablonu pro atribut *column-width* a nakonec jestliže předaný parametr je větší než jedna zavolá znovu samu sebe s hodnotou parametru o jedna menším. Tím se vytvoří správný počet sloupců.

Elementy *fo:table-body*, *fo:table-header* a *fo:table-footer* mají společnou šablonu, jelikož mají stejné potomky. Nejprve se zde volá šablona pro element *fo:table-row*. Jestliže však tento potomek neexistuje, šablona musí vytvořit elementy definující řádek tabulky a pak zavolá šablonu pro element *fo:table-cell*.

Šablona pro element *fo:table-row* vytvoří řádek tabulky a volá šablony pro vlastnosti tohoto řádku (jsou to šablony pro vlastnosti týkající se ohraničení, výšky řádku a definice řádku jako hlavičky tabulky). Nakonec tato šablona zavolá šablonu pro element *fo:table-cell*.

Šablona pro element *fo:table-cell* je poněkud složitější, jelikož musí zajistit vertikální i horizontální slučování buněk tabulky. Aby to mohla splnit, je v ní na začátku definovaná řada proměnných. Proměnná „pocetsl" obsahuje počet sloupců tabulky. Jak byla tato proměnná vypočítaná? Jestliže tabulka obsahuje definici sloupců tabulky, tak byla zjištěna podle vzorce : počet elementů *fo:table-column* + součet hodnot všech atributů *number-columns-repeated* těchto elementů - počet těchto elementů obsahujících atribut *number-columns-repeated*. Jestliže tabulka neobsahuje definici sloupců,

tak je proměnná zjištěna podle tohoto vzorce : počet buněk v prvním řádku tabulky, který neobsahuje atribut *number-rows-spanned* + součet hodnot atributů *number-columns-spanned* těchto elementů - počet těchto elementů obsahujících tyto atributy.

Další proměnnou je proměnná „pom“, která obsahuje celkový počet řádků tabulky, které se nacházejí před právě zpracovávaným řádkem a které obsahují atribut *number-rows-spanned*. Tato proměnná je využita při výpočtu další proměnné „pozicep“, která obsahuje číslo nejbližšího řádku tabulky od právě zpracovávaného řádku, jehož buňka obsahuje atribut *number-rows-spanned*. Toto číslo hledáme v cyklu přes předchozí řádky tabulky a do proměnné vložíme pozici tohoto řádku, jestliže řádek obsahuje buňku s atributem *number-rows-spanned* a jestliže počet předchozích řádků tabulky, které obsahují atribut *number-rows-spanned* je roven proměnné „pom“ snížené o jedničku. A tuto proměnnou nakonec využijeme při výpočtu proměnné „pozicer“, ve které je uložena vzdálenost nejbližšího řádku s atributem *number-rows-spanned* od právě zpracovávaného řádku. Tuto proměnnou vypočítáme podle vzorce : počet předchozích řádků tabulky + 1 - „pozicep“.

Další proměnná „pozices“ obsahuje číslo sloupce obsahujícího atribut *number-rows-spanned* nejbližšího řádku. Toto číslo získáme použitím stejného cyklu a stejné podmínky jako u proměnné „pozicep“ a to podle vzorce : „pocetsl“ - počet následujících buněk za buňkou obsahující atribut *number-rows-spanned* - součet hodnot atributů *number-columns-spanned* těchto buněk + počet buněk obsahujících takový atribut.

Poslední proměnná „pocetr“ obsahuje počet řádků, které se mají spojit a je zjištěna právě z atributu *number-rows-spanned* nejbližšího řádku.

Po výpočtu těchto proměnných šablona testuje, zda počet buněk definovaných v řádku je menší než počet sloupců a zda „pocetr“ > „pozicer“ a zda „pozices“ je rovna pozici aktuální buňky. Jestliže je podmínka splněna, vytvoříme prázdnou buňku a nastavíme vertikální spojení na hodnotu continue a též zavoláme šablony pro vlastnosti vnitřních okrajů[8].

Dále vytvoříme právě zpracovávanou buňku a zavoláme šablony pro vlastnosti této buňky. Dále se volají šablony pro blokové elementy, které tvoří samotný obsah buňky. Jestliže buňka neobsahuje žádný text, musíme vytvořit prázdný odstavec. Po uzavření aktuální buňky šablona ještě testuje, zda aktuální buňka obsahuje atribut *number-columns-spanned*. Jestliže ano, tak volá pojmenovanou šablonu „hmerge“ s parametrem říkajícím, kolik sloučených buněk se má vytvořit. Samotná šablona „hmerge“ pak vytvoří horizontálně spojenou buňku a zavolá sama sebe tolikrát, kolik je hodnota daného parametru. Nakonec se ještě testuje, zda není náhodou poslední buňka vertikálně spojená a jestliže je tak se vytvoří.

[8] Vertikální spojování buněk nebude fungovat úplně obecně. Jestliže totiž řádek obsahuje více buněk s atributem *number-rows-spanned* muselo by se při výpočtu proměnné „pozices“ ještě nějak zajistit, aby se vypočítala pozice správné buňky.

Nakonec ještě popíšu výše zmiňovanou šablonu pro vlastnosti určující šířku tabulky, sloupců a buněk. Tato šablona si též nejprve vytváří několik proměnných. První z nich „twidth“ obsahuje hodnotu atributu *width* elementu *fo:table*, jestliže tento atribut není určen, nastaví se hodnota proměnné na hodnotu nil. Druhá proměnná „master“ obsahuje hodnotu atributu *master-reference* aktuálního elementu *fo:page-sequence*. Dále proměnná „pgwidth“ obsahuje šířku stránky pro aktuální sekvenci stránek. Proměnné „mlefttwips“ a „mrighttwips“ obsahují velikost levého a pravého okraje v twipsech pro aktuální sekvenci stránek. Proměnná „twidthtwips“ pak obsahuje šířku tabulky v twipsech. Jestliže proměnná „twidth“ je rovna nil nebo auto, tak se určí převodem proměnné „pgwidth“ na twipsy. Jestliže proměnná „twidth“ obsahuje procenta, tak se určí převodem daného procenta proměnné „pgwidth“ na twipsy. Jinak se určí převodem samotné proměnné „twidth“ na twipsy. Další proměnná „soucet“ vytvoří řetězec tvořený čísly vyextrahovanými z hodnoty atributu column-width, který obsahuje funkci proportional-column-width(číslo), oddělenými znaménkem +. Poslední proměnná „secteno“ pak pomocí rekurzivní šablony „scitani“ vrací součet těchto čísel, přičemž parametr „s“ je dosavadní součet a parametr „r“ zbývající řetězec k sečtení (ze řetězce se zleva postupně odebírají již přičtená čísla).

