Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Bakalářská práce

XML a OpenDocument Format standardy a jejich použití v praxi

Marek Pluhař

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Marek Pluhař

Informatika

Název práce

XML a OpenDocument Format standardy a jejich použití v praxi

Název anglicky

XML and OpenDocument Format standards and their use in practice

---

Cíle práce

Bakalářská práce je tématicky zaměřena na problematiku datových formátů XML a OpenDocument Format. Hlavním cílem práce je analyzovat současný stav využívání datových formátů a jejich rozdíly. Dílčí cíle bakalářské práce jsou:
- vytvořit přehled řešené problematiky
- zhodnotit praktické použití jednotlivých datových formátů
- napsat makro pracující s vybraným datovým formátem

Metodika

Metodika řešené problematiky balaůářské práce je založena na studiu a analýze odborných informačních zdrojů. Dále také na vlastních zkušenostech získaných při tvorbě aplikací. Praktická část práce je zaměřena na porovnání datových formátů a jejich použití v praxi. Dále obsahuje makro pracující s vybraným datovým formátem.

Na základě syntézy teoretických poznatků, praktických zkušeností a výsledků vlastního řešení budou formulovány závěry bakalářské práce.

Doporučený rozsah práce

30-40 stran

Klíčová slova

datový formát XML, OpenDocument Format

Doporučené zdroje informací

EISENBERG, J., D. OASIS OpenDocument Essentials. Raleigh: Lulu.com. 2006. 303 pages. ISBN 1411668324.

FERILLI, S. Automatic Digital Document Processing and Management: Problems, Algorithms and Techniques (Advances in Computer Vision and Pattern Recognition). London: Springer. 2011. 308 pages. ISBN 0857291971.

ŠTĚDROŇ, Bohumír. Open Source software: ve veřejné správě a soukromém sektoru. Praha: Grada. 2009. 128 s. ISBN 978-80-247-3047-9.

Předběžný termín obhajoby

2015/16 LS – PEF

Vedoucí práce

Ing. Čestmír Halbich, CSc.

Garantující pracoviště

Katedra informačních technologií

Elektronicky schváleno dne 28. 10. 2015

Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 10. 11. 2015

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 13. 03. 2016

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "XML a OpenDocument Format standardy a jejich použití v praxi" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 14.3.2016 _____

Poděkování

Rád bych touto cestou poděkoval Ing. Čestmíru Halbichovi, CSc. za vstřícnost, trpělivost a spolupráci při tvorbě této práce. Dále bych chtěl poděkovat mojí rodině za podporu během studia.

XML a OpenDocument Format standardy a jejich použití v praxi

Souhrn

Bakalářská práce se zabývá XML a OpenDocument Format. Celá práce je rozdělena na teoretickou a praktickou část. Teoretická část je členěna na dvě samostatné sekce. První se věnuje popisu formátu XML. Druhá se týká formátu OpenDocument. V práci je přehledně vysvětlena problematika obou studovaných témat.

Praktická část je také rozdělena na dvě sekce. První z nich je věnována použití XML a druhá vytváření ODF makra. V oblasti věnované XML, bylo provedeno převedení XML souboru do HTML výstupu, pomocí XSLT. V oblasti věnované ODF makru, bylo vytvořeno makro, které exportuje soubor v programu Calc do formátu PDF. Praktická část ověřila znalosti získané v teoretické části a umožnila splnění obou praktických cílů.

Klíčová slova: XML, ODF, XPointer, XPath, XLink, XSLT, OpenDocument, DTD, Format, HTML

XML and OpenDocument Format standards and their use in practice

Summary

The bachelor thesis deals with XML and OpenDocument Format. The entire work is divided into theoretical and practical part. The theoretical part is divided into two separate sections. The first is devoted to the description of the XML format. The second is related to the OpenDocument Format. The work clearly explains the issues of studied topics.

The practical part is also divided into two sections. The first one is devoted to the use of XML and the second to creation of ODF macro. In the XML area was done converting XML file into HTML output using XSLT. In the ODF area was created ODF macro, which exports file in program Calc into PDF format. The practical part verified knowledge gained in the theoretical part and allowed to fulfill the both practical aims.

Keywords: XML, ODF, XPointer, XPath, XLink, XSLT, OpenDocument, DTD, Format, HTML

Obsah

Seznam tabulek

Seznam obrázků

1 Úvod

Než začnu psát o Open Document Format, rád bych nastínil problematiku současného stavu kancelářských balíků a aplikací. Všechny dokumenty jsou uloženy v chráněných formátech, které jsou většinou binární. Pokud se pracuje pouze v jednom kancelářském balíku, tak je vše v pořádku. Je možné přenášet data z jedné části do druhé. Toho se využívá například při tvorbě prezentací, kdy se použije text z Wordu a přenese se do PowerPointu. Často se také používá Excel, kdy se použije sada čísel a vloží se do Wordu do tabulky.

Bohužel problémy začínají v okamžiku, kdy je potřeba přenést data z jednoho kancelářského balíku do úplně jiného. Tímto způsobem není možné přenášet data, protože vnitřní struktura dat je neznámá. Nelze napsat program, který vytvoří nový dokument, který by obsahoval všechny hlavičky z jiného dokumentu. Pokud tedy potřebujeme něco, co nebylo poskytnuto dodavatelem softwaru, je nutné data převést do neutrálního nebo universálního formátu jako je například Rich Text Format (RTF). Bohužel i přes možnost konverze mohou být ztraceny formátovací informace, které jsme uložili do daných dat. Navíc stále existuje riziko, že se mohou stát naše data nepřístupná a to z důvodu, pokud poskytovatel přejde na nový interní formát a přestane podporovat současnou verzi.

Tato práce by měla objasnit problematiku datových formátů. Budu v ní popisovat nejdříve formát XML, který je základem. Poté se budu věnovat OpenDocument Format, který se čím dál častěji stává vhodnou alternativou. Nakonec popíši a ukážu na praktických příkladech využití obou technologií a to především konverzi daných formátů do jiných.

2 Cíl práce a metodika

2.1 Cíl práce

Bakalářská práce je tématicky zaměřena na problematiku datových formátů XML a OpenDocument Format. Hlavním cílem práce je analyzovat současný stav využívání datových formátů a jejich rozdíly. Mezi dílčí cíle bakalářské práce patří vytvoření přehledu řešené problematiky, zhodnocení praktického použití jednotlivých datových formátů a napsání makra pracujícího s vybraným datovým formátem.

2.2 Metodika

Metodika řešené problematiky bakalářské práce je založena na studiu a analýze odborných informačních zdrojů, především odborné literatury a internetových zdrojů. Dále také na vlastních zkušenostech získaných při tvorbě aplikací. Praktická část práce je zaměřena na využití datových formátů v praxi a také obsahuje makro pracující s OpenDocument formátem. Na základě syntézy teoretických poznatků, praktických zkušeností a výsledků vlastního řešení budou formulovány závěry bakalářské práce.

3 XML

3.1 Úvod do XML

Znalost XML je nezbytná, pokud je zapotřebí manipulovat přímo s OpenDocument soubory, jelikož XML je základem OpenDocument Format. XML je zkratka pro Extensible Markup Language, což je formát pro data a dokumenty, vytvořený World Wide Web Consorciem, zkráceně W3C. "Markup" označuje způsob, jak vyjádřit strukturu dokumentu v rámci samotného dokumentu. XML má kořeny ve značkovacím jazyce, který se nazývá SGML, což znamená Standard Generalized Markup Language. XML byl vytvořen z důvodu strojově čitelných dokumentů umístěných na webu, na rozdíl od HTML, které bylo vytvořeno, aby dokumenty byly lidsky čitelné. Díky tomu se dokumenty stávají přístupné všem uživatelům na rozdíl od HTML. V HTML jsou předdefinované tagy pomocí úhelníků <> a celý soubor je rozdělen na sekce, jako je hlavička, tělo a patička. XML se podobá HTML pouze v tom, že užívá úhelníky pro jednotlivé tagy. Na rozdíl od HTML nemá žádné předdefinované prvky, ale pouze soubor pravidel, podle kterého bychom se měli řídit při tvorbě. XML je podmnožinou SGML. Naproti tomu HTML je aplikací SGML. OpenDocument používá XML pro vyjádření svých operací.[1][5]

3.2 Anatomie XML dokumentu

Následující příklad ukazuje XML dokument, který je možno použít k popisu dvou zvířat, v mém případě dvou psů:

```
<? xml version = "1.0" encoding = "us-ascii"?>
        <psi>
                <pes id = "001">
                        <name>Honey</ name>
                        <rasa>Bandog</ rasa>
                </ pes>
                <pes id = "002">
                        <name>Doug</ name>
                        <rasa> Americký pitbullteriér</ rasa>
                </ pes>
        </psi>
```

První řádek dokumentu deklaruje XML soubor. Deklarace je potřebná pro určení, verze XML, která byla použita a jaká znaková sadu je použita pro daný dokument. V příkladu, který je uveden výše, je dokument zakódovaný v ASCII.