Po výpočtu proměnných pak následují testy, které podle rodičovského elementu a s použitím předpřipravených proměnných vrátí správnou šířku v twipsech.

8.6 Šablony týkající se seznamů

Jak už bylo dříve zmíněno, pro každý seznam v dokumentu musí existovat jeho definice. K vytvoření této definice slouží šablona pro element *fo:list-block* v módu „def“, která je volaná přímo z kořenové šablony jako potomek elementu w:lists. Navíc je volán pouze takový element *fo:list-block*, který nemá žádné stejnojmenné předky a to proto, že definice vícestupňového seznamu je vytvořena v jedné definici.

Z kořenové šablony je ještě volaná šablona pro element *fo:list-block* v módu „odk“, která pouze zajišťuje odkazování na danou definici (vysvětleno v části 5.6).

Jak tedy vypadá struktura šablony pro element *fo:list-block* v módu „def“? Nejprve je vytvořen element pro definici seznamu, kterému je přidělen identifikátor. Word požaduje, aby identifikátorem bylo celé číslo, proto nelze využít funkci generate-id(), jelikož tato funkce může generovat jako identifikátor cokoliv a navíc to záleží na dané implementaci. Já jsem tedy identifikátor stanovil jako počet předcházejících uzlů *fo:list-block* (před právě zpracovávaným uzlem) zvýšený o 1.

Šablona dále volá šablonu pro první element fo:list-item. Tato šablona definuje jednotlivé stupně seznamu (je volána rekurzivně pro každý vnořený element *fo:list-item*). Nejprve jsou definovány čtyři proměnné „si“, „pd“, „sitw“ a „pdtw“, které popořadě obsahují hodnoty vlastností *start-indent*, *provisional-distance-between-starts* a další dvě hodnoty stejných vlastností v twipsech. Šablona dále vytvoří stupeň seznamu a definuje text zarážky (zatím hotov pouze seznam o stejné textové zarážce).

Dále jsou definovány odstavcové a textové vlastnosti zarážky. Nakonec je volána tato šablona pro vnořené seznamy (další stupeň seznamu).

Dále je třeba vložit do odstavcových vlastností odstavce, který obsahuje seznam, odkaz na definici seznamu. V šabloně pro element *fo:block* je tedy u volání šablon odstavcových vlastností přidaná podmínka na existenci předka *fo:list-item-body* a když je splněna, je přidán odkaz na danou definici daného stupně seznamu.

Nakonec ještě existují šablony pro výskyt elementu *fo:list-block* a jeho potomků v těle dokumentu. Šablona pro element *fo:list-block* volá šablonu pro element *fo:list-item* a ta zase volá šablonu pro element *fo:list-item-body*. Z této šablony jsou už volány šablony pro všechny blokové elementy. Šablona *fo:list-block* je volaná z kořenové šablony, ze šablony *fo:block*, ze šablony *fo:table-cell* a ze šablony *fo:list-item-body*.

8.7 Šablona elementu fo:inline

Element *fo:inline* je používán k formátování textu v odstavci. Jeho šablona je proto volaná ze šablony *fo:block* a též ze šablon inline elementů, které mohou mít nějaké potomky.

Šablona nejprve volá pojmenovanou šablonu „predchozitext“, která vypíše správně naformátovaný textový uzel, nacházející se těsně před aktuálním uzlem *fo:inline*. Poté šablona volá šablony pro všechny inline elementy. Dále jestliže je jeho posledním potomkem text, tak volá pojmenovanou šablonu „vlastnostiip“, která volá šablony pro textové vlastnosti. Nakonec je tento poslední text zkopírován na výstup.

8.8 Šablony elementů fo:page-number a fo:page-number-citation

Tyto šablony jsou volány ze šablony elementu *fo:block* a ze všech šablon inline elementů, které mohou mít nějaké potomky.

Obě šablony mají velmi jednoduchou strukturu. Nejprve volají šablonu „predchozitext“ a poté šablonu „vlastnostiip“. Nakonec vytvoří buď instrukci PAGE nebo instrukci PAGEREF následovanou identifikátorem.

U instrukce PAGEREF se při převodu vyskytly hned dva problémy. První problém se týká též elementu *fo:basic-link* a bylo jím to, že při převodu atributu id, který obsahoval mezery, si Word na jejich místo doplnil pomlčky. Už si je však nedoplnil u odkazů na tyto identifikátory, takže odkazy nefungovaly. Tento problém jsem vyřešil jednoduše tak, že jsem pomlčku do identifikátorů i odkazů na ně doplnil sám.

Druhý problém spočívá v tom, že se při otevření dokumentu čísla stran vytvořená instrukcí PAGEREF neobjeví. Je to asi tím, že při postupném vytváření dokumentu Word ještě neví, na jaké straně se odkazovaný text objeví. Nikde jsem ani nenašel možnost, jak by se dala zajistit automatická aktualizace těchto odkazů na čísla stránky při otevření dokumentu. Existuje pouze možnost otevřením menu „nástroje“, v něm menu „možnosti“ a v něm kliknutím na záložku „tisk“ a v ní zaškrtnutí

položky „aktualizovat pole“ přinutit Word, aby při tisku dokumentu vytiskl správná čísla. Další možností je otevřít menu „úpravy“ a v něm kliknout na položku „vybrat vše“ a poté stisknout klávesu F9, čímž se aktualizují všechna pole v dokumentu.

8.9 Šablony pro element fo:leader

Pro každý odstavec, ve kterém se nachází element fo:leader, se musí vytvořit tabulátor. Pro to slouží šablona elementu *fo:leader* v módu „blok“, která je volaná ze šablony elementu *fo:block* v místě odstavcových vlastností. Tato šablona tedy vytvoří tabulátor, okolo kterého se má text za tabulátorem vycentrovat, dále je též převedena podoba tabulátoru (mezery, tečky, atd.) a nakonec též nepřesně jeho umístění. Jestliže je určen atribut *leader-length* a jsme na začátku odstavce, můžeme jeho umístění stanovit přesně. V ostatních případech je stanoven nepřesně jako pevné číslo, které znamená přibližně tři čtvrtiny stránky A4.

Dále ještě existuje normální šablona pro tento element, která se volá ze stejných míst, jako ostatní inline elementy. Tato šablona volá šablony „predchozitext“ a „vlastnostiip“ a dále vloží na místo textu tabulátor pomocí elementu w:tab.

8.10 Šablona pro element fo:basic-link

Je volaná a má stejnou strukturu jako šablona elementu *fo:inline*, až na to, že její obsah za voláním šablony „predchozitext“ je obalen elementem *w:hlink*, který zajišťuje funkci obsahu tohoto elementu jako odkazu. Při převodu tohoto elementu se vyskytl malý problém, který byl popsán v části 8.8.

8.11 Šablony pro elementy fo:marker a fo:retrieve-marker

Jak už jsem napsal, pro každý element *fo:marker* s různým atributem *marker-class-name*, je vytvořen styl se jménem odpovídajícím obsahu tohoto atributu. Tento styl nastavuje velikost písma na 0 a barvu písma na barvu pozadí. Šablona pro element *fo:marker* je pak volaná jako ostatní šablony pro inline elementy, pouze se musí z odstavce, který obsahuje element fo:marker a neobsahuje element *fo:block* vyloučit volání odstavcových vlastností (jinak by se v dokumentu mohla například objevit nechtěná mezera mezi odstavci). Samotná šablona pouze vypíše text obsažený v elementu *fo:marker* definovaným stylem (to znamená, že text je neviditelný).

Šablona elementu *fo:reterieve-marker* je též volaná jako ostatní inline elementy a má stejnou strukturu jako šablona elementu *fo:page-number-citation*. Místo instrukce PAGEREF však obsahuje instrukci STYLEREF následovanou názvem stylu, na který se odkazujeme. Jestliže je hodnota atributu *retrieve-position* nastavena na last-starting-within-page nebo last-ending-within-page je k instrukci též přidán přepínač \l, který zapíná prohledávání stránky zdola nahoru.

9 Závěr

Cílem této práce bylo prozkoumat možnosti převodu formátovacích objektů do formátu WordML. Oba formáty jsou založeny na XML a tak jsem ve druhé kapitole stručně pospal, co to XML vlastně je. Podstatnou část této práce tvoří kapitoly 3, 4 a 5 v nichž jsem popsal oba formáty a vzájemně je porovnal. V kapitole 6 jsem dále stručně nastínil možnosti implementace převodu formátovacích objektů do formátu WordML. V kapitole 7 uvádím možnost, jak se dá převod integrovat přímo do Wordu a kapitola 8 popisuje programovou realizaci převodu v XSLT.

Výsledkem této práce je zjištění, že podstatnou část formátovacích objektů lze převést do formátu WordML. Součástí této práce je též ukázka převodu části formátovacích objektů do formátu WordML s využitím XSLT. V příloze 1 a 2 najdete popis toho, jaké elementy a atributy tato ukázka převádí a též do jaké míry je převádí. Na přiloženém CD najdete tuto ukázku v souboru prevod.xsl a též několik souborů ve formátu .fo, na kterých byla testována funkčnost převodu. Na CD najdete též XML schéma formátovacích objektů v souboru fo.xsd, které vám umožní implementovat převod přímo do Wordu, jak bylo popsáno v 7. kapitole. Pro ty, kteří si nemohou vyzkoušet zobrazení ukázkových souborů ve Wordu, je určena příloha 3, kde najdete ukázku zobrazení několika souborů .fo ve Wordu (tak jak ji převedl můj styl) a též v Acrobatu Readeru (primární zobrazení .fo souboru).

Příloha 1

Seznam všech formátovacích objektů, jejich funkcí a stav převodu		
Formátovací objekt	**Funkce**	**Stav převodu**
fo:root	Kořenový element dokumentu.	**ano**
fo:page-sequence	Obsahuje další elementy, které definují obsah stránek v dokumentu	**ano**
fo:page-sequence-master	Definuje rozvržení sekvence stránek.	ne
fo:single-page-master-reference	Definuje část sekvence stránek skládající se z jedné stránky.	ne
fo:repeatable-page-master-reference	Definuje část sekvence stránek skládající se z více opakujících se stránek stejného rozvržení.	ne
fo:repeatable-page-master-alternatives	Definuje část sekvence stránek skládající se z více stránek různého rozvržení.	ne
fo:conditional-page-master-reference	Definuje, jaké rozvržení stránky se má použít za různých podmínek.	ne
fo:layout-master-set	V jeho potomcích jsou definována rozvržení stránek.	**ano**
fo:simple-page-master	Definuje rozvržení stránek.	**ano**
fo:region-body	Definuje část stránky pro normální tok textu.	**ano**
fo:region-before	Definuje oblast záhlaví stránky.	**ano**
fo:region-after	Definuje oblast zápatí stránky.	**ano**
fo:region-start	Definuje oblast stránky vlevo od normálního textu.	ne
fo:region-end	Definuje oblast stránky vpravo od normálního textu.	ne
fo:declarations	Obsahuje globální deklarace stylu.	ne
fo:color-profile	Definuje barevný profil ICC.	ne
fo:flow	Jeho obsah je rozdělen do stránek.	**ano**
fo:static-content	Jeho obsah je zobrazen v ostatních regionech stránky na jedné nebo více stranách dané sekvence stránek.	**ano**
fo:title	Obsahuje titulek dané sekvence stránek.	ne