Pokud by byla deklarace XML vynechána, procesor by předpokládal, že je použito kódování s označením UTF-8, což je kódování znakové sady Unicode. Nicméně, je nejlepší použít deklaraci XML vždy, kdy je to možné, z důvodu předcházení nejasnostem při kódování znaků a uvedení zpracovatelům verzi XML, jenž je použita.[1]

3.3 Elementy a atributy

Na druhém řádku výše uvedeného příkladu se nachází kořenový element dokumentu. Dokument XML musí obsahovat přesně jeden kořenový element, který obsahuje veškerý další obsah dokumentu. Název kořenového elementu definuje typ dokumentu XML. Prvky, které obsahují text i další prvky současně jsou klasifikovány jako smíšený obsah. Mnoho OpenDocument elementů obsahuje smíšený obsah. Ukázkový dokument "Psi" používá prvky k pojmenování a určení rasy psa. Každý prvek pes má atribut s názvem id. Na rozdíl od prvků, atributy mohou obsahovat pouze textový obsah. Jejich hodnoty musí být uvedeny v uvozovkách. Buď lze použít jednoduché uvozovky (') nebo složité uvozovky ("). Uvnitř dokumentů XML, se atributy často používají pro metadata, což jsou „data o datech“, a ty popisují vlastnosti obsahu elementů. ID obsahuje jedinečný identifikátor pro psa, který je popsán.[1][5]

Pokud jde o XML, nezáleží na tom, v jaké pořadí jsou umístěny atributy. Například, tyto dva prvky obsahují totožné informace:

<name = "zvíře pes" nohy = "4" />
<zvířecí nohy = "4" name = "pes" />

3.4 Syntaxe

XML verze 1.0 má určitá pravidla ohledně názvů elementů a atributů. Zejména:

- Jména mohou mít malá a velká písmena: například <zvíře /> není stejné jako <Zvíře />.
- Jména začínající na "xml jsou vyhrazeny pro použití v XML 1.0.
- Jméno musí začínat písmenem nebo podtržítkem, nesmí však začínat číslicí. Může pokračovat s jakýmkoliv písmenem, číslicí, podtržením.[1]

3.4.1 Well-Formed

Dokument XML, který je v souladu s pravidly syntaxe XML je označený jako „WellFormed“. Na té nejzákladnější úrovni WellFormed znamená, že by prvky měly být řádně uzavřeny.[1]

3.4.2 Komentáře

Stejně jako v HTML, je možné v rámci XML psát komentáře. XML komentáře jsou určeny pouze pro lidi, parser je ignoruje. Začátek komentář je indikován „<! –„ A konec komentáře s „->“. Komentáře jsou široce využívány, například pro zakontování dané části kódu či pro klasický komentář. Naopak tomu soubory OpenDocument jsou téměř vždy určeny strojům a obsahují jen malé množství komentářů nebo vůbec žádný.[1]

3.4.3 Entity

Dalším rysem XML, který je důležitý pro pochopení tvorby XML dokumentů je mechanismus pro znaky, které znaky mají zvláštní význam v XML. Musí existovat způsob, jak je vhodně zastoupit. Například, v některých případech by symbol „<“ skutečně mohl znamenat "méně než" spíše než, aby signalizoval začátek názvu elementu. Tím pádem by se špatně vytvořil dokument, protože by aplikace předpokládala, že byl započat nový element. Jiným příkladem tohoto problému je současné zahrnutí dvojitých uvozovek a apostrofů poli pro hodnotu atributu. XML se vyhýbá těmto problémům použitím předem definovaných entit. Slovo entita v souvislosti s XML znamená jednotku obsahu. XML

předefinovává subjekty pro následující symboly: levý úhel (<), pravá lomená závorka (>), apostrof ('), uvozovky ("), a ampersand (&).

Znak	Entita
<	&lt;
>	&gt;
’	&apos;
”	&quot;
&	&amp;

Tabulka 1: Přehled předdefinovaných entit.

Reference znaků nám umožňuje označit znak díky své číselné pozici ve znakové sadě Unicode. Kód může být vyjádřen v desítkové soustavě nebo s použitím X jako prefixu v hexadecimální soustavě.[1][4]

Aktuální znak	Reference znaků
1	&#48;
A	&#65;
Ń	&#xD1;
®	&#xAE;

Tabulka 2: Příklad referencí znaků v UTF-8.

3.4.4 Kódování znaků

Jedním ze základních požadavků pro XML procesory je podpora kódování standardních znaků Unicode. Unicode se pokouší zahrnout všechny světové jazyky v rámci jedné znakové sady. V důsledku toho je velmi obsáhlý.[1]

3.4.4.1 Kódování Unicode Schémat

Unicode 3.0 má více než 57,700 typů kódů, z nichž každá odpovídá jednomu znaku. Pokud bychom chtěli vyjádřit řetězec Unicode pomocí polohy každého znaku ve znakové sadě, tak by taková operace vyžadovala 4 oktety na každý znak. Pokud by tedy byl například dokument napsán celý v americké angličtině, byl by tento převod čtyřikrát

větší, než je požadováno. Všechny znaky v ASCII zapadají do 7-bit. Tato situace klade velké nároky jak na úložný prostor, tak na paměť. Naštěstí existují dva systémy kódování pro Unicode, které zmírňují tento problém: UTF-8 a UTF-16. V těchto kódováních je možno zpracovávat 8 nebo 16 bitové segmenty najednou. UTF-8 je vysoce efektivní kódování pro jazyky pomocí latinky, například angličtiny. Všechny znakové sady ASCII jsou zastoupené nativně v UTF-8 dokumentu ASCII. OpenDocument soubory jsou vždy kódovány v UTF-8. Oktet je řetězec 8 binárních čísel, nebo bitů.

Pokud XML parser zpracovává dokument, který obsahuje znaky z cizích jazyků, nemohou být takové znaky produkovány pouze jedním oktetem UTF-8, ale je potřeba dvou oktetů. V non-UNICODE prohlížeči nebo v textovém editoru bude tento soubor vypadat jako nečitelný text. XML 1.0 umožňuje vytvořit dokument registrovat u IANA. Evropské dokumenty jsou běžně zakódovány v jedné ze znakových sad ISO latiny, jako je ISO-8859-1. Japonské dokumenty běžně používají Shift-JIS, a čínské dokumenty používají GB2312 a Big 5. XML procesory nevyžadují specifikaci XML 1.0 podporovat nic víc než UTF-8 a UTF-16, ale nejčastěji podporují další kódování, jako je například USASCII a ISO-8859-1.[1][4]

3.4.5 Validita

Dokument označený jako „well formed“ neznamená totéž jako dokument označený jako "valid". Validní dokument musí být dobře strukturovaný. Kromě toho musí odpovídat definici typu dokumentu (DTD). Existují dvě různé definice typu dokumentu, který lze použít s XML:

1) DTD - Document Type Definition
2) XML schéma – XML jako alternativa k DTD

3.4.6 Definice typu dokumentu (DTD)

Účelem DTD je definovat povolené elementy a atributy v dokumentu a určit pořadí, ve kterém musejí být uvedeny v rámci typu dokumentu. DTD je obecně složeno z jednoho souboru, který obsahuje a vymezuje typy elementů a seznamů atributů.[1]

3.4.7 XML Jmenné prostory

XML 1.0 umožňuje vývojářům vytvářet své vlastní elementy a atributy, ale ponechává otevřený prostor pro názvy, které se jmenují stejně. Například zákazník knihkupectví použije tag <title> pro hodnoty, jako je paní nebo Dr., zatímco ve svém katalogu může používat <title> pro hodnotu, jako například nakladatelství. Vývojáři mohou použít k identifikaci názvů konkrétní slovníky pomocí Uniform Resource Identifikátorů (URI). Aby bylo možné rozlišit <title> prvky, je pro slovník přiřazeno jedinečné URI. Každý z těchto URI je spojen s prefixem, který je připojen k názvu elementu a dvojtečkou, tak aby určil, ke kterému slovníku patří. Pomocí těchto oborů názvů a jejich předpon, je možné, aby program prošel soubor a extrahoval pouze tituly knih a vyhýbal se například oslovení lidí.[1]

3.4.8 Nástroje pro zpracování XML

Existuje mnoho analyzátorů pro XML. Většina z těchto nástrojů je volně k dispozici. XML parser má obvykle podobu knihovny kódu. Program předá XML k analyzátoru a výsledkem jsou informace o obsahu dokumentu XML. Document Object Model (DOM) jsou analyzátory založené na práci odlišným způsobem. Konzumují celý vstupní XML dokument a předávají zpět stromovou strukturu dat.[1]

3.5 XSLT

XSL je zkratka pro Extensible Stylesheet Language, v překladu tedy styly jazyka pro XML dokumenty. XSLT je zkratka pro XSL transformace. XSLT umožňuje transformovat XML dokumenty do jiných formátů, například do XHTML. World Wide Web Consortium (W3C) začalo rozvíjet XSL, protože byla potřeba stylovat XML soubory.

CSS se používá k přidávání stylů k prvkům jazyka HTML. HTML používá předdefinované tagy, kde je význam tagů dobře znám. XML naopak nepoužívá předdefinované tagy, a proto není významu každého tagu často správně pochopen. Například element <table> by mohl naznačovat v HTML tabulku, kus nábytku, nebo něco jiného a prohlížeče nevědí, jak ji zobrazit. XSL popisuje, jak mají být zobrazeny jednotlivé elementy XML. XSL se skládá ze čtyř částí:

- XSLT - jazyk pro transformaci XML dokumentů
- XPath - jazyk pro navigaci v dokumentech XML
- XSL-FO - jazyk pro formátování XML dokumentů (přerušeny v roce 2013)
- XQuery - jazyk pro dotazování XML dokumentů

W3C přinesl nový standard pro formátování dokumentů. Od roku 2013 je CSS3 navržena jako náhrada za XSL-FO.

XSLT je nejdůležitější část XSL. XSLT zjednodušeně funguje tak, že změní každý element XML do (X) HTML elementu. Díky XSLT je možné přidat nebo odebrat elementy a atributy, je možné také uspořádat a řadit elementy, provádět testy a rozhodovat o tom, které elementy skrýt a které zobrazit. Běžný způsob, jak popsat proces transformace znamená, že XSLT transformuje zdrojový strom do zdrojového stromu XML.