Seznam všech formátovacích objektů, jejich funkcí a stav převodu		
Formátovací objekt	**Funkce**	**Stav převodu**
fo:block	Používán k formátování odstavců, nadpisů atd.	**ano**
fo:block-container	Používán k seskupení textových bloků s různými směry toků textu nebo různě natočeným textem.	ne
fo:table-and-caption	Používán k vytvoření tabulky s popiskem.	**ano**
fo:table-caption	Obsahuje popisek tabulky.	**ano**
fo:table	Používán k vytvoření tabulky.	**ano**
fo:table-column	Definuje sloupce tabulky.	**ano**
fo:table-header	Obsahuje hlavičku tabulky.	**ano**
fo:table-footer	Obsahuje patičku tabulky.	**ano**
fo:table-body	Obsahuje řádky tabulky.	**ano**
fo:table-row	Seskupuje buňky tabulky do řádku.	**ano**
fo:table-cell	Definuje obsah buňky tabulky.	**ano**
fo:list-block	Používán k formátování seznamů.	**ano**
fo:list-item	Položka seznamu sestávající z odrážky a textu.	**ano**
fo:list-item-label	Definuje odrážku seznamu.	částečně
fo:list-item-body	Definuje textovou část seznamu.	**ano**
fo:bidi-override	Přinutí napsat řetězec textu v určeném směru.	ne
fo:character	Používán k formátování jednoho písmene.	ne
fo:initial-property-set	Formátování prvního řádku odstavce.	ne
fo:external-graphic	Vkládání obrázků, které jsou uloženy vně dokumentu.	ne
fo:instream-foreign-object	Vkládání obrázků, které jsou součástí dokumentu.	ne
fo:inline	Formátování textu v části odstavce.	**ano**
fo:inline-container	Podobný účel jako fo:block-container, ale jeho obsahem mohou být pouze inline elementy.	ne
fo:leader	Vytváření vodících linek.	částečně
fo:page-number	Vloží aktuální číslo stránky.	**ano**

Seznam všech formátovacích objektů, jejich funkcí a stav převodu		
Formátovací objekt	**Funkce**	**Stav převodu**
fo:page-number-citation	Vrátí číslo stránky, na které se nachází určitý text.	**ano**
fo:basic-link	Vytvoří odkaz na jiný dokument nebo jiné místo v aktuálním dokumentu.	**ano**
fo:multi-switch	Obsahuje možnosti různého zobrazení textu na místě v dokumentu. Využíván k tvorbě dynamických efektů.	ne
fo:multi-case	Obsahuje alternativu zobrazení textu.	ne
fo:multi-toggle	Použita k přepínání mezi různými zobrazeními textu při jeho aktivaci.	ne
fo:multi-properties	Používán k přepínání mezi dvěmi nebo více množinami vlastností.	ne
fo:multi-property-set	Používán k definici alternativních množin formátovacích vlastností.	ne
fo:float	Umístění plovoucích objektů na stránku.	ne
fo:footnote	Vložení poznámky pod čarou.	ne
fo:footnote-body	Obsah poznámky pod čarou.	ne
fo:wrapper	Používán k definici děděných vlastností pro skupinu formátovacích objektů.	ne
fo:marker	Používán k tvorbě živých záhlaví.	**ano**
fo:retrieve-marker	Vložení obsahu nejbližšího elementu fo:marker do záhlaví či zápatí aktuální stránky.	**ano**

Příloha 2

Následující tabulka obsahuje seznam formátovacích vlastností, které jsem alespoň z části převedl. Údaje psané tučně zeleně jsou převedené, údaje psané červeně nikoliv a údaje psané oranžově jsou převedené nepřesně.

Vlastnost	Hodnoty	Počáteční hodnota	Dědičná?	Poznámka
background-color	**<klíčová slova>** | <funkce> | inherit | **transparent** | **<hexadecimální tvar>**	**transparent**	**ne**	Barva pozadí.
border-„strana“-color, border-color	**<klíčová slova>** | <funkce> | inherit | **<hexadecimální tvar>**	**hodnota vlastnosti color**	**ne**	Barva ohraničení.
border-„strana“-style, border-style	**none** | **hidden** | **dotted** | **dashed** | **solid** | double | **groove** | **ridge** | **inset** | **outset**	**none**	**ne**	Styl ohraničení. U hodnoty double špatná celková šířka hranice.
border-„strana“-width, border-width	**thin** | **medium** | **thick** | **<absolutní míra>** | inherit	**medium**	**ne**	Šířka ohraničení.
border-separation	**<absolutní míra** ve dvou dimenzích> | inherit	**0**	ano	Šířka mezery mezi ohraničením buněk. Převedena pouze jako jedna hodnota.
break-after	**auto** | column | **page** | even-page | odd-page | inherit	**auto**	**ne**	Vloží zalomení za obsah aktuálního elementu.
break-before	**auto** | column | **page** | even-page | odd-page | inherit	**auto**	**ne**	Vloží zalomení před obsah aktuálního elementu.
caption-side	**before** | **after** | start | end | **top** | **bottom** | left | right | inherit	**before**	ano	Strana titulku. Titulek vlevo přesunut nad a titulek vpravo pod tabulku.

Vlastnost	Hodnoty	Počáteční hodnota	Dědičná?	Poznámka
color	<klíčová slova> | <funkce> | inherit | <hexadecimální tvar>	není	ano	Barva písma.
column-width	<procenta> | <absolutní mí-ra> | <funkce proportional-column-width()>	není	ne	Šířka sloupce v tabulce.
display-align	auto | before | center | after | inherit	auto	ano	Vertikální zaro-vnání. Pouze u textu v buňce ta-bulky.
end-indent	<procenta> | <absolutní mí-ra> | inherit	0	ano	Mezera mezi pra-vým okrajem stránky a textem.
extent	<procenta> | <absolutní mí-ra> | inherit	0	ne	Velikost záhlaví a zápatí.
external-destination	<URI>	není	ne	Odkaz na doku-ment.
font-family	<řetězec názvů fontů a typů fontů seřazený dle priority>	není	ano	Použitý font. Správně funguje pouze při jednom zadaném fontu.
font-size	<klíčová slova absolutní> | <klíčová slova relativní> | <procenta> | <absolutní mí-ra> | inherit	medium	ano	Velikost písma.
font-stretch	ultra-condensed | extra-condensed | condensed | semi-condensed | normal | semi-expanded | expanded | extra-expanded | ultra-expanded | wider | narrower | inherit	normal	ano	Šířka písma.