XSLT používá XPath k nalezení specifické informace v dokumentu XML. XPath se také používá k navigaci mezi elementy a atributy v XML dokumentech. V transformačním procesu, XSLT používá XPath k definování části zdrojového dokumentu, který by měl odpovídat jednomu či více předdefinovaných šablon. Je-li nalezena shoda, XSLT bude transformovat odpovídající část zdrojového dokumentu do výsledného dokumentu. Všechny hlavní prohlížeče podporují XSLT a XPath. XSLT se stal doporučením W3C 16. listopadu 1999.[4]

3.5.1 Transformace XML do XHTML pomocí XSLT

Kořenový element, který deklaruje dokument, který má být stylován pomocí XSL má následují formu: <xsl: stylesheet> nebo <xsl: transform>. <xsl: stylesheet> a <xsl: transform> jsou zcela synonymní a je možné použít oboje. Správný způsob, jak deklarovat styly XSL podle W3C XSLT doporučení, vypadá následovně:

<xsl: stylesheet version = "1.0" xmlns: xsl = "http://www.w3.org/1999/XSL/Transform">

nebo

<xsl:transform version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

Výše zmíněný zápis jednoduše znamená, že pokud je potřeba získat přístup k XSLT elementům, atributům a funkcím, je nutné deklarovat jmenný prostor XSLT.[4]

- <xsl: template> je element, který se používá k vytvoření šablony. Atribut „match“ se používá k přidružení šablony k XML elementu. Atribut „match“ může být také použit pro definici šablony pro celý XML dokument. Výraz: match = "/" definuje celý dokument;
- <xsl: value-of> je element, který se používá pro extrakci hodnot z vybraného uzlu;
- <xsl: for-each> je element, který umožňuje provést tzv. looping v XSLT, může být použit pro výběr elementů v XML na základě zadané množiny uzlů, je možné také filtrovat výstup z XML souboru přidáním kritéria „select“ jako atributu v elementu <xsl: for-each>, například:[5]

  <xsl:for-each select="rejstrik/subjekt[id='003“]">

  Operátory filtrů jsou:
  - = (Rovná se)
  - ! = (Nerovná)
  - & Lt; méně než
  - & Gt; větší než
- <xsl: sort> je element, který se používá pro třídění výstupu, atribut „select“ určuje, dle čeho se mají hodnoty řadit;
- <xsl: choose> je element, který se používá ve spojení s <xsl: when> a <xsl: otherwise> pro vyjádření více podmíněných testů;
- <xsl: apply-templates> je element, který aplikuje šablonu do aktuálního elementu nebo potomka aktuálního elementu, pokud je vybrán atribut <xsl: apply-templates>, tak bude element zpracovávat pouze podřízený element, který odpovídá hodnotě atributu, je možné použít atribut select pro určení pořadí, ve kterém jsou zpracovány podřízené uzly;
- <xsl: if> je element, který je používán jako podmínka, například:

  <xsl:if test="cena > 500">

3.6 XPath

XPath se používá pro navigaci mezi prvky a atributy v XML dokumentu. XPath je důležitým prvkem ve standardu XSLT W3C. XPath poskytuje syntaxi pro vymezení části dokumentu a obsahuje knihovnu standardních funkcí. XPath používá určité výrazy pro stanovení cesty k výběru uzlů nebo množiny uzlů v XML dokumentu. XPath obsahuje více než 100 vestavěných funkcí. K dispozici jsou funkce pro srovnání a manipulaci s řetězci, číselnými hodnotami, datem a časem, logickými hodnotami atd. XPath je důležitým prvkem ve standardu XSLT. Bez jeho znalosti je schopnost vytvořit XSLT dokumenty ztížená. XPath se stal doporučení W3C 16. listopadu 1999.[5]

3.6.1 Uzly

V XPath existuje sedm druhů uzlů: element, atribut, text, jmenný prostor, zpracování-instrukce, komentář a uzly dokumentu. XML dokumenty jsou považovány za stromy uzlů. Nejvýše postavený prvek stromu se nazývá kořenový element. Níže uvádím příklad:

```
<knihovna>
  <kniha>
    <nazev lang="cs">Tajemný ostrov</nazev>
    <autor>J. A. Rawles</autor>
    <rok>2010</rok>
    <cena>100 Kč</cena>
  </kniha>
</ knihovna >
```

V tomto případě by kořenovým elementem byl element <knihovna> a element <kniha> by byl uzlový element. Atomické hodnoty jsou uzly s žádnými potomky nebo rodiči. Každý element a atribut má jeden rodičovský element. V příkladu uvedeném výše, byl element <kniha> potomkem elementu <knihovna>. Každý element může mít buď žádného, jednoho nebo více potomků. Sourozenci jsou uzly, které mají stejné rodiče.[5]

3.6.2 Syntaxe XPath

XPath pomocí výrazů vybírá jednotlivé uzly. Nejužitečnější výrazy pro stanovení cesty uvádím v tabulce, která je zobrazena níže:

Výraz	Popis
/	Výběr z kořenového uzlu.
//	Vybere uzly v dokumentu z aktuálního uzlu, které odpovídají výběru bez ohledu na to, kde se nacházejí.
.	Vybere aktuální uzel.
..	Vybere rodiče aktuálního uzlu.
@	Vybere atributy.

Tabulka 3: Nejužitečnější výrazy pro stanovení cesty.

Níže uvedená tabulka uvádí některé výrazy cesty a výsledek těchto výrazů:

Cesta	Výsledek
knihkupectví	Vybere všechny uzly s názvem "knihkupectví".
/knihkupectví	Vybere kořenový element knihkupectví.
knihkupectví / kniha	Vybere všechny elementy <kniha>, které jsou potomky elementu <knihkupectví>.
//kniha	Vybere všechny elementy knihy, bez ohledu na to, kde se nacházejí v dokumentu.
knihkupectví // kniha	Vybere všechny elementy <kniha>, které jsou potomci elementu <knihkupectví>, bez ohledu na to, kde se nacházejí.
//@lang	Vybere všechny atributy, které jsou pojmenovány názvem „lang“.

Tabulka 4: Příklady zadání cesty a její výsledek.

Predikáty jsou použity k nalezení konkrétního uzlu nebo uzlu, který obsahuje určitou hodnotu. Predikáty jsou vždy zakotveny v hranatých závorkách. V níže uvedené tabulce uvádím příklady některých z nich:

Cesta	Výsledek
/zoo/zvire[1]	Vybere první element <zvire>, který je potomkem elementu <zoo>.
/zoo/zvire[last ()]	Vybere poslední element <zvire>, který je potomkem elementu <zoo>.
/zoo/zvire[position () <3]	Vybere první dva elementy <zvire>, které jsou potomky elementu <zoo>.
//title[@ lang = 'cs']	Vybere všechny elementy <title>, které mají atribut "lang" s hodnotou "cs".

Tabulka 5: Přehled některých cest a její výsledků.

Pro výběr neznámých uzlů je možné zvolit zástupné znaky. Například výraz: „//*“ by vybral všechny elementy v dokumentu. Pomocí operátoru „|“, lze ve výrazu XPath použít několik cest. Například výrazem „//titul|//cena“ by vybral všechny tituly a ceny elementů v dokumentu.[5]

3.6.3 Osy XPath

Osa definuje relativní uzel nastavený na aktuálním uzlu. Níže předkládám tabulky s jednotlivými osami:

Název osy	Výsledek
ancestor	Vybere všechny předky aktuálního uzlu.
attribute	Vybere všechny atributy aktuálního uzlu.
child	Vybere všechny potomky aktuálního uzlu.
descendant	Vybere všechny potomky aktuálního uzlu.
following	Vybere vše v dokumentu po koncovou značku aktuálního uzlu.
namespace	Vybere všechny uzly jmenných prostorů aktuálního uzlu.
parent	Vybere rodiče aktuálního uzlu
preceding	Vybere všechny uzly, které se objevují před aktuálním uzlem v dokumentu, s výjimkou předků, atributů a uzlů jmenných prostorů.
self	Vybere aktuální uzel.

Tabulka 6: Přehled jednotlivých os.

Cesta může být absolutní nebo relativní. Absolutní cesta k umístění začíná lomítkem. V obou případech se cesta k umístění skládá z jednoho nebo více kroků, z nichž každý je oddělen lomítkem. Každý krok je ohodnocen na základě uzlu v aktuální množině uzlů. Krok se skládá z:

- osy, která definuje strom vztahů mezi vybranými uzly a aktuálním uzlem;
- test uzlu, který identifikuje uzel v rámci osy;
- žádný nebo více predikátů, které dále upřesňují vybranou sadu uzlů;

XPath nabízí také použití mnoha operátorů ve svých výrazech. Například: and, or, div, mod atd. [4][5]

3.7 XLink

XLink se používá k vytvoření hypertextových odkazů v XML dokumentech. Jakýkoliv element v dokumentu XML se může chovat jako odkaz. XLink je doporučení W3C.[5]

3.7.1 Syntaxe XLink

V HTML element <a> definuje hypertextový odkaz. Nicméně ve formátu XML odkazy fungují jinak. V XML dokumentech je možné použít libovolné názvy elementů. Jednoduchý příklad XLinku předkládám níže:

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

<stranky xmlns:xlink="http://www.w3.org/1999/xlink">
  <domovska_stranka xlink:type="simple" xlink:href="http://www.w3schools.com">W3Sc
hools.cz</domovska_stranka>
  <domovska_stranka xlink:type="simple" xlink:href="http://www.seznam.cz">Seznam.cz
</domovska_stranka>
</stranky>
```

Pokud je potřeba získat přístup k funkcím Xlink, je nutné se přihlásit ke jmenným prostorům Xlink. Jmenný prostor pro XLink je: "http://www.w3.org/1999/xlink". „XLink: type“ a „XLink: href“ atributy v elementu <domovska_stranka> procházejí jmenné prostory. „XLink: href“ atribut určuje adresu URL odkaz. Níže uvádím referenci atributů XLink:[5]

Atribut	Hodnota	Popis
xlink:actuate	onLoad, onRequest, other, none	Definuje, jak je připojený zdroj čten a zobrazen: • onLoad - zdroj by měl být načten a zobrazen, když je načten dokument • onRequest - zdroj není načten nebo zobrazen, předtím, než se klikne na odkaz .
xlink:href	URL	Určuje URL.
xlink:show	embed, new, replace, other, none	Určuje, kde se otevře odkaz. Výchozí hodnota je nastavena na "replace".
xlink:type	simple, extended, locator, arc, ressource, title, none	Určuje typ odkazu.

Tabulka 7: Reference atributů XLink .

3.8 XPointer

Pokud je třeba odkázat na určité místo v dokumentu nebo na jeho část, je možné použít XPointer. XPointer, celým názvem XML Pointer Language, je doplňkem jazyka XLink a umožňuje na konec URL připojit výraz, který určí pouze část celého dokumentu s daným URL. Z důvodu zachování zpětné kompatibility je možné v XPointeru používat i fragmenty, tak jako v HTML. Odkaz by mohl vypadat následovně:

<xlink:simple href=“dokument.xml#kap1>...</xlink:simple>

V HTML nelze vytvořit odkaz na určitou část dokumentu, která nemá své vlastní návěsti nelze. XPointer však nabízí způsob, jak identifikovat libovolnou část dokumentu na základě struktury dokumentu, textu elementů, obsahu atributů apod. Zápis by mohl vypadat následovně:[4]

#xpointer(„výraz“)

„výraz“ - specifikuje určité místo dokumentu, na které by byl vytvořen odkaz

3.9 XML Schema

XML schémata popisují strukturu dokumentu XML. Jazyk XML Schema je také označován jako definice schématu XML (XSD). Níže předkládám jednoduchý příklad využití XSD, které popisuje strukturu elementu „zvíře“.

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

<xs:element name="zvire">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="druh" type="xs:string"/>
      <xs:element name="jmeno" type="xs:string"/>
      <xs:element name="vyska" type="xs:string"/>
      <xs:element name="vaha" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

Element „zvíře“ je komplexní typ, protože obsahuje další elementy. Ostatní elementy jsou jednoduché typy, protože neobsahují další elementy. Účelem XML schématu je definovat stavební kameny XML dokumentu, tedy určit:

- elementy a atributy, které se mohou objevit v dokumentu;
- počet a pořadí podřízených elementů;
- datové typy pro prvky a atributy;
- výchozí a pevné hodnoty pro prvky a atributy;

XML Schema je alternativa k DTD. Největší předností XML Schema je podpora datových typů.

Mezi jeho výhody patří:

- snazší popis přípustného obsahu dokumentu;
- snazší ověření správnosti údajů;
- snazší definování omezení dat;
- jednodušší definice datových formátů;

- jednodušší převedení dat mezi různými datovými typy;
- možnost transformace pomocí XSLT;
- XML schémata jsou rozšiřitelné, protože jsou psány v XML;

Při odesílání dat od odesílatele k příjemci je důležité, aby obě části měli stejná "očekávání" o obsahu. S XML schématy může odesílatel popsat data takovým způsobem, že jim bude příjemce rozumět. Například datum "03. 11. 2016" bude v některých zemích interpretováno jako 3. Listopadu a v jiných zemích jako 11. Března. XML element s datovým typem jako je tento:

<date type = "date"> 03. 11. 2016 </ date>

zaručuje vzájemné porozumění obsahu, protože datový typ „date“ vyžaduje datum v formátu "RRRR-MM-DD", tedy rok-měsíc-den.

Tzv. well-formed XML dokument je dokumentem, který je v souladu s pravidly syntaxe XML, které jsou:

- dokument musí začínat deklarací XML;
- musí mít jeden unikátní kořenový element;
- začínající tagy musí mít odpovídající koncové tagy;
- elementy mohou obsahovat velká a malá písmena;
- všechny elementy musí být uzavřeny;
- všechny elementy musí být správně vnořené;
- všechny hodnoty atributů musí být v uvozovkách;
- pro speciální znaky musí být použity Entity.

Dokonce, i když jsou dokumenty well-formed, mohou stále obsahovat chyby a tyto chyby, mohou mít vážné následky. [4][5]

4 ODF

4.1 Verze

Verze 1.0 se stala OASIS standardem 1. 5. 2005, poté 7. 2. 2007 následovala verze 1.1 a nakonec 29. 9. 2011 verze 1.2. Mezi hlavní změny formátu OpenDocument 1.2 patří podpora digitálních podpisů, otevřeného standardu OpenFormula pro výměnu vzorců v tabulkových kalkulátorech nebo metadat založených na specifikaci RDF, což je zkratka pro Resource Description Framework, který pochází z dílny konsorcia W3C. ODF 1.2 byl schválen organizací OASIS, jejíž pracovní skupiny se na jeho vývoji podílí, již od září 2011. Teprve standardizace ISO jej však činí závazným a neměnným.[1]

Připravovaná verze ODF 1.3 by měla vylepšit zejména sledování změn. Cílem podvýboru OASIS Advanced Document Collaboration je standardizovat otevřenou technologii tvorby a úprav dokumentů pro sledování změn v reálném čase, resp. při spolupráci v reálném čase.[1]

4.2 Co je ODF?

OpenDocument Format, zkráceně ODF a celým názvem OASIS OpenDocument Format for Office Applications, v překladu OASIS otevřený formát dokumentu pro kancelářské aplikace, je otevřený souborový formát určený pro ukládání a výměnu dokumentů vytvořených kancelářskými aplikacemi. ODF zahrnuje textové dokumenty, prezentace, tabulky, grafy a databáze. Standard ODF byl vyvinut sdružením OASIS a vychází ze staršího souborového formátu používaného aplikacemi OpenOffice.org. Formáty ODF a OOXML jsou založeny na XML.

OpenDocument Format byl 3. května 2006 standardizován Mezinárodní organizací pro normalizaci jako standard ISO/IEC 26300. Rozšířený formát OpenDocument 1.2 je standardem od 1. července 2015 jako standard ISO/IEC 26300-1/2015 až ISO/IEC 26300-3/2015.

Termín „otevřený“ vychází především z toho, že formát byl vyvinut veřejně množstvím organizací, jeho specifikace je veřejně přístupná a formát může být použit libovolnou aplikací bez omezení. Účelem formátu OpenDocument je nabídnout otevřenou alternativu uzavřeným formátům, např. rozšířeným DOC, XLS a PPT využívané

kancelářským balíkem Microsoft Office. Microsoft na to reagoval vytvořením vlastního otevřeného standardu Office Open XML, otevřením vlastního dosud uzavřeného XML standardu, který je v tuto chvíli kvůli dominantnímu postavení Microsoftu v oblasti kancelářských balíků rozšířenější, ale čelí mnohé kritice např. pro svůj komplikovaný návrh. Microsoft také do svých produktů zahrnul omezenou podporu ODF.

Uživatelé OpenDocument formátu získali možnost volby dalších dodavatelů alternativních kancelářských aplikací a mohou si volit také mezi více potenciálními dodavateli řešení, například v případě ukončení vývoje nebo podpory jejich stávajících aplikací nebo v případě změny cenové politiky či licenčního ujednání pro komerční nebo nekomerční použití konkrétního softwarového řešení.

OpenDocument podporuje dva způsoby zobrazení dokumentu:

1) Jako sadu několika sub-dokumentů, kde každý z nich ukládá část celého dokumentu. Formát používá souborové přípony: .odt, .ott, .ods, .odp. Jedná se o standardní ZIP soubor s různými souborovými příponami a s definovanou strukturou sub-dokumentů. Každý ze sub-dokumentů má rozdílný kořen dokumentů a ukládá část stránky XML dokumentu. Všechny typy dokumentů, jako například textové a tabulkové dokumenty, používají stejnou množinu definic dokumentů a sub-dokumentů.
2) Jako jednoduchý XML dokument – známý také jako „Flat XML“ nebo nekomprimované soubory XML. Soubory s touto definicí nejsou obecně používány a také nejsou podporovány některými kancelářskými softwary, které tvrdí, že podporují ODF, ale například Microsoft Office aplikace verze 2007 až 2013 je obtížně rozpoznávají. Názvy přípon pro jednoduchý XML OpenDocument nejsou definovány v OpenDocument, ale běžně se používají: .xml, .fods atd.

Doporučené názvy přípon souborů a MIME typy jsou zahrnuty v oficiálním standardu. MIME typy a přípony, které jsou obsaženy v ODF specifikaci jsou použitelné pouze pro kancelářské dokumenty.[1][2][6]

4.2.1 Dokumenty

Nejvíce používané přípony pro OpenDocument dokumenty jsou:

- .odt pro textové dokumenty;
- .ods pro tabulkové dokumenty;
- .odp pro prezentační programy;
- .odg pro grafické dokumenty.

Přípony jsou snadno zapamatovatelné, jelikož se vždy používá „.od“ jako zkratka pro OpenDocument, a třetí znak reprezentuje specifický typ souboru, jako je t pro textový soubor. Níže předkládám seznam typů dokumentů, představující typy souborů, doporučené přípony souborů a MIME typy:[1][2]

Dokument	Přípona	MIME Type
textový dokument	.odt	application/vnd.oasis.opendocument.text
tabulka	.ods	application/vnd.oasis.opendocument.spreadsheet
prezentace	.odp	application/vnd.oasis.opendocument.presentation
grafika	.odg	application/vnd.oasis.opendocument.graphics
graf	.odc	application/vnd.oasis.opendocument.chart
matematický výraz	.odf	application/vnd.oasis.opendocument.formula
databáze	.odb	application/vnd.oasis.opendocument.database
obrázek	.odi	application/vnd.oasis.opendocument.image
hlavní textový dokument	.odm	application/vnd.oasis.opendocument.text-master

Tabulka 8: Přehled přípon a MIME typů pro formáty OpenDocument.

4.2.2 Šablony

OpenDocument také podporuje sada typů šablon. Šablony reprezentují naformátované informace, zahrnující styly, pro dokumenty, bez samotného obsahu. Doporučené názvy přípon začínají s „.ot“, což značí OpenDocument template, a kde poslední znak indikuje šablonu, například pro text znak t.[1]

Níže předkládám podporovanou sadu:

Dokument	Přípona	MIME Type
šablona textového dokumentu	.ott	application/vnd.oasis.opendocument.text-template
šablona tabulky	.ots	application/vnd.oasis.opendocument.spreadsheet-template
šablona prezentace	.otp	application/vnd.oasis.opendocument.presentation-template
šablona grafiky	.otg	application/vnd.oasis.opendocument.graphics-template
šablona grafu	.otc	application/vnd.oasis.opendocument.chart-template
šablona matematického výrazu	.otf	application/vnd.oasis.opendocument.formula-template
šablona obrázku	.oti	application/vnd.oasis.opendocument.image-template
šablona webové stránky	.oth	application/vnd.oasis.opendocument.text-web

Tabulka 9: Přehled přípon a MIME typů pro šablony.

4.3 Vlastnosti ODF

Oficiální OpenDocument standard verze 1.0, který byl vydán 1. května 2005, definuje vlastnosti OpenDocumentů.[1]

4.3.1 Metadata