Vlastnost	Hodnoty	Počáteční hodnota	Dědičná?	Poznámka
font-style	**normal** | **italic** | **oblique** | backslant | inherit	**normal**	**ano**	Styl písma. Hodnota backslant převedena na italiku.
font-variant	**normal** | **small-caps** | inherit	**normal**	**ano**	Varianta písma.
font-weight	**normal** | **bold** | bolder | lighter | 100 | 200 | 300 | **400** | 500 | 600 | 700 | 800 | 900 | inherit	**normal**	**ano**	Tloušťka písma. Hodnoty bold, bolder a 500 a výše jsou psány tučně, ostatní normálně.
height	<procenta> | **<absolutní míra>** | auto | inherit	auto	**ne**	Výška oblasti. Převedeno pouze jako výška řádků tabulky.
id	**<id>**	není	**ne**	Identifikátor. Negenerován, pouze převedena jeho hodnota.
initial-page-number	**auto** | **<číslo>** | auto-odd | auto-even | inherit	**auto**	**ne**	Počáteční číslo stránky.
internal-destination	**<idref>**	není	**ne**	Odkaz na id. Negenerován, pouze převedena jeho hodnota.
keep-together	**auto** | **always** | <číslo>	**auto**	**ano**	Nevkládat zalomení mezi obsah elementu. Převedeno pouze jako nesložený datový typ.

Vlastnost	Hodnoty	Počáteční hodnota	Dědičná?	Poznámka
keep-with-next	**auto** | **always** | <číslo>	**auto**	**ne**	Nevkládat zalomení za obsah elementu. Převedeno pouze jako nesložený datový typ.
keep-with-previous	**auto** | **always** | <číslo>	**auto**	**ne**	Nevkládat zalomení před obsah elementu. Převedeno pouze jako nesložený datový typ.
leader-length	<procenta> | **<absolutní míra v rozmezí>** | inherit	minimum 0, optimum 12pt, maximum 100%	ano	Délka vodící linky.
leader-pattern	**space** | **rule** | **dots** | use-content | inherit	**space**	ano	Vzor vodící linky.
letter-spacing	**normal** | **<absolutní míra>** | <absolutní míra v rozmezí> | inherit	**normal**	**ano**	Úprava mezer mezi písmeny.
line-height	**normal** | **<absolutní míra>** | <absolutní míra v rozmezí> | <číslo> | <procenta> | inherit	**normal**	**ano**	Výška řádku.
margin-„strana“ margin	<procenta> | **<absolutní míra>** | auto	**0**	**ne**	Velikost okrajů. Převedeno pouze pro tabulky (zde jen levý a pravý) a rozvržení stránek.
marker-class-name	**<string>**	není	**ne**	Název záhlaví.

Vlastnost	Hodnoty	Počáteční hodnota	Dědičná?	Poznámka
master-name	<string>	není	ne	Název definice rozvržení stránek.
master-reference	<string>	není	ne	Odkaz na master-name.
number-columns-repeated	<číslo>	1	ne	Počet sloupců, pro které platí definice.
number-columns-spanned	<číslo>	1	ne	Počet buněk vertikálně spojených.
number-rows-spanned	<číslo>	1	ne	Počet buněk horizontálně spojených.
orphans	<číslo> | inherit	2	ano	Počet osamocených řádků na konci stránky. Výchozí hodnota 1, čísla různá od 0 převedena na 1.
padding-„strana“ padding	<absolutní míra> | <absolut-ní míra v rozmezí> | inherit	0	ne	Vnitřní okraje. Jen u ohraničení.
page-height	auto | <absolutní míra> | indefinite | inherit	auto	ne	Výška stránky.
page-width	auto | <absolutní míra> | indefinite | inherit	auto	ne	Šířka stránky.
provisional-distance-between-starts	<procenta> | <absolutní mí-ra> | inherit	24pt	ano	Vzdálenost mezi začátkem odráž-ky a začátkem textu v seznamu.
ref-id	<idref> | inherit	není	ne	Odkaz na id.
retrieve-class-name	<string>	není	ne	Odkaz na název záhlaví.