OpenDocument podporuje ukládání metadat (data o datech) sadou předdefinovaných elementů metadat a umožňuje také uživateli definovat a upravovat metadata. Formát předefinovává následující pole metadat: generátor, titul, popis, předmět, klíčová slova, původní tvůrce, tvůrce, datum a čas vytvoření, datum a čas modifikace, šablona dokumentů, automatické načtení, chování odkazů, jazyk, cykly úprav, dokumentové statistiky, doba trvání úprav.[1]

4.3.2 Obsah

OpenDocument Format podporuje všechny možnosti úprav dokumentů. Je možné používat například nadpisy různých úrovní, seznamy různých druhů, číslované odstavce a sledování změn, a další funkce jsou samozřejmě podporovány také. Sekvence stránek a sekce atributů mohou být použity ke kontrole zobrazení textu. Hypertextové odkazy, záložky a klasické odkazy jsou také podporovány. OpenDocument Format implementuje tabulkové procesory jako soubory tabulek a nabízí rozsáhlé možnosti pro jejich formátování. OpenDocument také podporuje databáze, filtry a data piloty, které jsou v Microsoft Excel známé jako kontingenční tabulky. Grafický formát podporuje reprezentaci vektorové grafiky, ve které je definována každá vrstva souboru. Ke kreslení obrazců jsou k dispozici různé tvary, například obdélníky, čáry, křivky, mnohoúhelníky, kružnice, elipsy a konektory. 3D tvary jsou také k dispozici, ale pouze pro použití v rámci kancelářského balíku a nikoliv pro video či tvorbu 3D scén.

Prezentace jsou samozřejmě také podporovány. Uživatelé mohou použít animace, včetně použití multimédií, různých tvarů či úprav textu a ostatních úprav. Grafy definují jak graficky vyjádřit číselná data. Podporují tituly, podtituly, zápatí a legendu vysvětlující graf. Grafy mohou být různého druhy, například koláčové grafy, sloupcové grafy a další. Formuláře jsou speciálně podporovány, a to na základě standardu stávajících XForms. [1][2][3][6]

4.3.3 Objekty

Dokument v OpenDocument formátu může obsahovat dva typy objektů:

1) Objekty, které jsou znázorněny jako OpenDocument (vzorce, grafy, tabulky, textové dokumenty, grafika a prezentace);
2) Objekty, které nemají XML representaci. Tyto objekty jsou znázorněni binárně. Příkladem pro tento druh objektů jsou OLE objekty.

Používáním objektů OLE, celým názvem Microsoft Object Linking and Embedding neboli Propojování a vkládání objektů, omezuje součinnost, protože tyto objekty nejsou obecně podporovány v programech pro prohlížení a úpravu souborů. Pokud software, který umí pracovat s OLE objekty, není dostupný, tak je objekt obvykle nahrazen obrázkem, bitmapovým zobrazením objektu nebo není zobrazeno nic.[1]

4.3.4 Formátování

Stylů a formátovacích tlačítek je mnoho a poskytují nepřeberné možnosti úprav. Rozvržení stránky je kontrolováno množstvím vlastností. Ty zahrnují velikost stránky, číselný formát, orientaci tisku, výplně, stíny, pozadí, číslování stránek, měřítka, maximální výška, poznámka pod čarou a mnoho dalších. U záhlaví či zápatí může být nastavena minimální výška, okraje, šířky, pozadí, stínu, dynamické mezery atd. Pomocí množství atributů je možné specifikovat texty, odstavce, sekce, tabulky, sloupce, seznamy. Konkrétní znaky mohou měnit fonty, velikosti. Odstavce mohou mít vlastní vertikální prostor kontrolovaný pomocí atributů.[1]

4.3.5 Tabulky

OpenDocument verze 1.2 popisuje matematické vzorce, které se zobrazují na obrazovce. Je plně schopný vyměňovat tabulková data, formáty, kontingenční tabulky a další informace, které jsou zahrnuty v tabulce. OpenDocument vyměňuje vzorce jako hodnoty atributů.[1]

4.3.6 Šifrování

Je-li OpenDocument soubor chráněn heslem, struktura souborů zůstává stejná, ale obsah souborů XML je šifrován pomocí následujícího algoritmu:[1][6]

1. Obsahy souboru jsou komprimovány pomocí tzv. algoritmu DEFLATE.
2. Kontrolní součet části komprimovaného souboru je vypočítána (SHA-1 z obsahu souboru, nebo SHA-1 z prvních 1024 bajtů souboru, nebo SHA-256 z prvních 1024 bajtů souboru) a uložena tak správnost hesla, která může být ověřena při dešifrování.
3. Otisk uživatelského zadaného hesla v kódování UTF-8 je vytvořen a předán jako součást balíčku. ODF verze 1.0 a 1.1 pouze ukládaly podporu pro SHA-1, zatímco verze 1.2 doporučuje používat SHA-256.
4. Tento otisk se používá k výrobě odvozeného klíče, který prochází klíč PBKDF2 pomocí HMAC-SHA1 libovolné délky (v ODF 1.2 - to je 16 bytů v ODF 1.1 a níže), generované generátorem náhodných čísel pro libovolný počet opakování (1024 ve výchozím nastavení ODF 1.2).
5. Náhodný generátor čísel se používá ke generování náhodné inicializace vektoru pro každý soubor.
6. Inicializační vektor a doručený klíč jsou použity k zašifrování komprimovaného souboru. ODF 1.0 a 1.1 používají Blowfish 8-bit šifru režimu zpětné vazby, zatímco ODF 1.2 ho považuje za starší algoritmus a umožňuje použít Triple DES a AES (s 128, 196 nebo 256 bitů), a to v režimu Cipher Block.

4.3.7 Struktura formátu

Soubory OpenDocument se běžně ukládají ve standardním ZIP souboru (JAR souboru), který obsahuje celou řadu souborů a adresářů. OpenDocument soubor se také může skládat pouze z jediného dokumentu XML. OpenDocument soubor je obvykle sada několika subdokumentů uvnitř ZIP souboru. OpenDocument soubor reprezentovaný jako jeden XML není obecně používaný. Díky jednoduchému mechanismu komprese, který se používá pro zabalení souborů, jsou soubory OpenDocument podstatně menší než ekvivalentní soubory DOC nebo PPT. Menší velikost je velmi důležitá pro organizace, které skladují obrovské množství dokumentů, a to po dlouhou dobu a vyměňují dokumenty pomocí pomalých připojení. Někdy je soubor nekomprimovaný a tehdy je většina dat obsažena v jednoduchém textovém souboru XML, takže nekomprimovaný obsah údajů umožňuje snadné úpravy a zpracování XML souborů. Standard umožňuje také vytvoření jediného dokumentu XML, který používá <*office: document*> jako kořenový prvek,

používaný při zpracování dokumentů. Tento standard umožňuje zahrnout adresáře pro ukládání obrázků, non-SMIL animace a další soubory, které jsou používány v dokumentu, ale nelze je vyjádřit přímo v XML. Vzhledem k otevřenosti kompresního formátu, je možné, aby uživatel k extrahování souboru s obsahem ručně upravil uvedené soubory. To umožňuje opravu poškozeného souboru nebo nízko úrovňovou manipulaci s obsahem. ZIP soubor souborů a adresářů zahrnuje následující:

- XML soubory
    - content.xml
    - meta.xml
    - settings.xml
    - styles.xml
- Other soubory
    - mimetype
- Adresáře
    - META-INF/
        - manifest.xml
    - Thumbnails/
        - thumbnail.png

Formát OpenDocument poskytuje silné oddělení obsahu, uspořádání a použití metadat. Nejzajímavější složky formátu popisuji v následujících podkapitolách. Soubory ve formátu XML jsou dále definovány pomocí Relax NG jazyku, který slouží definování schémat XML. RELAX NG je sám definován specifikací OASIS a také druhou částí mezinárodní normy ISO / IEC 19757: Schema Definition Document Languages, zkráceně DSDL.[1][2][6]

4.3.7.1 content.xml

content.xml je nejdůležitější soubor, který obsahuje skutečný obsah dokumentu, tedy kromě binárních dat, jako jsou obrázky. Základ formátu je inspirována HTML, a přestože je mnohem složitější, měl by být přiměřeně dostupný pro člověka.[1]

4.3.7.2 styles.xml

styles.xml obsahuje informace o stylu dokumentu. OpenDocument poskytuje silné použití stylů pro formátování a rozvržení. Většina informací o stylu se nachází v tomto dokumentu, ačkoli některé informace jsou také v souboru content.xml.[1]

Do stylu patří:

- Styly odstavců;
- Styly stránek;
- Styly znaků;
- Styly rámů;
- Styly seznamů.

OpenDocument format je poněkud neobvyklý v tom, že se použití stylů pro formátování nelze vyhnout. Dokonce i "manuální" formátování je realizováno prostřednictvím stylů. Aplikace vytváří dynamicky nové styly podle potřeby.[1]

4.3.7.3 meta.xml

meta.xml obsahuje metadata souboru, tedy data o datech. Například, autor, "naposledy upraveno", datum poslední změny, atd.[1]

4.3.7.4 settings.xml

settings.xml obsahuje nastavení, jako je například faktor zoom nebo pozici kurzoru. To jsou vlastnosti, které nejsou součástí obsahu nebo rozvržení.[1]

4.3.7.5 mimetype

MIME typ je jen jednořádkový soubor s mimetypem dokumentu. Jedním z důsledků je, že přípona souboru je vlastně nepodstatná část formátu. Přípona souboru se používá pouze ve prospěch uživatele.[1]

4.3.7.6 Thumbnails

Thumbnails neboli v překladu Miniatury je samostatná složka pro náhled dokumentu. Miniatura musí být uložen jako "thumbnail.png". Miniatura reprezentace dokumentu by měla být generována ve výchozím nastavení, ve chvíli, kdy je soubor

uložen. Požadovaná velikost náhledů je 128x128 pixelů. Aby byly v souladu s náhledy Thumbnail Managing Standard (TMS), miniatury musí být uloženy jako 8bit, neprokládaný PNG s plnou průhledností alfa.[1]

4.3.7.7 META-INF

META-INF je samostatná složka. Obsahuje informace o souborech obsažených v OpenDocument a je uložena v souboru XML pod názvem manifest.xml. Tento soubor je vždy uložen na následující cestě *META-INF / manifest.xml*. Hlavní části informací uložených v manifestu jsou:[1]

- seznam všech souborů v balíčku;
- typ média každého souboru v balíčku;
- pokud soubor uložený v balíčku je zašifrován, informace potřebné k dešifrování souboru jsou uloženy v souboru manifest.xml.

4.3.7.8 Pictures

Adresář Picture je samostatná složka pro obrázky obsažené v dokumentu. Tato složka není definována v OpenDocument specifikaci. Soubory v této složce mohou využívat různé formáty obrázků, v závislosti na formátu vloženého souboru. Zatímco obrazová data mohou mít libovolný formát, je doporučeno, aby bitmapové grafiky byly uloženy v PNG formátu a vektorová grafika ve formátu SVG.[1]

4.3.8 Opětovné použití existujících formátů

Podle návrhu, OpenDocument opakovaně používá existující otevřené standardy XML vždy, když jsou k dispozici a vytváří nové značky pouze tehdy, pokud žádná existující norma nemůže poskytnout potřebné funkce. OpenDocument používá podmnožinu Dublin Core pro metadata, MathML pro zobrazení vzorců, SMIL pro multimédia, XLink pro hypertextové odkazy atd. Ačkoliv není definováno opětovné použití SVG pro vektorovou grafiku, OpenDocument nepoužívá SVG kompatibilní s vektorovou grafikou v rámci ODF formátu, ale zahrnuje také non-SVG grafiky.[1][6]

5 XML a ODF v praxi

Tato část je rozdělena na dvě samostatné sekce. V první části se budu zabývat vytvořením XML souboru, který postupně převedu na uživatelsky přívětivou formu, pomocí jazyka XSL. Touto transformací se budu snažit poukázat na snadnost převodu a podobnost se stylováním stránek HTML pomocí CSS. Ve druhé části budu vytvářet makro pracující s OpenDocumet Format. Jedná se o makro převádějící dokument v LibreOffice do formátu PDF.

5.1 XML v praxi

V této sekci vytvořím zkušební XML soubor, který bude zároveň obsahovat DTD. Použiji k tomu open source software a tím bude XML Copy Editor. Po vytvoření a popsání tohoto souboru ho budu převádět do uživatelsky přívětivé formy pomocí technologie XSL.

5.1.1 XML soubor

XML soubor obsahuje data získaná z obchodního rejstříku. U každé sekce kódu uvádím také komentář.