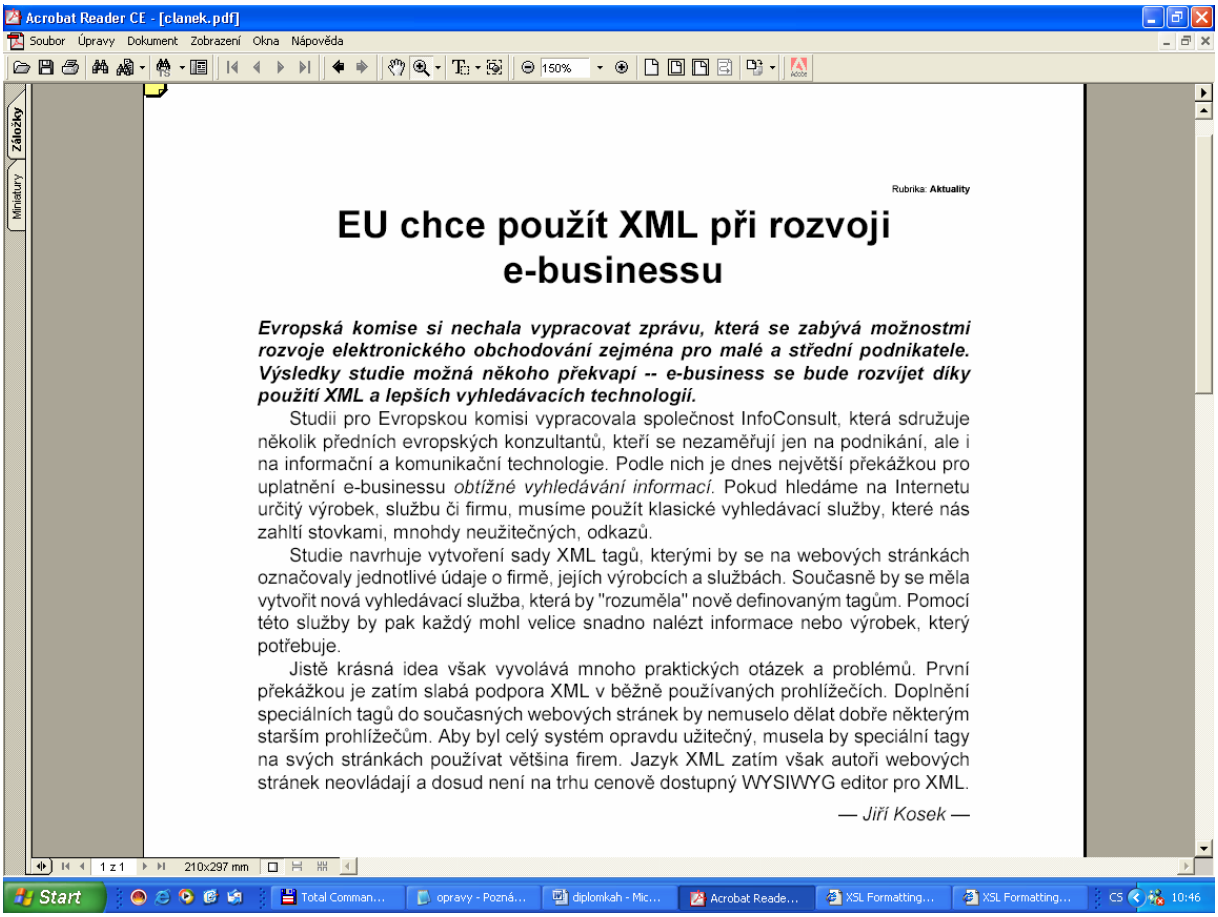
Vlastnost	Hodnoty	Počáteční hodnota	Dědičná?	Poznámka
retrieve-position	first-starting-within-page | first-including-carryover | last-starting-within-page | last-ending-within-page	first-starting-within-page	ne	Pozice elementu fo:marker na stránnce, odkud se vezme.
rule-style	none | dotted | dashed | solid | double | groove | ridge | inherit	solid	ano	Styl vodící linky.
score-spaces	true | false | inherit	true	ano	Textová dekorace i u mezer? Převedeno jen pro podtržení.
space-before	<absolutní míra v rozmezí> | inherit	0	ne	Mezera nahoře.
space-after	<absolutní míra v rozmezí> | inherit	0	ne	Mezera dole.
start-indent	<procenta> | <absolutní míra> | inherit	0	ano	Mezera mezi levým okrajem stránky a textem.
table-layout	auto | fixed | inherit	auto	ne	Rozvržení tabulky. Pokud jsou definovány sloupce, nastaveno na fixed.
table-omit-header-at-break	true | false	false	ne	Zobrazit hlavičku tabulky v zalomení?
text-align	left | right | center | justify | start | end | inside | outside | <string> | inherit	start	ano	Zarovnání textu. (start=left, end=right)
text-decoration	none | underline | overline | line-through | blink	none	zvláštní	Dekorace textu.
text-indent	<procenta> | <absolutní míra> | inherit	0	ano	Odsazení prvního řádku odstavce.

Vlastnost	Hodnoty	Počáteční hodnota	Dědičná?	Poznámka
text-shadow	**none** | <barva> | <absolutní míry> | inherit	**none**	**ne**	Stínování textu. Stínování pouze zapnuto či vypnuto.
text-transform	**none** | **uppercase** | capitalize | lowercase | inherit	**none**	**ano**	Transformace textu.
vertical-align	**baseline** | **sub** | **super** | middle | text-top | text-bottom | top | bottom | **<absolutní míra>** | **<procen-ta>** | inherit	**baseline**	**ne**	Vertikální posun textu.
widows	<číslo> | inherit	2	**ano**	Počet osamocených řádků na začátku stránky. Výchozí hodnota 1, čísla různá od 0 převedena na 1.
width	**<procenta>** | **<absolutní mí-ra>** | **auto** | inherit	**auto**	**ne**	Šířka oblasti. Převedeno pouze pro tabulku a buňku.