```
<!-- Zde je uvedena deklarace verze a použité kódování. Na druhém řádku je připojen
soubor XSL.  -->
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="rejstrik.xsl"?>

<!-- Zde je uvedena Definice Typu Dokumentu, která deklaruje jednotlivé elementy. -->
<!DOCTYPE registr
[
<!ELEMENT registr (subjekt+)>
<!ELEMENT subjekt
(ico,nazev,pravni_forma,adresa,zakladni_kapital,datum_zapisu,predmet_podnikani)>
<!ATTLIST subjekt
        id CDATA #REQUIRED>
<!ELEMENT ico (#PCDATA)>
<!ELEMENT nazev (#PCDATA)>
<!ELEMENT pravni_forma (#PCDATA)>
<!ELEMENT adresa (ulice,mesto,psc)>
<!ELEMENT ulice (#PCDATA)>
<!ELEMENT mesto (#PCDATA)>
<!ELEMENT psc (#PCDATA)>
<!ELEMENT zakladni_kapital (#PCDATA)>
<!ELEMENT datum_zapisu (#PCDATA)>
```

```
<!ELEMENT predmet_podnikani (#PCDATA)>
]>

<!-- Zde je samotný registr, který se skládá z jednotlivých subjektů označených vlastním id.
-->
<registr>

   <subjekt id="001">
      <ico>27604977</ico>
      <nazev>Google Czech Republic, s.r.o.</nazev>
      <pravni_forma>Společnost s ručením omezeným</pravni_forma>
      <adresa>
         <ulice>Stroupežnického</ulice>
         <mesto>Praha 5</mesto>
         <psc>15000</psc>
      </adresa>
      <zakladni_kapital>2 800 000 Kč</zakladni_kapital>
      <datum_zapisu>27.9.2006</datum_zapisu>
      <predmet_podnikani>Výroba, obchod a sluzby.</predmet_podnikani>
   </subjekt>

      <subjekt id="002">
      <ico>26168685</ico>
      <nazev>Seznam.cz, a.s.</nazev>
      <pravni_forma>Akciová společnost</pravni_forma>
      <adresa>
         <ulice>Radlická 3294/10</ulice>
         <mesto>Praha-Smíchov</mesto>
         <psc>150 00</psc>
      </adresa>
      <zakladni_kapital>3.7 miliony Kč</zakladni_kapital>
      <datum_zapisu>5. 4 2000</datum_zapisu>
      <predmet_podnikani>Výroba, obchod a služby.</predmet_podnikani>
   </subjekt>

      <subjekt id="003">
      <ico>28247132</ico>
      <nazev>JOBS CZ, s.r.o.</nazev>
      <pravni_forma>Společnost s ručením omezeným</pravni_forma>
      <adresa>
         <ulice>Klečákova 347/5</ulice>
         <mesto>Praha 9</mesto>
         <psc>190 00</psc>
      </adresa>
      <zakladni_kapital>200 000 Kč</zakladni_kapital>
      <datum_zapisu>11. 2 2008</datum_zapisu>
      <predmet_podnikani>Zprostředkování obchodu a služeb.</predmet_podnikani>
   </subjekt>
```

```
<subjekt id="004">
   <ico>62419641</ico>
   <nazev>TRASK SOLUTIONS, a.s.</nazev>
   <pravni_forma>Akciová společnost</pravni_forma>
   <adresa>
      <ulice> Na Pankráci 1724/129</ulice>
      <mesto>Praha 4 - Nusle</mesto>
      <psc>PSČ 140 00</psc>
   </adresa>
   <zakladni_kapital>20 000 Kč</zakladni_kapital>
   <datum_zapisu>29. 11. 1994</datum_zapisu>
   <predmet_podnikani>Vyroba, obchod a služby.</predmet_podnikani>
</subjekt>

   <subjekt id="005">
   <ico>24297381</ico>
   <nazev>Aqua Digital, s.r.o.</nazev>
   <pravni_forma>Společnost s ručením omezeným</pravni_forma>
   <adresa>
      <ulice>Tigridova 1501/6</ulice>
      <mesto>Praha - Michle</mesto>
      <psc>140 00</psc>
   </adresa>
   <zakladni_kapital>200 000 Kč</zakladni_kapital>
   <datum_zapisu>25. 4. 2012</datum_zapisu>
   <predmet_podnikani>Výroba, obchod a služby.</predmet_podnikani>
</subjekt>

   <subjekt id="006">
   <ico>01507729</ico>
   <nazev>SOFTIM.CZ spol. s r. o.</nazev>
   <pravni_forma>Společnost s ručením omezeným</pravni_forma>
   <adresa>
      <ulice>Karla Engliše 3201/6</ulice>
      <mesto>Praha 5 - Smíchov</mesto>
      <psc>150 00</psc>
   </adresa>
   <zakladni_kapital>200 000 Kč</zakladni_kapital>
   <datum_zapisu>22. 3. 2013</datum_zapisu>
   <predmet_podnikani>Výroba, obchod a služby.</predmet_podnikani>
</subjekt>

   <subjekt id="007">
   <ico>25751417</ico>
   <nazev>Národní vzdělávací fond, o.p.s.</nazev>
   <pravni_forma>Obecně prospěšná společnost</pravni_forma>
   <adresa>
      <ulice>Opletalova 25,</ulice>
```

```
      <mesto>Praha 1</mesto>
      <psc>110 00</psc>
    </adresa>
    <zakladni_kapital>-</zakladni_kapital>
    <datum_zapisu>2. 4. 1999</datum_zapisu>
    <predmet_podnikani>Společnost veřejnosti poskytuje obecně prospěšné
služby.</predmet_podnikani>
  </subjekt>

    <subjekt id="008">
    <ico>00222640</ico>
    <nazev>PTV, spol. s r.o.</nazev>
    <pravni_forma>Společnost s ručením omezeným</pravni_forma>
    <adresa>
      <ulice>Československé armády 23</ulice>
      <mesto>Hostivice</mesto>
      <psc>253 01</psc>
    </adresa>
    <zakladni_kapital>1 950 000 Kč</zakladni_kapital>
    <datum_zapisu>          26. 2. 1991</datum_zapisu>
    <predmet_podnikani>Výroba, obchod a služby.</predmet_podnikani>
  </subjekt>

    <subjekt id="009">
    <ico>45192944</ico>
    <nazev>Model Obaly a.s.</nazev>
    <pravni_forma>Akciová společnost</pravni_forma>
    <adresa>
      <ulice>Těšínská 2675/102</ulice>
      <mesto>Opava - Předměstí,</mesto>
      <psc>746 01</psc>
    </adresa>
    <zakladni_kapital>31 000 000 Kč</zakladni_kapital>
    <datum_zapisu>16. 6. 1992</datum_zapisu>
    <predmet_podnikani>Pronájem nemovitostí, bytů a nebytových prostor bez
poskytování jiných než základních služeb.</predmet_podnikani>
  </subjekt>
</registr>
```

5.1.2 XML soubor s XSLT

Použitím XSLT se velice snadno docílí grafického vzhledu původního XML dokumentu. Zde předkládám vytvořené styly pro daný XML soubor. Styly jsou vytvořeny pomocí XSLT a CSS stylů.

```
<!-- Deklarace použitého kódování. -->
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

<!-- Použití XSL šablony, kde znak „/“ označuje, že bude aplikována na všechny elementy. Jednotlivé styly jsou připojeny pomocí slova „style“ a hodnoty pomocí atributů. -->
<xsl:template match="/">
  <html>
  <body style="font-family:Arial;font-size:12pt;text-align:center;background-color:#F5F5F5">
  <h1>Obchodní rejstřík</h1>
  <h3 style="text-align:left">Co je obchodní rejstřík</h3>
<p style="text-align:left">Obchodní rejstřík je veřejný registr právnických a fyzických osob. Obchodní rejstřík je veden příslušným rejstříkovým soudem. Obchodní rejstřík na internetu umožňuje rychlé hledání údajů, které mají informativní charakter. Pro právní úkony je třeba ověřený výpis z obchodního rejstříku, který je možné získat u libovolného rejstříkového soudu, ale i v jakékoli notářské kanceláři, na obecním úřadě a na pobočkách pošty se službou Czech POINT</p>
<h3 style="text-align:left">Kdo se zapisuje do obchodního rejstříku</h3>
<p style="text-align:left">Do obchodního rejstříku se zapisují podnikatelské subjekty - obchodní společnosti a družstva, podnikající fyzické osoby, které o to požádají, nebo ze zákona ty, které mají podle výše příjmů tuto povinnost. Dále některé zahraniční osoby a další osoby, pokud mají podle právních předpisů zápis do obchodního rejstříku povinný.</p>
    <table border = "1" style="padding:10px;text-align:center">
      <tr style="background-color:teal;color:white;font-weight:bold">
        <td>IČO</td>
        <td>Název</td>
        <td>Právní forma</td>
        <td>Ulice</td>
        <td>Město</td>
        <td>PSČ</td>
        <td>Základní kapitál</td>
        <td>Datum zápis</td>
        <td>Předmět podnikání</td>
      </tr>
      <xsl:for-each select="registr/subjekt">
        <tr>
          <td> <xsl:value-of select="ico"/></td>
           <td><xsl:value-of select="nazev"/></td>
```

```
            <td><xsl:value-of select="pravni_forma"/></td>
            <td><xsl:value-of select="adresa/ulice"/></td>
             <td><xsl:value-of select="adresa/mesto"/></td>
             <td><xsl:value-of select="adresa/psc"/></td>
             <td><xsl:value-of select="zakladni_kapital"/></td>
             <td><xsl:value-of select="datum_zapisu"/></td>
             <td><xsl:value-of select="predmet_podnikani"/></td>
          </tr>
       </xsl:for-each>
     </table>
     </body>
     </html>

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

Díky těmto stylům vytvořených v XSL souboru bude mít dokument následující podobu:

Obchodní rejstřík

Co je obchodní rejstřík

Obchodní rejstřík je veřejný registr právnických a fyzických osob. Obchodní rejstřík je veden příslušným rejstříkovým soudem. Obchodní rejstřík na internetu umožňuje rychlé hledání údajů, které mají informativní charakter. Pro právní úkony je třeba ověřený výpis z obchodního rejstříku, který je možné získat u libovolného rejstříkového soudu, ale i v jakékoli notářské kanceláři, na obecním úřadě a na pobočkách pošty se službou Czech POINT

Kdo se zapisuje do obchodního rejstříku

Do obchodního rejstříku se zapisují podnikatelské subjekty - obchodní společnosti a družstva, podnikající fyzické osoby, které o to požádají, nebo ze zákona ty, které mají podle výše příjmů tuto povinnost. Dále některé zahraniční osoby a další osoby, pokud mají podle právních předpisů zápis do obchodního rejstříku povinný.

IČO	Název	Právní forma	Ulice	Město	PSČ	Základní kapitál	Datum zápis	Předmět podnikání
27604977	Google Czech Republic, s.r.o.	Společnost s ručením omezeným	Stroupežnického	Praha 5	15000	2 800 000 Kč	27.9.2006	Výroba, obchod a sluzby.
26168685	Seznam.cz, a.s.	Akciová společnost	Radlická 3294/10	Praha-Smíchov	150 00	3.7 miliony Kč	5. 4 2000	Výroba, obchod a služby.
28247132	JOBS CZ, s.r.o.	Společnost s ručením omezeným	Klečákova 347/5	Praha 9	190 00	200 000 Kč	11. 2 2008	Zprostředkování obchodu a služeb.
62419641	TRASK SOLUTIONS, a.s.	Akciová společnost	Na Pankráci 1724/129	Praha 4 - Nusle	PSČ 140 00	20 000 Kč	29. 11. 1994	Vyroba, obchod a služby.
24297381	Aqua Digital, s.r.o.	Společnost s ručením omezeným	Tigridova 1501/6	Praha - Michle	140 00	200 000 Kč	25. 4. 2012	Výroba, obchod a služby.

Obrázek 1: Zobrazení XML s XSL.

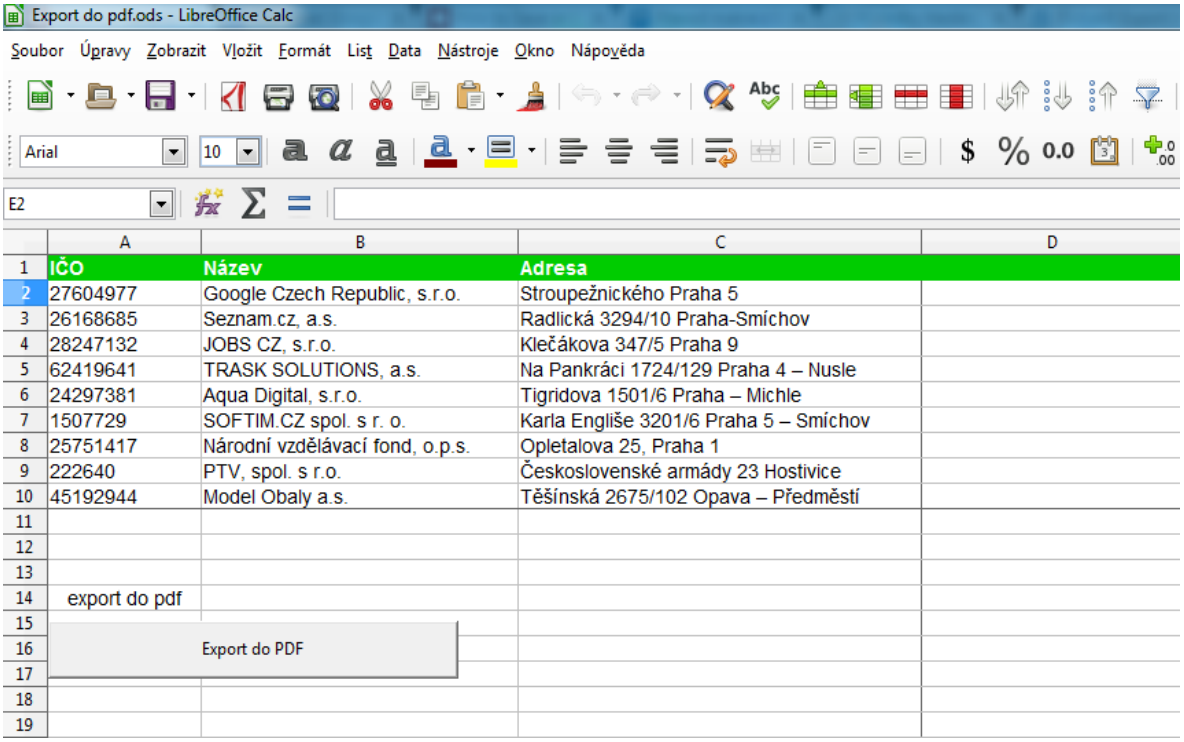
5.1.3 Zhodnocení XML

Převedení prostého XML souboru pomocí technologie XSLT je velice snadné. Podle mě je téměř snadnější než tvorba samotného dokumentu XML. Šablona XSLT stačí připojit k XML a poté již stačí využívat tagů používaných ve známějším CSS. Pokud tedy uživatel zná alespoň základy kaskádových stylů, není problém stylovat XML a získat tak uspokojivé výsledky. XML umožňuje základní dokument konvertovat do jiných výstupů velmi snadno a v tom spatřuji jeho velkou výhodu.

5.2 ODF makro

V této sekci vytvořím makro pracující s ODF. Makro exportuje tabulkový dokument vytvořený v programu LibreOffice do formátu PDF. Dané makro exportuje vybranou část listu do formátu PDF.

Pro export jsem vytvořil zkušební dokument v programu Calc. Dokument obsahuje opět hodnoty z obchodního rejstříku a má následující tvar:



	A	B	C	D
1	IČO	Název	Adresa	
2	27604977	Google Czech Republic, s.r.o.	Stroupežnického Praha 5	
3	26168685	Seznam.cz, a.s.	Radlická 3294/10 Praha-Smíchov	
4	28247132	JOBS CZ, s.r.o.	Klečákova 347/5 Praha 9	
5	62419641	TRASK SOLUTIONS, a.s.	Na Pankráci 1724/129 Praha 4 – Nusle	
6	24297381	Aqua Digital, s.r.o.	Tigridova 1501/6 Praha – Michle	
7	1507729	SOFTIM.CZ spol. s r. o.	Karla Engliše 3201/6 Praha 5 – Smíchov	
8	25751417	Národní vzdělávací fond, o.p.s.	Opletalova 25, Praha 1	
9	222640	PTV, spol. s r.o.	Československé armády 23 Hostivice	
10	45192944	Model Obaly a.s.	Těšínská 2675/102 Opava – Předměstí	
11				
12				
13				
14	export do pdf			
15				
16				
17				
18				
19				

Obrázek 2: Ukázkový dokument.

.