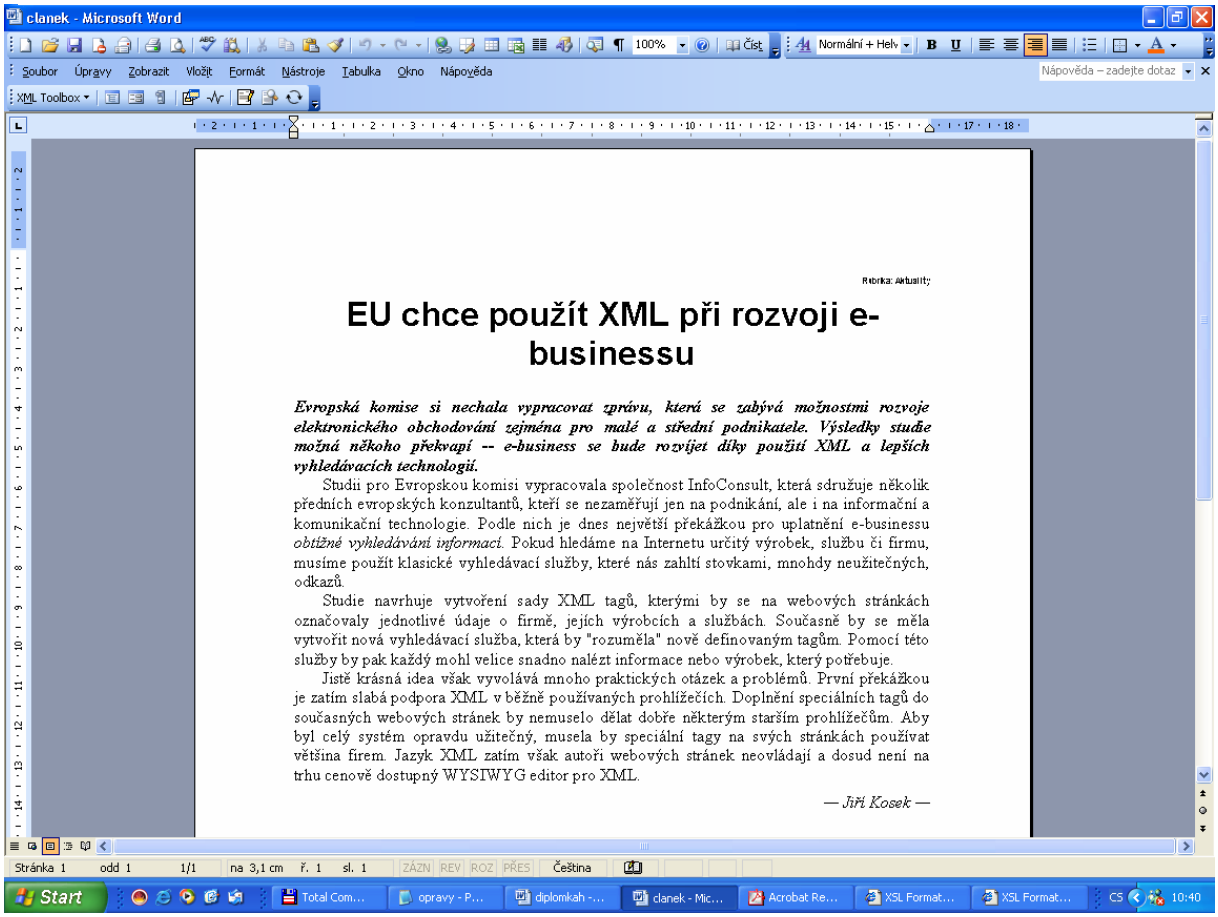
Příloha 3

V této příloze najdete ukázku zobrazení částí souborů .fo tak, jak se zobrazily v Acrobatu Readeru (primární zobrazení .fo s použitím XEPu) a tak, jak se zobrazily ve Wordu po použití mého stylu prevod.xsl.

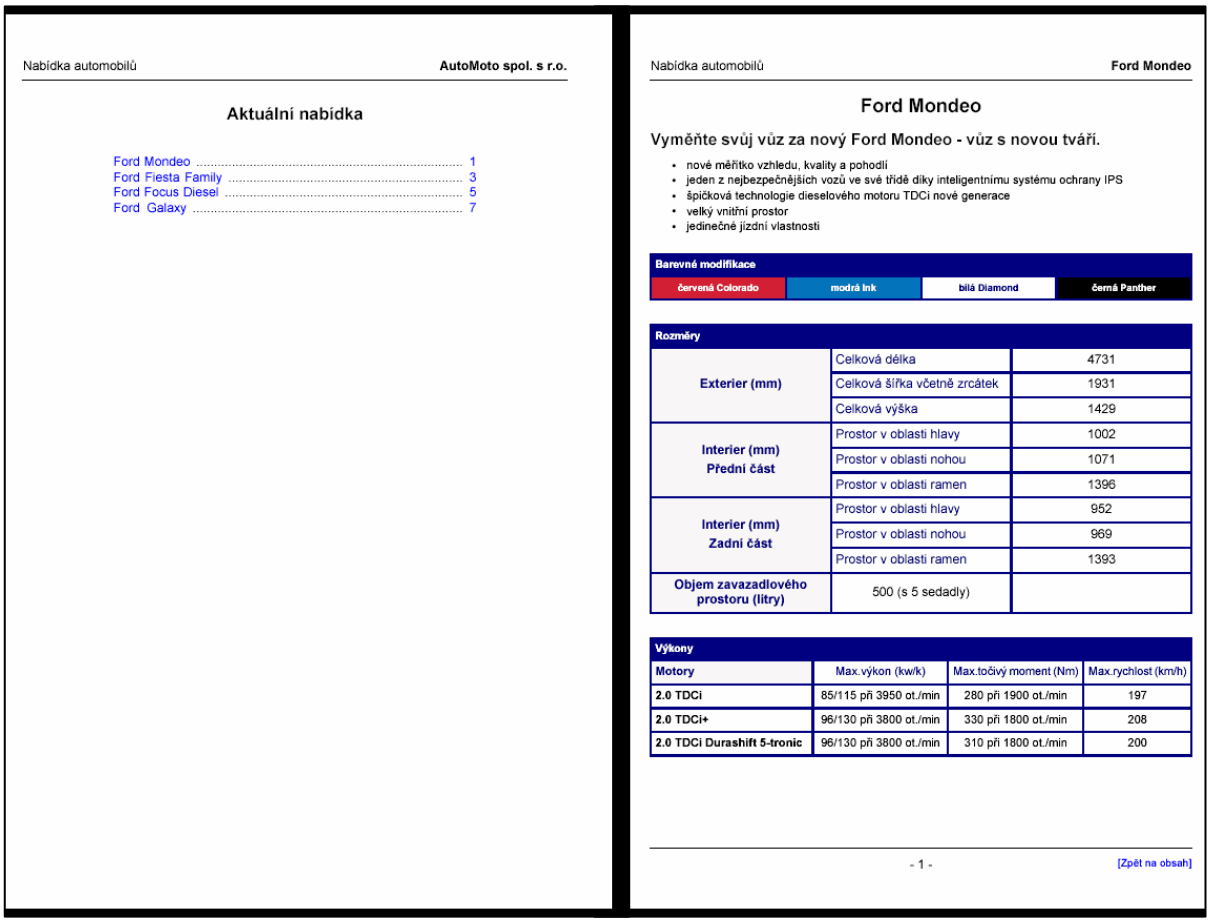
Soubor clanek.fo – zobrazení v Acrobatu Readeru



Soubor clanek.fo – zobrazení ve Wordu



Soubor katalog.fo (první dvě stránky) – zobrazení v Acrobatu Readeru



Aktuální nabídka

Ford Mondeo

Vyměňte svůj vůz za nový Ford Mondeo - vůz s novou tváří.