Výsledné PDF by mělo obsahovat hodnoty v buňkách A1 až C10. Název PDF by se měl skládat z názvu aktuálního listu, aktuálního data a pevnou hodnotu zadanou v poli A14. Makro je rozděleno na dvě části. První dynamicky připravuje název PDF souboru na základě určité hodnoty. Následující kus kódu popisuje výše zmíněnou funkčnost:

```
dim fileName
  fileName = ThisComponent.Sheets(0).Name & _
  "_ratie_" & ThisComponent.Sheets(0).getCellRangebyName("B5").String & _
  "_" & Replace(Date,"/","-") & ".pdf"
```

Získání názvu z aktivního listu se provede následující částí kódu:

```
fileName = ThisComponent.getCurrentController.getActiveSheet.Name & _
```

Nyní vytvořím objekt „Frame“ a dispatcher, které využívají službu: „com.sun.star.frame.DispatchHelper“, která umožní exportovat PDF pomocí funkce. Následující částí kódu se vytvoří prázdný dokument s danou přístupovou cestou, kterou je potřeba nastavit. Nakonec se dokument uzavře.

```
path ="file:///c:/users/dangerousdonkey/desktop/ " & fileName
  Open path For Append As #1
  Close #1
```

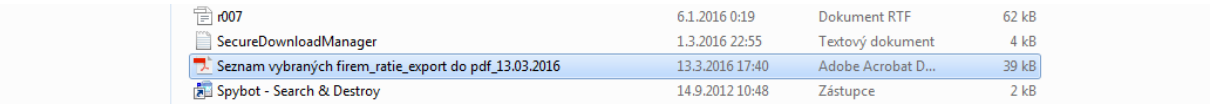
Nakonec vložím obsah do dokumentu. K tomu se využijí filtry. Filtry jsou typy a dvojice hodnot, které lze vložit do polí a mohou být předány funkci „executeDispatch“ jako argument. V tomto případě se musí projít název souboru a rozsahu listu, který má být vytištěn v PDF. Zde je potřeba použít zabudovanou vlastnost filtru „FilterData“:

```
document = ThisComponent.CurrentController.Frame
  dispatcher = createUnoService("com.sun.star.frame.DispatchHelper")
  oSheet =  ThisComponent.CurrentController.getActiveSheet()
  oCellRange = oSheet.getCellRangeByName("A1:C10")

  dim aFilterData(0) as new com.sun.star.beans.PropertyValue
  aFilterData(0).Name = "Selection"
  aFilterData(0).Value = oCellRange
  dim args1(1) as new com.sun.star.beans.PropertyValue

  args1(0).Name = "URL"
  args1(0).Value = "file:///c:/users/dangerousdonkey/desktop/" & fileName
  args1(1).Name = "FilterData"
  args1(1).Value = aFilterData()
```

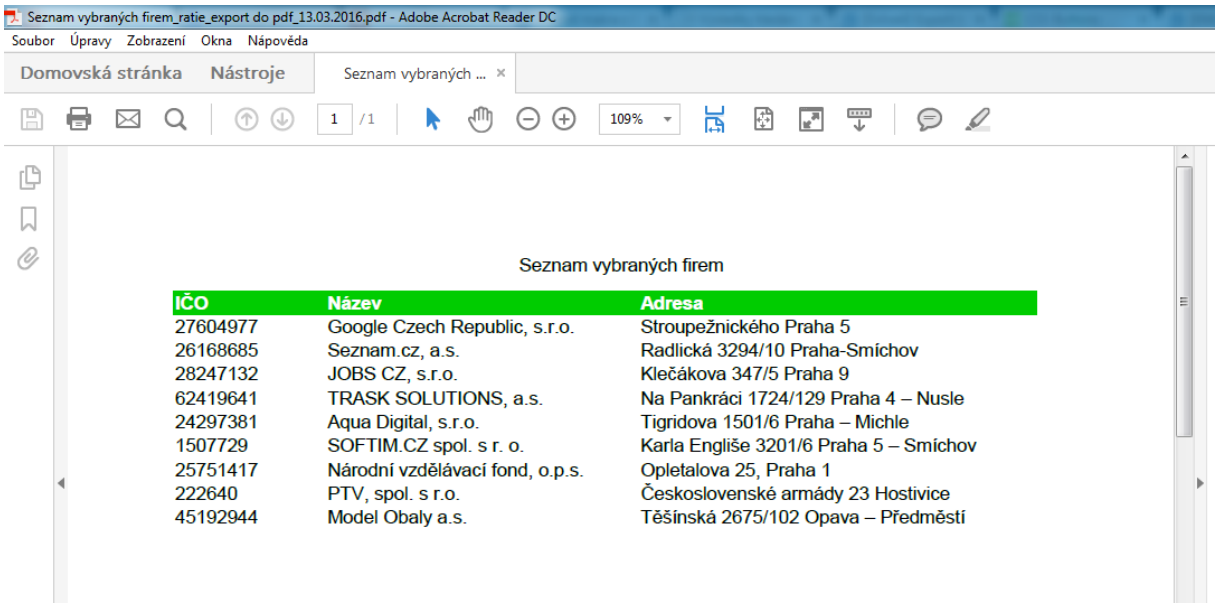
Nyní se předá tento argument funkci „executeDispatch“:

```
dispatcher.executeDispatch(document, ".uno:ExportDirectToPDF", "", 0, args1())
```

Po stisknutí tlačítka „Export do pdf“ a soubor se uloží na definované cestě. Zde je soubor ve formátu PDF, který byl vytvořen s daným makrem:



Obrázek 3: Ukázka názvu vytvořeného souboru.



Obrázek 4: Ukázka výsledného dokumentu.

Kompletní makro vypadá následovně:

```
Sub ExportDoPDF()
  dim document as object
  dim dispatcher as object
  dim fileName
  fileName = ThisComponent.Sheets(0).Name & _
                                        "_ratie_" &
ThisComponent.Sheets(0).getCellRangebyName("A14").String & _
                                        "_" & Replace(Date,"/","-") & ".pdf"
  document = ThisComponent.CurrentController.Frame
  dispatcher = createUnoService("com.sun.star.frame.DispatchHelper")

  oSheet =  ThisComponent.CurrentController.getActiveSheet()
  oCellRange = oSheet.getCellRangeByName("A1:C10")

  dim aFilterData(0) as new com.sun.star.beans.PropertyValue
  aFilterData(0).Name = "Selection"
```

```
  aFilterData(0).Value = oCellRange

  path ="file:///c:/users/dangerousdonkey/desktop/" & fileName
  Open path For Append As #1
  Close #1

  dim args1(1) as new com.sun.star.beans.PropertyValue
  args1(0).Name = "URL"
  args1(0).Value = "file:///c:/users/dangerousdonkey/desktop/" & fileName
  args1(1).Name = "FilterData"
  args1(1).Value = aFilterData()

  dispatcher.executeDispatch(document, ".uno:ExportDirectToPDF", "", 0, args1())
End Sub
```

5.2.1 Zhodnocení ODF

LibreOffice podporuje čtyři typy programovacích jazyků a to: LibreOffice Basic, Javascript, BeanShell a Python. Vybral jsem si vytvořit makro v LibereOffice Basic a to z důvodu existence dostatečného množství tutoriálů. Pro výběr makra jsem měl alternativu vytvoření importu z Aresu, tedy z registru ekonomických subjektů. Z tohoto plánu jsem nakonec upustil, jelikož jsem se potýkal s častými problémy a to především s nemilým zablokováním mé ip adresy při dotazech na ičo subjektu. Po několikerém přečtení dokumentace jsem dokázal vytvořit fungující makro, které po stisknutí tlačítka exportuje vybraný obsah do PDF. Makro by šlo samozřejmě různě modifikovat, například tak, aby oblast pro export byl celý sešit nebo aby se exportoval jen výběr provedený pomocí počítačové myši.

6 Závěr

V teoretické části byly popsány základy formátů XML a OpenDocument. V praktické části bylo provedeno převedení XML pomocí XSLT do HTML výstupu. Rovněž bylo vytvořeno makro pracující s ODF, konkrétně makro pro export PDF.

Převod XML do HTML pomocí XSLT je velice snadný a přináší prostému souboru XML grafickou podobu. Snadno jsem v mém příkladu docílil publikovatelné formy tabulky a textu se záznamy z obchodního rejstříku. XML umožňuje základní dokument konvertovat do různých výstupů a v tom spatřuji jeho hlavní přednost. Na druhou stranu je XML technologie přímo závislá na použití dodatečných technologií, jako je XSLT, XPath, XLink, XPointer a XML Schema. Díky těmto rozšířením je možné získat z XML maximum.

Při tvorbě makra v rámci ODF jsem se ho rozhodl vytvořit v programovacím jazyce LibereOffice Basic. Výsledkem je fungující makro, které po stisknutí tlačítka exportuje vybraný obsah do PDF. Makro se dá napsat velmi srozumitelně s možností dalšího rozšíření. S prací v LibreOffice jsem byl spokojen, jen mě překvapila nízká stabilita programu při vytváření makra. Na druhou stranu nabízí téměř veškeré možnosti jako konkurenční software a přitom je zdarma. Nevýhody spatřuji pouze v nižší stabilitě, zdrojích podpory a problémech při exportu dokumentů.

7 Seznam použitých zdrojů

1. EISENBERG, J., D. OASIS OpenDocument Essentials. Lulu.com, 2006. 303 pages. ISBN 1411668324.
2. FIRELLI, S. Automatic Digital Document Processing and Management: Problems, Algorithms and Techniques (Advances in Computer Vision and Pattern Recognition). Springer 2011. 308 pages. ISBN 0857291971.
3. ŠTĚDROŇ, Bohumír. Open Source software: ve veřejné správě a soukromém sektoru. Praha: Grada, 2009. 128 s. ISBN 978-80-247-3047-9.
4. KOSEK, Jiří. XML pro každého. Praha: Grada Publishing, 2000. 164 s. ISBN 80-7169-860-1.
5. W3 Schools. http://www.w3schools.com/ (Cit. 3. 3. 2016).
6. OpenDocument Format. http://www.opendocumentformat.org/ (Cit. 21. 1. 2016).

8 Přílohy

Součástí bakalářské práce je elektronická příloha, která je na přiloženém CD.