- nové měřítko vzhledu, kvality a pohodlí
- jeden z nejbezpečnějších vozů ve své třídě díky inteligentnímu systému ochrany IPS
- špičková technologie dieselového motoru TDCi nové generace
- velký vnitřní prostor
- jedinečné jízdní vlastnosti

Barevné modifikace

červená Colorado	modrá Ink	bílá Diamond	černá Panther

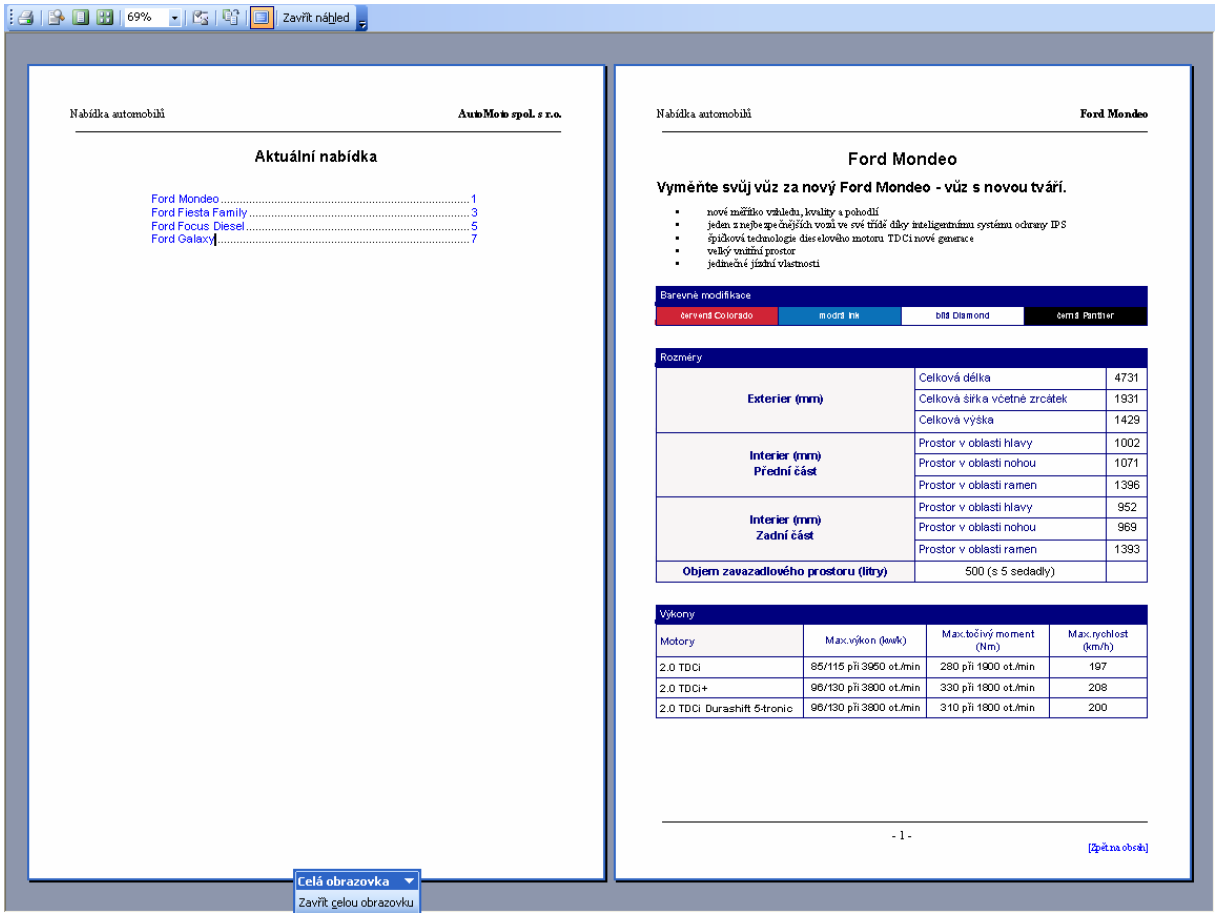
Rozměry

Exterier (mm)	Celková délka	4731
	Celková šířka včetně zrcátek	1931
	Celková výška	1429
Interier (mm) Přední část	Prostor v oblasti hlavy	1002
	Prostor v oblasti nohou	1071
	Prostor v oblasti ramen	1396
Interier (mm) Zadní část	Prostor v oblasti hlavy	952
	Prostor v oblasti nohou	969
	Prostor v oblasti ramen	1393
Objem zavazadlového prostoru (litry)	500 (s 5 sedadly)	

Výkony

Motory	Max.výkon (kw/k)	Max.točivý moment (Nm)	Max.rychlost (km/h)
2.0 TDCi	85/115 při 3950 ot./min	280 při 1900 ot./min	197
2.0 TDCi+	96/130 při 3800 ot./min	330 při 1800 ot./min	208
2.0 TDCi Durashift 5-tronic	96/130 při 3800 ot./min	310 při 1800 ot./min	200

Soubor katalog.fo (první dvě stránky) – zobrazení ve Wordu



Literatura

[1] Adler, S. – Berglund, A. – Caruso, J. – Deach, S. – Grosso, P. – Gutentag, E. – Milowski, A. – Pernell, S. – Richman, J. – Zilles, S.: *Extensible Stylesheet Languague (XSL) – Version 1.0.* W3C. 2001. Dostupné na WWW: http://www.w3.org/TR/xsl

[2] Bos, B. – Lie, H. – Lilley, Ch. – Jacobs, I.: *Cascading Style Sheets, level 2 CSS2 Specification.* W3C. 1998. Dostupné na WWW: http://www.w3.org/TR/REC-CSS2/

[3] Fallside, D.C.: *XML schema Part 0 – Primer.* W3C. 2001. Dostupné na WWW: http://www.w3.org/TR/xmlschema-0/

[4] Goldfarb, Ch. – Walmsley, P.: *XML in Office 2003.* Pearson Education, 2004. ISBN 0-13-142193-X.

[5] Harold, E.: Chapter 18 of the XML Bible, Second Edition : XSL Formatting Objects. 2002. Dostupné na WWW: http://www.ibiblio.org/xml/books/bible2/chapters/ch18.html

[6] Holzner, S.: *XSLT Příručka internetového vývojáře.* Computer Press, Praha 2002. ISBN 80-7226-600-4.

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

[8] *Microsoft Office Word XML Content Development Kit.* Microsoft Corporation. 2004. Dostupné na WWW: http://msdn.microsoft.com/office/understanding/xmloffice/documentation/default.aspx

[9] *XML software.* Dostupné na WWW: http://www.xmlsoftware.com

[10] Young, M.: *XML krok za krokem.* Mobil Media, Praha 2002. ISBN 80-86593-28-2.